Secure collaborative management platform applied to IoT Networks

Okba Ben Atia

Année 2024 N° d'ordre : (attribué par le SCD)

UNIVERSITÉ DE HAUTE-ALSACE
UNIVERSITÉ DE STRASBOURG

THESE

Pour l'obtention du grade de

DOCTEUR DE L'UNIVERSITÉ DE HAUTE-ALSACE

ECOLE DOCTORALE : Mathématiques, sciences de l'information et de l'ingénieur (ED 269)

Discipline : **Institut de Recherche en Informatique Mathématiques Automatique Signal** (**IRIMAS**)

Spécialité : **Informatique**

Présentée et soutenue publiquement

Par

Okba BEN ATIA

Le 25/06/2024

Secure collaborative management platform applied to IoT Networks

---

Sous la direction du Prof. **Pascal LORENZ**

Jury :

Prof. Lyes KHOUKHI, Université de ENSICAEN, Normandie	(**Rapporteur)**
Prof. Jalel BEN OTHMAN, Université de Paris-Saclay	(**Rapporteur)**
Prof. Hacène FOUCHAL, Université de Reims Champagne-Ardenne	**(Examinateur)**
Prof. Oumaya BAALA CANALDA, Université de Technologie de Belfort-Montbéliard	**(Examinatrice)**
Associate Prof. Ismail BENNIS, Université de Haute Alsace	**(Invité)**
Prof. Pascal LORENZ, Université de Haute Alsace	**(Directeur de thèse)**
Prof. Jaafar GABER, Université de Technologie de Belfort-Montbéliard	**(Co- Directeur de thèse)**

Abstract

Federated Learning (FL) is an innovative Machine Learning (ML) approach that allows clients to collaboratively train a model while keeping their data private. Despite its benefits, FL is vulnerable to poisoning attacks, particularly, Label-flipping and Backdoor attacks, where malicious clients intentionally tamper with data to compromise the model's privacy and security. This need has resulted in many research efforts to propose efficient FL defense solutions and improve the performance of existing ones.

This thesis addresses the problem of malicious model detection in FL system as opposed to poisoning attacks for Internet of Things (IoT) networks. First, we provide an extensive literature review and a comparative study of recent malicious model detection techniques in FL system used in the IoT context. We also propose a Federated Learning Secure Layered Adaptation and Behavior (FLSecLAB) framework. It is designed to fortify the FL system against poisoning attacks with different simulation scenarios in IoT networks, integrating both our novel and established defense mechanisms into a versatile platform for security assessment. FLSecLAB represents a significant advancement toward secure FL deployments, offering comprehensive customization for evaluating defenses across various datasets and performance metrics. Secondly, we propose an enhanced malicious model detection for the FL system, which dynamically selects an optimal threshold for detecting malicious models, specifically targeting Label-flipping attacks. Furthermore, we present a novel and scalable solution for detecting and excluding malicious clients using entropy information and an adaptive threshold. This aims to enhance the robustness of malicious model detection and improve the accuracy and computational cost against Label-flipping attacks. We explore further into the complex scenarios inherent within the FL system and propose a novel malicious model detection aimed at countering both Label-flipping and Backdoor attacks simultaneously. By harnessing the intrinsic robustness and simplicity of the implementation of two outlier detection algorithms, we establish a reinforced, more scalable multi-layered defense approach. Additionally, We propose an adaptive model for detecting malicious clients against Label-flipping and Backdoor attacks, addressing the complex and nuanced challenges that arise in real-world FL system scenarios. This approach is specifically tailored to tackle the issue of Non-Independent and Identically Distributed (Non-IID) data, a common hurdle in FL system that adds layers of complexity and heterogeneity to model training and evaluation.

We assess and validate our proposed approaches through various simulation scenarios using different clients' numbers and several dataset types including MNIST, MNIST-Non-IID, Fashion-MNIST, CIFAR10, CIFAR10-Non-IID, and IMDB datasets, and we compare them with existing approaches from the literature. The results show the effectiveness of our approaches in enhancing various malicious detection performance metrics such as Accuracy (ACC), Loss Rate (LR), Attack Success Rate (ASR), CPU run-time, Recall, and Precision.

Keywords: Federated Learning (FL), Machine Learning (ML), Internet of Things (IoT), Genetic Algorithm (GA), Local Outlier Factor (LOF), Entropy, Accuracy (ACC), Loss Rate (LR), Attack Success Rate (ASR).

Résumé

L'apprentissage fédéré (FL) est une approche innovante en apprentissage automatique (ML) qui permet aux clients de former collaborativement un modèle tout en gardant leurs données privées. Malgré ses avantages, le FL est vulnérable aux attaques de poisoning, en particulier, les attaques de changement d'étiquettes et les attaques de porte dérobée, où des clients malveillants altèrent intentionnellement les données pour compromettre la confidentialité et la sécurité du modèle. Ce besoin a résulté en un grand nombre d'efforts de recherche visant à proposer des solutions de défense FL efficaces et à améliorer les performances de celles existantes.

Cette thèse aborde le problème de la détection des modèles malveillants dans le système FL face aux attaques de poisoning pour les réseaux de l'Internet des Objets (IoT). Premièrement, nous fournissons une revue de littérature exhaustive et une étude comparative des techniques récentes de détection de modèles malveillants dans le système FL utilisé dans le contexte de l'IoT. Nous proposons également un cadre d'adaptation et de comportement sécurisé en apprentissage fédéré (FLSecLAB). Il est conçu pour fortifier le système FL contre les attaques de poisoning avec différents scénarios de simulation, intégrant à la fois des mécanismes de défense nouveaux et établis dans une plateforme versatile pour l'évaluation de la sécurité dans les réseaux IoT. FLSecLAB représente une avancée significative vers le déploiement sécurisé du FL, offrant une personnalisation complète pour évaluer les défenses à travers divers jeux de données et métriques de performance. Deuxièmement, nous proposons une détection améliorée de modèles malveillants pour le système FL, qui sélectionne dynamiquement un seuil optimal pour détecter les modèles malveillants, ciblant spécifiquement les attaques de changement d'étiquettes. De plus, nous présentons une solution novatrice et évolutive pour détecter et exclure les clients malveillants en utilisant des informations d'entropie et un seuil adaptatif. Cela vise à améliorer la robustesse de la détection de modèles malveillants et à améliorer la précision et le coût computationnel contre les attaques de changement d'étiquettes. Nous explorons plus loin dans les scénarios complexes inhérents au système FL et proposons une détection de modèles malveillants novatrice visant à contrer simultanément les attaques de changement d'étiquettes et de porte dérobée. En exploitant la robustesse inhérente et la simplicité de mise en œuvre de deux algorithmes de détection d'anomalies, nous établissons une approche de défense multi-couches renforcée et plus évolutive. De plus, nous proposons un modèle adaptatif pour détecter les clients malveillants contre les attaques de changement d'étiquettes et de porte dérobée, abordant les défis complexes et nuancés qui surgissent dans les scénarios réels. Cette approche est spécifiquement conçue pour s'attaquer au problème des données Non-Indépendantes et Identiquement Distribuées (Non-IID), un obstacle courant dans les systèmes FL qui ajoute des couches de complexité et d'hétérogénéité à l'entraînement et à l'évaluation des modèles. Nous évaluons et validons nos approches proposées à travers divers scénarios de simulation en utilisant différents nombres de clients et plusieurs types de jeux de données incluant MNIST, Fashion-MNIST, MNIST-Non-IID, CIFAR10, CIFAR10-Non-IID, et les jeux de données IMDB, et nous les comparons avec les approches existantes de la littérature. Les résultats montrent l'efficacité de nos approches pour améliorer diverses métriques de performance de détection malveillante telles que le taux de précision (ACC), le taux de perte (LR), le taux de réussite des attaques (ASR), le temps d'exécution CPU, le rappel et la précision.

Mots-clé:

Apprentissage fédéré (FL) , Apprentissage automatique (ML), Internet des Objets, Algorithme génétique (GA), Facteur d'Anomalie Local (LOF), Entropie, Précision, Taux de Perte (LR), Taux de Réussite d'Attaque (ASR).

Contents

List of Figures

List of Tables

List of Abbreviations

ACC Accuracy Rate

AIoT Artificial Intelligence of Things

AI Artificial Intelligence

ASR Attack Success Rate

BiLSTM Bidirectional Long/Short-Term Memory

CIFAR Canadian Institute For Advanced Research

CNN Convolutional Neural Network

CPU Central Processing Unit

DPA-FL Defending Poisoning Attacks in FL

DP Differential Privacy

EMDG-FL Entropy-Driven Multi-Domain Governance for Federated Learning

EMNIST Extended Modified National Institute of Standards and Technology

ERD-FL Entropy-Driven Robust Defense for Federated Learning

Fashion MNIST Fashion Modified National Institute of Standards and Technology

FATE Federated AI Technology Enabler

FEDML Federated Machine Learning

FEMNIST Federated Extended MNIST

FHE Fully Homomorphic Encryption

FLSecLAB Federated Learning Secure Layered Adaptation and Behavior

FL Federated Learning

FN False Negatives

FP False Positives

FTL Federated Transfer Learning

GAN Generative Adversarial Network

GA Genetic Algorithm

GBDT Gradient Boosted Decision Trees

GDPR General Data Protection Regulation

HFL Horizontal Federated Learning

IForest Isolation Forest

IID Independent and Identically Distributed

IIoT Industrial Internet of Things

IoT Internet of Things

IQC Incoming Quality Control

KNN K-Nearest Neighbours

LLM Large Language Models

LOF Local Outlier Factor

LoRaWAN Long Range Wide Area Network

LPWANs Low-Power Wide-Area Networks

LRD Local Reachability Density

LR Loss Rate

M3D-FL Multi-layer Malicious Model Detection for Federated Learning

MAD Median Absolute Deviation

mGAN-AI Multi-Task Generative Adversarial Network for AI

MKrum Multi-Krum

ML Machine Learning

MNIST Modified National Institute of Standards and Technology

MQTT Message Queuing Telemetry Transport

NAM2D-FL Non-IID Adaptive Malicious Model Detection in Federated Learning

NB-IoT Narrowband IoT

Non-IID Non-Independent and Identically Distributed

OCR Optical Character Recognition

PQC Process Quality Control

RFID Radio Frequency Identification

RL Reinforcement Learning

RNNs Recurrent Neural Networks

SGD Stochastic Gradient Descent

SHE Semi-Homomorphic Encryption

SMPC Secure Multi-Party Computation

TLS Transport Layer Security

TMean Trimmed Mean

TN True Negatives

TP True Positives

UAVs Unmanned Aerial Vehicles

VFL Vertical Federated Learning

Chapter 1

General introduction

Contents

1.1 General introduction

Federated Learning (FL) is an emerging collaborative approach to Machine Learning (ML) that enables multiple participating clients to train a model while keeping their data private. Instead of sharing raw data, each client trains its model using local data and shares only model updates with a central server for aggregation. FL is increasingly utilized in sensitive domains such as healthcare, finance, and the Internet of Things (IoT), where maintaining data privacy and security is imperative. It uses data from many clients to improve model performance, while keeping data confidential. FL offers significant advantages, particularly in scenarios involving sensitive or regulated data, such as medical records or financial transactions. It allows the creation of ML models that learn from the collective knowledge of clients while maintaining their data privacy and security. However, FL faces challenges in ensuring data quality and model consistency across clients, involving issues such as privacy and security concerns, data heterogeneity, communication and resource constraints, model aggregation complexities, and adapting to model drift. Despite its challenges, FL is attracting attention due to its potential benefits in terms of privacy, security, and accuracy. In this context, one specific security threat is poisoning attacks in FL system. These attacks involve adversaries deliberately injecting malicious data into the training sets of participating clients to compromise the integrity of the global model. By modifying training data, attackers can influence the model's parameters or cause it to produce incorrect predictions. The detection and mitigation of poisoning attacks pose significant challenges, as sophisticated attackers might evade detection by subtly injecting malicious data. Addressing the issue of poisoning attacks in FL, requires robust detection and alleviation techniques to safeguard the integrity and performance of the global model.

1.2 Context

In IoT environments, FL system have become increasingly popular. This system involves multiple devices or clients, each training a model on their own data. Crucially, they do this without sharing

their private data, but rather by sending updates from their individually trained models to a central server. This server aggregates these updates to enhance the overall model, combining collaborative learning with data privacy. However, this FL architecture is vulnerable to specific security threats, notably poisoning attacks specifically Label-flipping and Backdoor attacks. In such attacks, malicious clients manipulate their model updates to corrupt the overall learning process. These attacks target the integrity of the FL system, aiming to reduce the accuracy and reliability of the server's global model of FL system in the IoT environment. Our thesis is dedicated to the exploration and development of comprehensive approaches to identify, mitigate, and prevent these types of poisoning attacks by detecting malicious clients in FL system within the intricate and interconnected landscape of the IoT networks. By leveraging advanced ML techniques, meta-heuristic algorithms, and robust outlier detection methods. We aim to enhance the security of FL system, ensuring the integrity and reliability of data and models utilized by the participating clients in the FL system within IoT networks.

1.3 Motivations and issues

In an era where data privacy and security are paramount, this thesis addresses the urgent challenges of deploying FL within the vast, vulnerable networks of the IoT. Driven by the need to protect data from emerging threats without sacrificing efficiency, we dive into the complexities of ensuring robust security in FL system. From combating sophisticated poisoning attacks to navigating the intricacies of IoT environments, our research is a quest for balance; achieving high security without compromising performance. We explore innovative strategies for detecting malicious models, evaluating their effectiveness across diverse datasets and scenarios, as well as the metrics utilized for performance assessment. This work goes beyond tackling current challenges; it's about forging the path for a secure FL system in the IoT networks, where the integration of privacy and performance is clearly outlined, as follows:

- **Privacy preservation** FL's privacy-preserving nature is a key motivation for this thesis. While FL ensures client privacy by not sharing raw data, the presence of malicious clients can compromise this privacy, necessitating enhanced security measures.

- **Security threats** The pressing need to confront security challenges, particularly poisoning attacks like Label-flipping and Backdoor attacks, serves as a primary motivation for this thesis. These attacks specifically target and disrupt the training process, posing a significant threat to the security and reliability of FL system in IoT networks.

- **IoT context** The thesis focuses specifically on FL system used in IoT networks. IoT devices produce huge amounts of data, making them vulnerable to attacks. Securing FL system in IoT networks is uniquely challenging due to the distributed topology and resource constraints. This thesis is motivated by the need to address security issues specific to implementing FL on IoT networks.

- **Malicious model detection** One of the key issues tackled in this thesis is detecting malicious models in FL system. Spotting clients who generate poisoned data or inject Backdoor attacks is challenging because FL is distributed, so there is no direct access to clients' data. Developing effective detection techniques that can accurately identify malicious models is a significant challenge.

- **Performance trade-offs** Enhancing the security of FL system must be balanced with the performance requirements of the system. The proposed solutions need to achieve high detection accuracy while minimizing the computational overhead associated with the detection process. Striking the right balance between security and performance is a critical issue in this thesis.

- **Dataset and scenario variability** In FL system used on the IoT networks, dealing with different datasets is crucial. We are focusing on several types of datasets, including both IID (Independent and Identically Distributed) and Non-IID datasets (such as MNIST and CIFAR),

Fashion-MNIST, and IMDB, as each one introduces unique challenges. Besides these datasets, the variety of scenarios is also important. This includes considering different network designs, how many resources are available, and the specific conditions each system works in. Our goal is to create security solutions that are strong and flexible enough to work well in all these different datasets and scenarios. This is important because the success of our solutions depends on how they handle the variety of challenges in these environments.

- **Evaluation and validation** A key challenge in this thesis is to validate the effectiveness of our proposed approaches. To do this, the evaluation process will involve conducting simulations and benchmarking the performance of our solutions against existing techniques in the field. It is crucial to design appropriate evaluation metrics that ensure the reliability and generalizability of our findings across various scenarios. However, it's important to note that existing approaches in the literature for detecting malicious models in FL system within IoT networks have significant limitations. Many of these methods involve high computational costs, which can be impractical in real-world applications. Additionally, techniques requiring continuous adjustment of thresholds can lead to instability and disrupt the learning process. These methods also have limitations in terms of scalability to large FL system. Overcoming these computational, efficiency, performance and scalability limitations is an integral part of our work. Addressing these drawbacks is a fundamental part of our research, aiming to enhance the efficiency and practicality of malicious model detection in FL system.

1.4 Main contribution of the thesis

The main contributions in this thesis concentrate on the following points:

- We provide a comprehensive literature review of techniques designed for detecting malicious models in FL system, with a specific focus on countering poisoning attacks in IoT networks. FL facilitates collaborative ML model training among devices without sharing raw private data. However, this collaborative nature exposes FL system to poisoning attacks, where malicious clients intentionally inject mislabeled data to compromise the accuracy of the global model. Our review encompasses various techniques to detect and mitigate such threats, including methods based on malicious model detection, analysis of client update diversity, secure aggregation protocols, privacy-preserving, and other innovative approaches. The primary goal is to safeguard the integrity and accuracy of FL system in IoT networks by identifying the presence of malicious contributions. This overview emphasizes addressing security threats by exploring malicious detection techniques for poisoning attacks within FL system while shedding light on ongoing researches aimed at developing more robust defenses. To provide greater insight, we construct a detailed and thorough comparative study analyzing the key differences between various defense mechanisms. this study serves as a valuable reference, highlighting the capabilities and limitations of existing techniques.

- We propose the Federated Learning Secure Layered Adaptation and Behavior (FLSecLAB), a fully customized and self-implemented FL framework to run efficient simulations that is specifically designed to secure FL system against poisoning attacks in IoT networks. FLSecLAB encapsulates our novel defense strategies alongside existing defenses into a comprehensive platform for evaluating FL security in IoT. We designed FLSecLAB to address FL's susceptibility to data poisoning attacks in IoT networks. FLSecLAB is a significant advancement towards secure and efficient FL deployments in IoT, offering extensive customization to assess various defense strategies and their robustness against poisoning attacks, and incorporating a comprehensive architecture to test a wide array of scenarios and simulations, FLSecLAB stands as the pinnacle of our efforts in fortifying FL for IoT networks, providing integrated and holistic measures against poisoning attacks.

- We present our first novel approach, Enhanced Malicious Model Detection based on Genetic Algorithm for Federated Learning (EMDG-FL), designed to enhance the detection of malicious

models in FL system, specifically against Label-flipping attacks. The detection process relies on a classification threshold to distinguish updates from normal and malicious clients. Selecting an optimal threshold is challenging; a too low threshold may miss some poisoning attacks, while a too high threshold may result in false alarms on legitimate models. To address this, our approach utilizes the Genetic Algorithm (GA) [1] to dynamically generate an optimal threshold for malicious model detection. The GA searches for the threshold value that maximizes accuracy, significantly improving the effectiveness of detecting malicious models. GAs simulate natural selection, iteratively evolving candidate solutions based on a "fitness" metric to find robust solutions efficiently. We conduct a simulation using Python and evaluates the performance of EMDG-FL on two datasets: MNIST [2] and Fashion-MNIST [3]. Comparative analysis against three existing approaches in terms of performance metrics, including Accuracy (ACC), Attack Success Rate (ASR), Loss Rate (LR), and CPU aggregation run-time, demonstrates the superior efficacy of the EMDG-FL approach against Label-flipping attacks in FL system.

- We propose a novel scalable defense approach called Entropy-Driven Robust Defense for Federated Learning (ERD-FL) across a wider range of clients number and dataset types. ERD-FL utilizes entropy information and simple adaptive threshold to detect and reject potentially malicious clients in FL system against Label-flipping attacks. This attack disrupt the underlying data distribution by introducing incorrect class labels, causing underlying distortion that can be captured using the statistical metric of entropy. Entropy quantifies the level of disorder, indicating manipulation of the expected label distribution. By calculating entropy on client model updates, we obtain a consistent statistical metric to detect anomalies introduced by the data contamination from Label-flipping attacks. This entropy-based malicious detection avoids the need for costly parameter or threshold adjustments, unlike Reinforcement Learning (RL)-based [4] approach, providing an elegant way to detect deviations that captures evidence of data manipulation activities without extensive parameter tuning. By leveraging entropy, the defense mechanism can effectively identify Label-flipping attacks and protect the integrity of trained models. We performed simulations using the MNIST [2], Fashion-MNIST [3] and IMDB [5] datasets to evaluate the performance of ERD-FL. The results demonstrate that ERD-FL surpasses other previously studied methods in the literature across multiple performance metrics, including ACC, ASR, and LR. This comparison highlights the superior effectiveness of ERD-FL in defending against Label-flipping attacks in FL system.

- We introduce a new approach called Multi-layer Malicious Model Detection for Federated Learning (M3D-FL) that aims to effectively identify malicious models in more complex scenarios with multiple simultaneous attacks and a larger number of clients in FL system. Our method begins with the integration of the Local Outlier Factor (LOF) [6] algorithm to determine a Maliciousness Score (MS) for each model update, assessing the severity of anomalies based on density estimates, this replaces prior Isolation Forest (IForest) [7] algorithm used in previous works and further flags statistical outliers against the ensemble baseline according to their LOF score. Following this initial step, we enhance our detection capabilities by incorporating the Median Absolute Deviation (MAD) [8] algorithm, which provides robust outlier identification. This sequence ensures that even when sophisticated attackers craft model updates to mimic normal behavior and evade detection, our approach maintains strong defenses, complementing the core detection mechanisms in FL system. Not only does this improve security through multi-layered defense, but it also safeguards the detector from missing manipulated models that must evade both core processes and outlier analysis. Simulations using CIFAR10 [9], MNIST [2], and Fashion-MNIST [3] datasets compare with our M3D-FL approach against four others approaches across different clients number. Evaluation of ACC and CPU aggregation run-time demonstrates M3D-FL's efficiency and computational efficiency, showing superiority over existing approaches.

- We showcase a significant contribution to FL with the development of the Non-IID Adaptive Malicious Model Detection (NAM2D-FL) approach, focused on tackling the complex challenges associated with both IID and Non-IID data against Label-flipping and Backdoor attacks. This novel server-client defense mechanism, for identifying and removing malicious models in the FL

system. Motivated by the necessity to overcome the limitations of existing defenses in the literature, our research advances FL's resilience and adaptability under various data conditions. Notably, this marks the first application of server-client defense mechanism in our thesis, innovating by adopting enhancements to the LOF [6] algorithm and incorporating an adaptive threshold mechanism derived from the GA [1], marking a step forward in FL defense strategies. Our evaluations across multiple datasets, including both IID and Non-IID datasets: CIFAR10 [9] and MNIST [2], have verified the NAM2D-FL's superior performance in terms of ACC, ASR, Precision, Recall, and CPU run-time. By enhancing FL to effectively counter poisoning attacks, specifically the Label-flipping and Backdoor , we have established a solid and dependable foundation for its application across various real-world environments. This enhancement secures the performance and reliability of the FL system, thereby significantly boosting trust and confidence in collaborative learning.

1.5 Thesis structure

The thesis is structured as follows:

Chapter 2: A comprehensive literature review of attacks and defense mechanisms in FL system in IoT networks

First, we introduce the fundamentals of the FL system, discussing its system architecture and challenges within the context of IoT networks. Subsequently, we explore general applications of FL in various IoT scenarios, followed by an examination of secure and privacy-preserving applications tailored to IoT environments. Further, we delve into a comprehensive analysis of attacks targeting FL system in IoT networks, categorized into inference attacks and poisoning attacks. To counteract these threats, we survey defense strategies tailored for FL system in IoT networks, encompassing secure aggregation, privacy-preserving techniques and malicious model detection. For each defense technique, we present related works, providing a thorough exploration of existing research. Additionally, we offer a comparative study of these techniques, offering a holistic view of their strengths and limitations. Finally, we introduce FLSecLAB - our self-implemented, fully customized FL framework to secure FL against poisoning attacks in IoT networks.

Chapter 3: Enhanced Malicious Model Detection based on Genetic Algorithm for Federated Learning in IoT networks

We propose EMDG-FL, an innovative approach that significantly enhances the detection of malicious models in FL system, against poisoning attacks particularly Label-flipping attacks. EMDG-FL employs a GA [1] to optimize the threshold for identifying malicious model updates, improving overall detection capabilities. The GA contributes to more accurate detection, making EMDG-FL effective against threat models involving Label-flipping attacks by malicious clients in FL system. The proposed approach consistently outperforms other techniques from the literature.

Chapter 4: Entropy-Driven Robust Defense for Federated Learning IoT networks

We present ERD-FL, an innovative method designed to protect FL system within IoT networks from Label-flipping attacks. Uniquely, this approach is the first ever to apply entropy and an adaptive threshold to effectively identify and combat these attacks. Through extensive evaluations on diverse datasets such as MNIST [2], Fashion-MNIST [3], and IMDB [5] with a different numbers of clients. ERD-FL distinguishes itself where other methods do not, providing scalability and adaptability across different data types and participation sizes against Label-flipping attacks. We have rigorously assessed ERD-FL's performance, the findings confirm its superiority with efficient attack detection, and computational speed, outshining previous defenses.

Chapter 5: Multi-layer Malicious Model Detection for Federated Learning IoT networks

We demonstrate our novel approach M3D-FL, developed to enhance the detection and mitigation of malicious models in FL system within IoT networks, addressing complex scenarios through a multi-layered defense strategy against both of Label-flipping and Backdoor attacks. We introduce a sophisticated multi-layered detection mechanism leveraging LOF [6] and MAD algorithms [8], allowing M3D-FL to efficiently identify and eliminate malicious clients from the FL process. M3D-FL has been validated, demonstrating superior keys performance compared to other existing methods. Showcasing M3D-FL's effectiveness in improving the security and integrity of FL system. It highlights its potential for real-world applications, especially in scenarios involving complex, numerous simultaneous attacks and a larger number of participating clients.

Chapter 6: Non-IID Adaptive Malicious Model Detection in Federated Learning IoT networks

In this chapter, we unveiled the NAM2D-FL server-client approach, developed to fortify the FL system against poisoning attacks within IoT networks dealing with IID and Non-IID data. Recognizing that real-world FL scenarios often entail data that is not uniformly distributed across devices, NAM2D-FL refines the LOF algorithm to improve detection accuracy amidst diverse data distributions, further augmented by a GA-driven optimal threshold mechanism. This groundbreaking advancement in FL addresses the intricate challenges encountered in both IID and Non-IID data contexts, specifically targeting Label-flipping and Backdoor attacks. Furthermore, our approach outperforms other mechanisms by integrating advanced data analysis techniques and adaptive responses to evolving threat landscapes, reinforcing the security and reliability of FL system in various settings.

Chapter 7: Conclusion and perspectives

This chapter concludes my work and outlines several facets of recommended future research works.

Chapter 2

A comprehensive literature review of attacks and defense mechanisms in federated learning system in IoT networks

Contents

2.1 Introduction

FL emerges as a powerful ML paradigm that enables collaborative yet privacy-preserving model training across multiple clients. By keeping data decentralized, FL mitigates the privacy risks associated with traditional centralized learning [10]. The aggregation of model updates from participating clients is a key aspect of FL, allowing a shared model to be trained without directly accessing the raw data. This aspect of FL is especially critical in IoT networks, known for a large network of edge devices that generate and process massive amounts of data [11]. In such environments, FL not only optimizes the use of network resources but also significantly improves the quality of learning outcomes across the IoT networks. However, the integration of FL into IoT networks introduces unique security and privacy challenges. These challenges are further complicated by the potential for various attacks on the FL system, which can undermine the integrity and reliability of the shared model, leading to erroneous outcomes and exposing IoT networks to new vulnerabilities. This chapter aims to underscore the significance of FL within IoT networks by exploring its applications, highlighting its benefits, and addressing the inherent challenges and security concerns. Furthermore, we delve into the landscape of attacks specific to FL in IoT networks, the types of datasets involved, and the defensive strategies that can be employed. Through a comprehensive evaluation and comparative study, this discussion presents FLSecLAB, our fully customizable FL framework for IoT networks, integrating our defenses and additional approaches for extensive testing. It enables experimentation across diverse of scenarios, signifying a pivotal enhancement in FL security. Detailed discussions of our comprehensive defenses, aimed at broadening the testing scope, are presented in the following chapters.

2.2 FL fundamentals in IoT context

FL is an ML technique that allows multiple entities to collaboratively train a model while keeping their data private. This approach is different from traditional centralized ML methods where data is collected and processed in a single location. Instead, FL enables the model to be trained across multiple decentralized devices or servers without sharing the actual data among them [12]. As demonstrated in Figure 2.1, we explore the intricacies of this distributed training methodology. The process of FL involves distributing the model to the devices or clients where the data resides. Each of these clients trains the model on its local data and computes an update to the model [13]. These updates are then sent to a central server where they are combined to improve the global model. This updated model is then sent back to the clients, and the process repeats until the model achieves the desired performance.

FL is particularly beneficial in scenarios where data privacy is paramount or where data cannot be centralized due to regulatory, technical, or ethical reasons. In the context of IoT, where devices often collect sensitive information in environments like healthcare, transportation, smart cities, financial services, and autonomous vehicles, FL's ability to aggregate data from diverse sources while maintaining privacy and security is especially valuable. This makes it applicable across various industries, enhancing IoT applications with its approach to handling data in a decentralized yet secure manner.

However, FL also faces challenges such as dealing with heterogeneous data across clients, ensuring robustness against failures or malicious actors, and managing the computational and communication

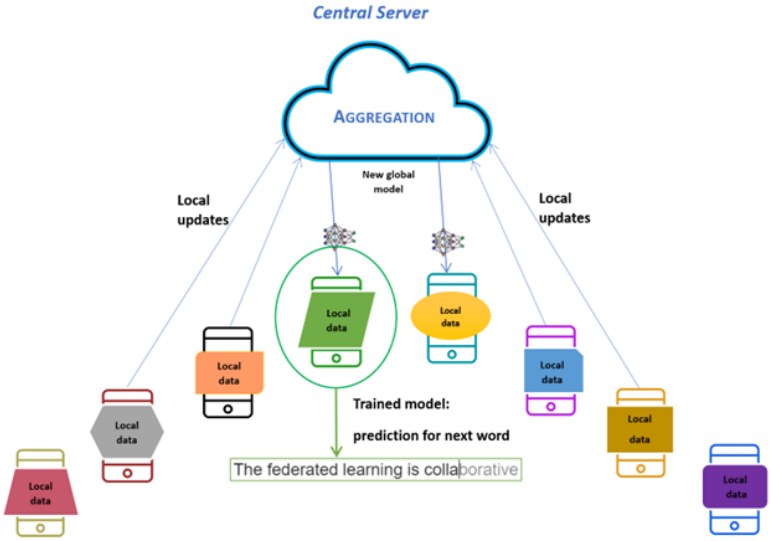


Figure 2.1: An overview of FL system

overhead involved in training models in a decentralized manner. Despite these challenges, FL is considered a promising field in Artificial Intelligence (AI) due to its potential to improve data privacy and security while enabling collaborative model training.

2.3 Aggregation algorithms for FL

FL aggregation algorithms are crucial for merging local updates from multiple devices into a unified global model, adeptly addressing key challenges in FL such as enhancing communication efficiency, managing diverse data characteristics, and scaling to accommodate numerous nodes. The selection of an optimal aggregation algorithm is not trivial; it is central to the efficacy, privacy, and overall performance of the FL system. This decision influences not just the immediate learning outcomes but also the long-term viability and adaptability of the FL system to diverse and evolving data environments.

To facilitate a deeper understanding and enable a rigorous comparison of these aggregation algorithms, a structured overview is essential. In the provided Table 2.1, we meticulously summarize the most prevalent aggregation algorithms within the FL system. This summary encapsulates each algorithm's core definition, highlighting its strengths and limitations, applicable data types, and the underlying techniques or methodologies employed. By dissecting these algorithms into these fundamental components, we offer readers a panoramic yet detailed perspective on how each algorithm operates, its ideal use case scenarios, and the potential challenges one might encounter during implementation. Notably, the table serves as a compass for navigating the landscape of FL aggregation algorithms. For instance, it contrasts the simplicity and communication efficiency of Federated Averaging (FedAvg) [14] with the more sophisticated approach of FedProx [11], which introduces a proximal term to better manage the challenges of statistical heterogeneity. It also delves into the realms of privacy enhancement with SecAgg [15] and the innovative use of knowledge distillation in FedDistill [16]. This comparative analysis not only underscores the diversity of approaches within FL but also illuminates the trade-offs involved in selecting an aggregation algorithm suited to a specific FL application's needs.

Furthermore, this overview is instrumental in bridging the gap between theoretical understanding and practical application, especially regarding the use of aggregation algorithms in FL models. These algorithms are crucial as they directly impact data privacy, efficiency, and scalability. Highlighted by Table 2.1, this guide is a vital tool for anyone working with FL systems, helping them to select the right aggregation algorithms to meet their specific needs. It also encourages further discussion on the

future of FL algorithm research and development, emphasizing the significance of these algorithms in the FL ecosystem. The aim is to foster more research and practical applications, pushing FL forward as a secure, efficient, and privacy-focused technology.

Algorithm	***Definition***	***Data type***	***Pros***	***Cons***	***Used technique***
FedAvg [14]	Averages local updates to form the global model, effectively combining learning from all participating devices	IID / Non-IID	Reduces communication overhead; Simple and effective	May struggle with Non-IID data; Convergence issues in heterogeneous networks	Averaging
FedSGD[14]	Aggregates gradient updates after each local batch or epoch, closely aligning with traditional SGD	IID	Potentially faster convergence	High communication cost; Less efficient in bandwidth-limited scenarios	Gradient Aggregation
FedProx [11]	Adds a proximal term to the local optimization problem to mitigate the effects of statistical heterogeneity	Non-IID	Improves handling of Non-IID data; Reduces client drift	Increased computational complexity; May require tuning of proximal term	Proximal Optimization
SecAgg [15]	Uses cryptographic techniques to ensure that the server can only see aggregated updates, enhancing privacy	IID / Non-IID	Enhances data privacy; Suitable for sensitive applications	Cryptographic operations can be computationally intensive	Secure Aggregation
FedDistill [16]	Utilizes knowledge distillation to aggregate models, focusing on learning a global model that mimics the behavior of local models	IID / Non-IID	Efficient in heterogeneous networks; Reduces communication costs	May require additional hyperparameter tuning; Complexity in implementation	Knowledge Distillation
FedSkip [13]	Allows clients to skip the transmission of updates under certain conditions to save bandwidth and computational resources	IID / Non-IID	Reduces communication costs; Efficient in stable learning phases	Skipping criteria need careful design to avoid loss of critical updates	Conditional Update Skipping

Table 2.1: An overview of FL aggregation algorithms.

2.4 IoT and IoT networks

IoT embodies the concept of a network of interconnected devices that can communicate, gather environmental data, process this information, and share insights to fulfill specific objectives. This concept has led to the automation of various aspects of daily life, exemplified by Smart Homes through automated air conditioning, security surveillance, and lighting systems, among numerous other services and devices. Furthermore, IoT's impact extends beyond domestic convenience, influencing other facets of human life and giving rise to smart education systems, smart healthcare, smart farming, and smart industries. It is projected that by the end of 2025, the IoT sector will have expanded to include 22 billion smart devices [17]. This growth is largely fueled by the integration of IoT systems with advanced sensing and computing capabilities, revolutionizing the connectivity of objects and devices. This evolution is not only aimed at enhancing customer services and applications through the linkage of a diverse array of objects and devices but also at facilitating seamless interactions within the IoT ecosystem.

The integration of AI techniques such as ML and Deep Learning (DL) has been pivotal in endowing IoT systems with intelligence. These technologies are celebrated for their ability to derive insightful knowledge from IoT data, paving the way for the development of smart applications including human activity recognition, vehicular traffic management, and accurate weather forecasting [18]. In the realm of IoT networks, the analysis of IoT data and the delivery of intelligent services are two areas that demand close attention. Leveraging AI techniques for IoT data analytics allows for the creation of sophisticated data processing functions capable of handling the massive volumes of data produced

by a wide range of IoT devices, such as sensors, actuators, smartphones, personal computers, and Radio Frequency Identification (RFID) tags. ML and DL, especially using neural networks, provide advanced computational models with multiple processing layers that enable learning from data at different levels of abstraction. This is essential for extracting meaningful features from various types of IoT data, including images, time series, video, and text, and for managing the vast amount of data streaming from millions of sensors. The adaptability and efficiency of these AI models in processing different data modalities, coupled with their capability to manage large data volumes without complex feature engineering, make them exceptionally suitable for IoT applications [18].

An example of AI's role in IoT data analytics is the deployment of Recurrent Neural Networks (RNNs), which are equipped with interconnected neurons that form a directed graph, thus facilitating the processing of sequential data of variable lengths. This is particularly advantageous for tasks that involve the analysis of sequential data, showcasing the sophisticated capabilities of AI techniques in the IoT domain [19]. Furthermore, FL illustrates how AI can improve scalability and privacy within IoT networks. By decentralizing AI training to multiple devices at the network's edge, FL ensures that data remains secure on local devices. This method not only safeguards privacy but also fosters the development of innovative IoT services and applications, underscoring AI's potential to drive progress in the IoT field.

On the other hand, IoT networks are fundamental in facilitating the connection and interaction of billions of devices globally. They form the cornerstone of a new digital integration era across diverse sectors, including healthcare, smart cities, and industrial automation. These networks are distinguished by their variety, integrating numerous communication technologies, protocols, and standards tailored to meet the specific needs of a wide range of IoT applications.

The selection of connectivity technology is crucial, as it directly affects the functionality and efficiency of IoT solutions. The connectivity options are varied, each serving different requirements: Wi-Fi and Ethernet are suited for environments needing high data rates within small areas; cellular networks such as 4G and 5G provide wider coverage, suitable for more extensive applications; and Low-Power Wide-Area Networks (LPWANs), including Long Range Wide Area Network (LoRaWAN) and Narrowband IoT (NB-IoT), cater to devices requiring prolonged battery life over large distances. This diversity in IoT network infrastructure illustrates the complexity and adaptability needed to support the extensive and varied demands of the interconnected world, highlighting the importance of careful technology selection in the design and implementation of IoT systems.

In the healthcare sector, for example, IoT networks have a significant and positive impact, marking a transition towards more proactive and customized care models. Wearable biosensors and connected medical devices provide continuous monitoring of vital signs, enabling healthcare professionals to monitor patient health in real time. This advancement not only improves the quality of care but also makes it more personalized, supporting treatments specifically designed for the individual needs of each patient. Furthermore, IoT networks are essential in managing chronic diseases, using connected devices to track disease markers and ensure patients adhere to their medication regimes. This approach greatly enhances patient outcomes and reduces the necessity for face-to-face hospital visits, offering a more efficient and effective healthcare system.

However, the widespread adoption of IoT networks, especially in sensitive areas like healthcare, introduces significant privacy and security challenges. Protecting patient data from unauthorized access and ensuring the integrity of health information across IoT networks are critical issues that demand comprehensive security measures. As IoT networks continue to advance, their growing role in supporting smart, interconnected solutions across various domains underscores the importance of maintaining a delicate balance between embracing technological innovation and ensuring rigorous security and privacy safeguards, aiming for a future that is both more connected and secure.

2.5 Importance of FL in IoT networks

Through its pioneering operational system, FL presents numerous significant advantages for the deployment in IoT scenarios. This model showcases the critical benefits and functional efficiencies FL introduces to IoT networks, as outlined in Figure 2.2, as follows:

a. **Enhancement of data privacy:** FL ensures that raw data remains on the local device, eliminating the need to transmit sensitive information to a centralized aggregator. This approach significantly reduces the risk of private data exposure to unauthorized third parties, thereby offering an enhanced level of privacy protection. This is particularly relevant in the context of strict data privacy regulations like the General Data Protection Regulation (GDPR) [20], positioning FL as a preferable choice for developing secure and intelligent IoT environment.

b. **Reduction in network latency:** The absence of the necessity to send IoT data to a central server translates into decreased network latency due to less reliance on data transmission. Consequently, this efficiency not only conserves network resources, such as bandwidth and energy, during the data training phase but also enhances the overall system performance.

c. **Improvement in learning efficacy:** By leveraging computational power and diverse data sets from an extensive network of IoT devices, FL can potentially accelerate the training process and attain superior levels of learning accuracy [21]. Such achievements may not be feasible with centralized AI models limited by data scarcity and computational constraints. Furthermore, the distributed nature of FL augments the scalability of intelligent networks.

d. **Scalability via flexibility:** FL leverages the distributed computing power of numerous IoT devices across different locations, processing data in parallel without burdening any single point in the network. As the capabilities of edge devices increase, leading to larger individual data volumes, the centralization of data processing becomes inefficient, either under utilizing local computing resources or straining the wireless network. FL addresses these challenges by distributing the learning process across a wider array of devices, thereby improving the network's scalability without imposing additional demands on central servers. This approach also eliminates the need for sending large amounts of raw data across the network, further enhancing scalability by reducing communication costs and easing the load on networks with limited bandwidth.

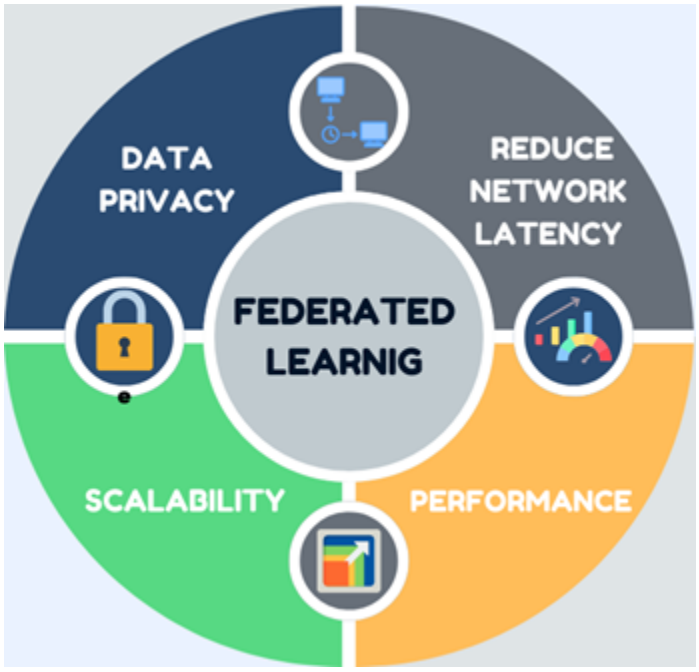


Figure 2.2: The benefits of FL in IoT networks

These distinct benefits have led to the proposition of FL in a spectrum of IoT applications, including but not limited to smart healthcare, intelligent transportation systems, and Unmanned Aerial Vehicles (UAVs). For instance, in the healthcare sector, FL has enabled the confidential modeling of ML

algorithms without necessitating the exchange of patient information among various medical entities [22]. This ensures that hospitals can train AI models locally and share only the model parameters for aggregation, thus fostering a cooperative healthcare ecosystem that accelerates patient care without compromising privacy. In the realm of smart transportation, FL demonstrates its capacity to enhance vehicular services [23] from autonomous navigation to traffic safety forecasting and vehicle recognition—all while ensuring data privacy through collaborative learning between vehicles and roadside units. The emerging successes in FL applications within the IoT field underscore the importance of focusing research efforts on this promising and evolving domain.

2.6 The use of FL in IoT networks

The burgeoning proliferation of IoT devices introduces significant privacy challenges, diverging from traditional, centralized ML system [24]. The practice of incessantly transmitting sensitive information from a multitude of IoT sensors for server-based model training engenders profound data security concerns, potentially stalling broader acceptance. FL, as depicted in Figure 2.3, emerges as a revolutionary approach that facilitates collaborative ML training across IoT networks without necessitating the aggregation of raw data. This paradigm shift allows IoT devices to locally train ML models while ensuring that private data remains within the device, thus obviating the need to share it with a central server or amongst other devices [25]. The adoption of FL offers substantial privacy enhancements alongside benefits such as diminished network expenses, augmented model personalization, increased resilience, and the facilitation of low-latency edge computing [26]. Moreover, FL provides a gateway

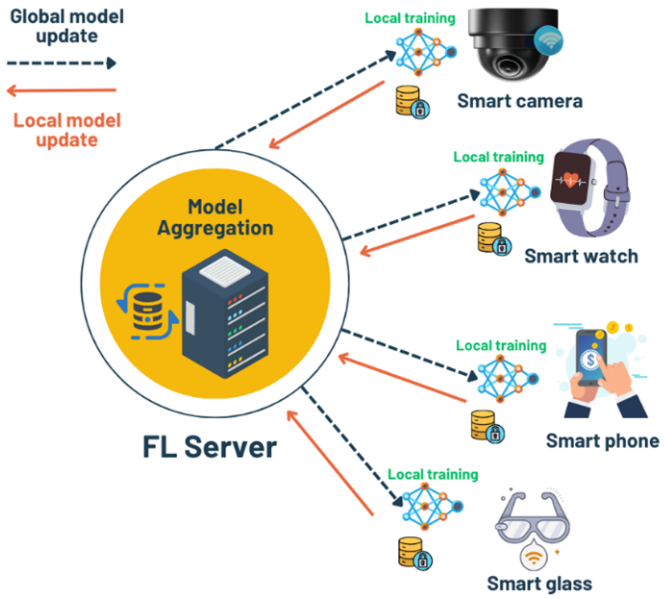


Figure 2.3: A standard FL-IoT networks

to previously untapped innovative prospects by leveraging localized IoT data, which, due to security considerations, was largely unexplored. The availability of on-device data processing enables the IoT ecosystem to exploit context-aware insights, leading to significant advancements in the predictive accuracy, functionality, and robustness of ML models tailored for IoT applications across various sectors including healthcare, transportation, and infrastructure monitoring [25]. Despite these advancements, FL faces research challenges concerning system limitations, adversarial threats, and the statistical variability inherent in highly decentralized frameworks. Nonetheless, FL stands as a pivotal innovation for fostering privacy-preserving AI and unleashing the full potential of IoT data on a grand scale [27], thus introducing significant privacy advancements in the IoT domain.

2.7 Application of FL in IoT networks

Incorporating the previously highlighted advantages, FL has played a transformative role in advancing IoT networks. This section delves into some of the most pivotal applications within these networks, emphasizing their significance. The depiction of these applications, as outlined in Figure 2.4, is as follows:

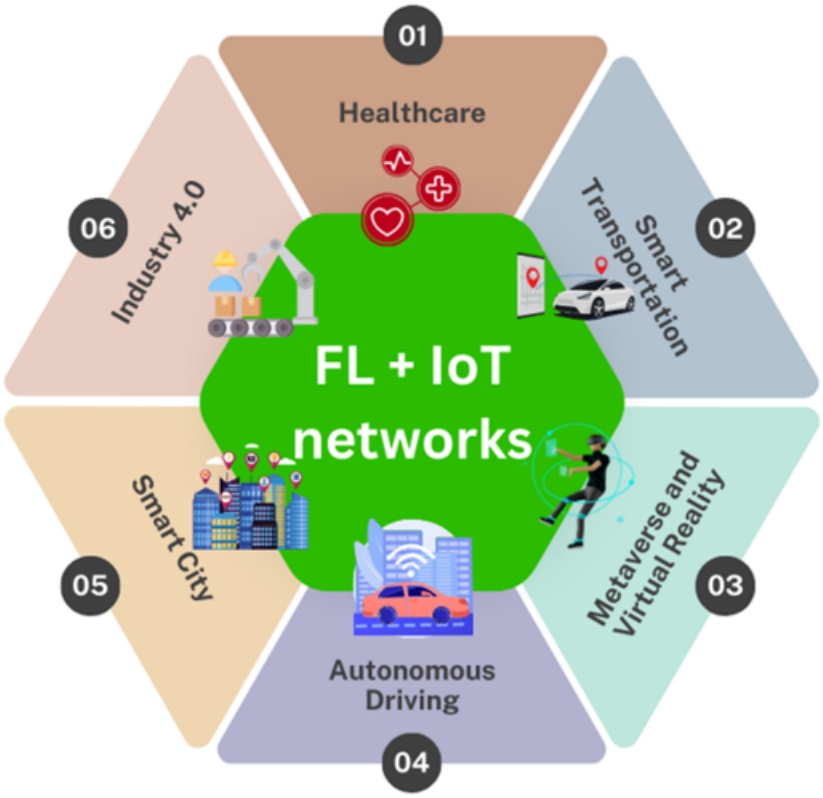


Figure 2.4: Application of FL in IoT networks

a. **FL in healthcare privacy and innovation:**

The increasing use of IoT devices in our daily lives raises significant privacy concerns, especially in healthcare. Wearable devices that monitor vital signs pose heightened sensitivity around personal health data, which is strictly governed by legal and regulatory frameworks. FL is essential for developing advanced ML models across a fragmented and regulated data landscape. It enables large-scale precision medicine by training models across multiple institutions without data consolidation, protecting patient privacy and data integrity [28].

b. **FL's role in industry 4.0:**

The Industrial Internet of Things (IIoT) marks a significant advancement in information technology for manufacturing. Industry 4.0, also known as the fourth industrial revolution, is driven by the increased interconnectivity of IIoT and the availability of real-time data. This enables new levels of insight, control, and visibility across supply chains. Mature Industry 4.0 implementations include Optical Character Recognition (OCR) for labeling, advanced and automated Incoming Quality Control (IQC), and sophisticated Process Quality Control (PQC). However, challenges remain, including the limited data from individual factories for comprehensive model training and the need for privacy in commercially sensitive data. FL offers a potential solution by preserving data privacy while enabling model training [29].

c. **Advancing smart transportation with FL:**

IoT technologies are transforming smart transportation, revolutionizing how we manage and interact with different modes of transport to enhance efficiency and safety. Innovations like intelligent traffic management systems, real-time vehicle tracking, and autonomous vehicles utilize IoT to optimize routing, reduce congestion, and improve overall travel experiences. FL plays a vital role in this sector, enabling collaborative learning from various data sources without compromising individual data privacy. This facilitates smarter and more responsive transportation solutions. [13].

d. **FL's Impact on smart city development:**

Smart cities leverage IoT to optimize operational efficiency and enhance citizen well-being. IoT devices facilitate real-time monitoring of city infrastructure, providing valuable insights into traffic patterns, public safety, healthcare, and more. FL, with its ability to process data locally, addresses privacy concerns and aligns well with smart city applications. FL-based smart grids, for instance, enable learning from power consumption patterns without compromising individual data, fostering an intelligent and interconnected energy network [30].

e. **Enhancing autonomous driving with FL:**

Vehicular IoT and autonomous driving demand robust, real-time communication and adaptability to environmental changes. Centralized data processing raises privacy concerns and may struggle with real-time adjustments due to data transmission volumes and network limitations. FL, implemented in vehicular edge computing, tackles these challenges by reducing data transmission requirements and enabling vehicles to adapt to local changes, significantly enhancing autonomous driving systems.

f. **FL in the spatial computing and Virtual Reality (VR):**

The metaverse, envisioned as the internet's next evolution, offers immersive three dimensional online experiences via computers, VR, and Augmented Reality (AR) devices. A crucial aspect is the digital twin concept, mirroring the real-world's status in virtual environments. FL fosters collaboration between edge computing and servers, enhancing performance and privacy. For example, it enables private training of user-generated data, like eye or motion tracking, on local devices, ensuring a privacy-centric metaverse experience.

2.8 Challenges of FL in IoT networks

In the rapidly evolving landscape of the IoT, the implementation of FL presents a groundbreaking approach to harnessing the power of decentralized data while safeguarding user privacy. This innovative learning paradigm enables a multitude of devices to collaboratively learn a shared prediction model while keeping all the training data on the device, thus mitigating the risks associated with traditional centralized data storage. Despite its promising advantages, the adoption of FL in IoT environments encounters several significant challenges that necessitate meticulous attention and dedicated research efforts.

- **Communication efficiency:** The challenge of communication efficiency is exacerbated by the inherent limitations of IoT devices, which are often constrained by low bandwidth and limited computational resources. The frequent exchange of model updates, a staple in traditional FL processes, becomes a bottleneck, severely impacting the scalability and feasibility of FL deployments in IoT environments [31]. Innovations such as sophisticated data compression algorithms, intelligent client selection mechanisms, and the incorporation of edge computing for localized aggregation aim to mitigate these issues [32, 33]. Yet, the development of ultra-efficient communication protocols that are specifically optimized for the unique landscape of IoT devices remains a critical area for future research. Such protocols must not only minimize data transmission costs but also ensure timely and efficient model updates across a highly distributed network.

- **Statistical heterogeneity:** IoT devices generate data that are inherently diverse in terms of volume, features, and distributions, leading to significant statistical heterogeneity [34, 35]. This variability introduces challenges in training FL models, as it can hinder model convergence and degrade overall performance. Addressing this requires the development of sophisticated model aggregation techniques that can effectively handle unbalanced and Non-IID data across clients. Furthermore, strategies such as employing statistical metrics for quantifying distribution divergence, conducting localized fine-tuning on devices, and applying advanced clustering algorithms are essential for adapting to and overcoming the challenges posed by data heterogeneity.

- **System heterogeneity:** The diversity in the IoT landscape introduces significant challenges due to system heterogeneity, which encompasses a broad spectrum of devices with varying computational capabilities, memory sizes, processing speeds, and software environments [36, 37]. This variability affects not just the performance and efficiency of FL algorithms but also their ability to effectively learn from distributed data sources. The challenge is further compounded by the Non-IID nature of data across these devices. Non-IID data refers to scenarios where the data distribution is not identically and independently distributed across all nodes participating in the FL process. In the context of IoT, this means that the data collected by different devices can vary greatly in terms of features, distribution, and volume, reflecting the diverse environments and contexts in which these devices operate.

- **Privacy and security:** The decentralized nature of FL offers a foundational layer of privacy by design, yet the system remains vulnerable to sophisticated cyber threats [38]. These include inference attacks that can deduce sensitive information from model updates, data poisoning that corrupts the training process, and unauthorized manipulations aimed at compromising the model's integrity. Developing solutions to these challenges requires a multifaceted approach that encompasses advanced cryptographic methods, malicious detection, Secure Multi-Party Computation (SMPC) [39] techniques, and robust privacy-preserving protocols. These measures must be designed to protect against both external breaches and insider threats, ensuring the confidentiality and integrity of FL processes across untrusted networks.

- **Interoperability:** The heterogeneity of IoT networks and the lack of standardized communication protocols present significant interoperability challenges [31]. Achieving seamless interaction between diverse IoT devices and FL system necessitates the development of universal frameworks that can standardize data formats, communication protocols, and model sharing mechanisms. This effort requires broad collaboration across industry and academia, aimed at establishing open standards that facilitate compatibility and ease the integration of FL into existing and future IoT ecosystems.

- **Real-world applicability:** Transitioning from theoretical models to real-world implementation exposes FL system to the complexities of practical IoT infrastructures [40]. Conducting extensive deployments and pilot studies across varied environments is critical for evaluating the effectiveness of FL in real-world settings. These studies not only provide insights into the operational challenges but also highlight the scalability and resilience of FL models under diverse conditions. This knowledge is invaluable for refining FL algorithms and ensuring their robustness and reliability in practical applications.

- **Lack of incentives:** A major obstacle to widespread participation in FL processes is the absence of direct incentives for IoT device owners [41]. Developing compelling incentive structures that motivate individuals and organizations to contribute their computational resources to FL is essential. Innovative approaches, such as leveraging blockchain technology for transparent reward mechanisms and creating dynamic pricing models, can offer tangible benefits to participants, thereby fostering a more collaborative and participatory FL ecosystem.

- **Limited data utilization:** Despite the promise of FL to enable collaborative learning, the full potential of the vast, fragmented data landscape of IoT remains largely untapped [42]. Overcoming this challenge requires the exploration of hybrid models that facilitate controlled,

privacy-preserving data sharing. Such models can bridge the gap between the need for data privacy and the demand for comprehensive analytics, enabling more effective decision-making and unlocking new opportunities for innovation within the IoT domain.

- **Legal and ethical hurdles:** The implementation of FL within IoT networks is fraught with legal and ethical complexities, including issues related to data governance, transparency, accountability, and inherent biases [43]. Navigating these challenges necessitates the establishment of clear industry standards and best practices that align with evolving regulatory frameworks. Such efforts are crucial for ensuring ethical use of FL, fostering public trust, and facilitating its acceptance across various sectors.

2.9 Attacks in FL within IoT networks

The adoption of FL in IoT networks introduces unique security challenges, magnified by the distributed and often resource-constrained nature of IoT devices. This section outlines the primary attacks targeting FL system in IoT contexts, such as model and data poisoning, alongside inference attacks that jeopardize data privacy. These attacks distort the training process and compromise the model's integrity and effectiveness, as illustrated in Figure 2.5. Highlighting these vulnerabilities is essential for the development of advanced security strategies to protect FL system against malicious activities, ensuring the reliability and confidentiality of the learning models and the data they process within IoT environment.

2.9.1 Poisoning attacks

At the core of the FL training mechanism, the reliance on local updates from clients to the central server introduces a pivotal vulnerability due to the server's limited capacity for validating the integrity of these updates. Malicious clients exploit this flaw through poisoning attacks, delivering corrupted updates to the system. These attacks are broadly segmented into targeted, such as Backdoor attacks [44], aimed at compromising the model's function for specific inputs, and untargeted attacks, which diminish the model's overall accuracy [44]. As illustrated in Figure 2.5, two main categories of poisoning attacks are identified: model poisoning and data poisoning, each impacting the FL system in different ways.

a. **Model poisoning attacks:**

Model poisoning attacks represent a direct assault on the integrity of the FL model during its training phase [45]. These attacks are characterized by the manipulation of the model's parameters, a tactic that can dramatically alter the course of the model's learning process. Techniques employed in these attacks range from model exploration, which seeks to uncover vulnerabilities in the model's architecture, to model inversion, an approach that attempts to reverse-engineer model outputs to disrupt the training process. Such attacks not only compromise the model's accuracy but also pose significant challenges to maintaining the confidentiality and integrity of the model's training data.

b. **Data poisoning attacks:**

Data poisoning attacks target the very foundation of the FL model's learning process: the training data [46]. By tampering with this data, adversaries can significantly impact the model's performance. This category encompasses two primary attack vectors: data modification and data injection. Data modification attacks, including Label-flipping [47], involve altering correct labels to incorrect ones, thereby confusing the model during the training phase. On the other hand, data injection attacks, such as Backdoor strategies [48], introduce manipulated data into the training set, aiming to create vulnerabilities that can be exploited once the model is deployed. We delineate the characteristics and mechanisms of these attacks as follows:

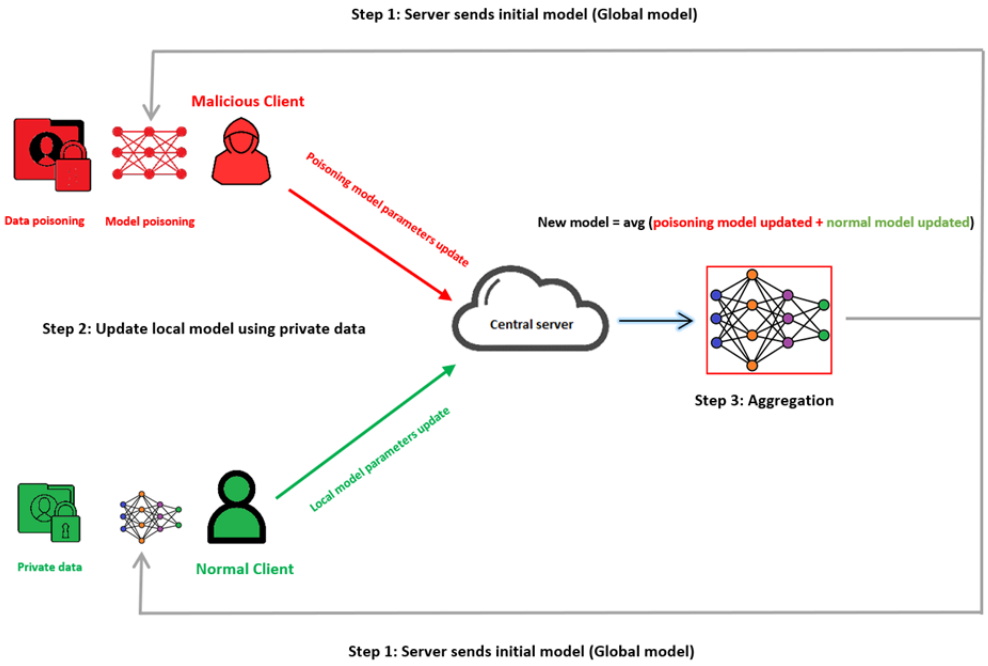


Figure 2.5: The data and model poisoning attacks in FL system

- **Label-flipping attack:**

  Despite their straightforward approach, Label-flipping attacks pose a significant threat to model integrity. Such attacks manipulate the model's learning process by simply altering the labels of selected data points [47]. This method does not necessitate a deep understanding of the model's inner workings, making it a viable option for a broad spectrum of attackers. The act of mislabeling data forces the model to establish incorrect patterns and associations, significantly undermining its reliability and accuracy. This attack can also be explained in a different manner; In these attacks, malicious clients change the labels in the training data to degrade model performance. The number of attacks is chosen randomly. Let the normal training dataset contain N sample-label pairs: $\{(a_1, b_1), (a_2, b_2), ..., (a_N, b_N)\}$. The attacker flips some labels to construct a poisoned dataset: $\{(a_1, b'_1), (a_2, b'_2), ..., (a_N, b'_N)\}$. Where $b'i$ not equal bi for some subset of labels that are flipped. A model trained on the poisoned data performs worse than one trained on the original clean labels. The normal label set is: $B = \{b_1, b_2, ..., b_N\}$ and the poisoned label set is: $B' = \{b'_1, b'_2, ..., b'_N\}$.

- **Backdoor attack:**

  Backdoor attacks in FL represent a form of deliberate sabotage executed through the manipulation of training data. These attacks introduce hidden triggers or subtle data alterations designed to corrupt the model's behavior [48]. The effectiveness of such attacks hinges on two critical aspects: the timing, where the model is nearly fully trained and thus more susceptible to manipulation, and the attacker's knowledge of the FL system's intricacies. This includes an understanding of how the system operates and identifying the minimum number of compromised participants required to successfully embed the corruption within the model.

2.9.2 Inference attacks

Inference attacks are a major security concern for FL, as they directly threaten the privacy of training data. These attacks aim to uncover sensitive information embedded in the data used to train the FL model. By strategically querying the model, attackers engage in two primary types of inference

attacks: reconstruction and membership. Reconstruction attacks try to generate copies of individual data points, potentially revealing private details, while membership attacks determine if a specific piece of data was used in training the model. Both types of attacks risk disclosing sensitive information, challenging the privacy and security measures in place within the FL system. We clearly define the workings of these attacks as follows:

- **Reconstruction attack:**

  Reconstruction attacks, such as the multi-task Generative Adversarial Network (mGAN-AI) proposed by [48], showcase the potential to generate data samples that closely mimic real training data. These attacks operate by exploiting the shared model updates, either passively or actively, to siphon off sensitive information about the model's training data. While remarkably sophisticated, the effectiveness of these attacks can be mitigated using encrypted communications between the clients and the server, highlighting the importance of security measures in preserving the integrity of FL system.

- **Membership attack:**

  Membership attacks are particularly invasive, as they allow adversaries to infer whether a given data sample was part of the model's training set. This type of attack exploits the output of a well-trained model to make determinations about the data's provenance, thereby breaching the privacy of the data subjects. Techniques such as the one proposed by [49] demonstrate the nuanced approach required to successfully execute membership attacks, underscoring the sophisticated nature of threats facing FL environments.

As FL becomes increasingly integrated into IoT networks, recognizing the associated security risks is essential. This approach, which aims to improve data privacy and processing efficiency by allowing devices to collaboratively learn without sharing their data, introduces distinctive challenges. Prior to exploring the specific vulnerabilities and attacks targeting FL within IoT networks, it's important to appreciate the context of FL's application and the security implications of its widespread use. This understanding is critical for addressing the security needs of FL systems in IoT settings, leading to the crucial discussion on developing effective defenses against such threats.

2.10 FL defenses in FL within IoT networks

In this section, we delve into defense mechanisms tailored to protect IoT networks against poisoning attacks, focusing on approaches like secure aggregation, data encryption, model validation, malicious model detection, and privacy-preserving techniques. These measures are critical for enhancing the security of the FL system. Given the significance of combating poisoning attacks within these systems, we have dedicated a portion of our thesis to this issue. As a result, we include Table 2.2, which directly compares various defense mechanisms against poisoning attacks in the FL system. This emphasis reflects our thesis's focus on identifying and evaluating effective defenses against such attacks, demonstrating the necessity of robust security measures in securing IoT networks.

2.10.1 Secure aggregation

There are different secure aggregation algorithms, such as [50, 51], which can guarantee privacy protection when gathering model updates from clients in the FL process. These algorithms also tackle the problem of client dropout during iterations. Nevertheless, secure aggregation algorithms have limitations, including the server's visibility of pre-iteration data aggregates and inefficiency for sparse data aggregation. For example, after one round of local computations and sharing of encrypted results, the server may see "Sum = 153, Count = 5" representing the aggregated sum and count of data from 5 users, without seeing their raw data. The authors of [52] introduced a secure aggregation algorithm named FASTSECAGG. This algorithm uses a technique called sketching to lower the amount of data that needs to be transmitted, thereby reducing communication costs. Additionally, it employs a method based on Shamir's secret sharing, referred to as fastshare, to enhance scalability. The benefits

of FASTSECAGG include reduced computational and communication expenses, the ability to handle participant dropouts, and security measures to prevent collusion among servers.

Instead of averaging clients' updates, researchers have explored a new approach called Byzantine robust aggregation [53]. This technique involves removing outliers from local models' data before aggregation, using methods like Median [54], Trimmed Mean (TMean) [54], Multi-Krum (MKrum) [55]. However, these techniques rely on certain assumptions, such as IID data distribution and smooth loss function, to provide theoretical guarantees. However, these rules become less effective when dealing with Non-IID data because they ignore important information during update aggregation. Also, the error in their estimates increases as the model size grows.

The authors of [56] proposed a Byzantine aggregation algorithm that involves the server collecting a small clean training dataset and maintaining a server model similar to clients' local models. The server assigns trust scores to clients based on cosine similarity between their local models and the server model, known as bootstrap trust. These trust scores are used during aggregation. FLtrust, as described by Cao et al., bounds the difference between the global model learned by FLtrust and the optimal global model under no attack, assuming certain conditions. Furthermore, FLtrust is robust against adaptive attacks.

2.10.2 Privacy-preserving techniques

Several studies have been concentrating on enhancing the confidentiality aspects of FL through the utilization of various methodologies including SMPC [39], HE technique as referenced in [57], and Differential Privacy (DP) as discussed by [58]. The upcoming sections will provide a detailed exploration of these techniques.

2.10.2.1 SMPC

SMPC [39], an intricate branch of cryptography [59], focuses on developing new techniques that allow multiple parties to collaboratively compute a function using their inputs while safeguarding the privacy of these inputs. This process often involves distributing the overall computation into smaller, manageable tasks, with each task executed independently by a party, before aggregating these individual results. Within FL, each client employs SMPC methods, such as secret sharing outlined by [15], to obscure their local model updates (including gradients or weights) by converting them into shared secrets. Following this, the aggregation process can utilize HE, as detailed in the next section, or protocols based on secret. According to [60], SMPC provides strong privacy guarantees and reliable defense against adversarial attacks in FL. Nonetheless, the implementation of SMPC comes with its set of difficulties:

- **Computational demand:** The use of SMPC [39] techniques frequently requires significant computational efforts, which can lead to prolonged training duration and heightened resource usage.
- **Communication load:** The adoption of secure protocols typically involves a higher frequency of communication and larger message sizes, which might result in increased bandwidth demands and the risk of communication bottlenecks.
- **Scalability:** Expanding FL system to support numerous nodes presents difficulties, given that the efficiency of SMPC protocols tends to decrease with the addition of more clients.

2.10.2.2 HE

HE represents a distinct category of encryption techniques, crafted to enable the performance of mathematical operations on data while it remains encrypted [61]. Fully Homomorphic Encryption (FHE), as introduced by [62], is a variant of HE that supports computations on ciphertexts of any type. On the other hand, Semi-Homomorphic Encryption (SHE), detailed by [63], permits computations on ciphertexts but is limited to specific operations like addition and multiplication. While HE offers

promising prospects for safeguarding the privacy of model training and evaluation data in FL, it is not without drawbacks:

- HE is computationally intensive, resulting in longer training periods and greater consumption of resources on individual nodes.
- Certain HE techniques are limited to fundamental mathematical operations, such as addition and multiplication, which may not meet the requirements for advanced ML models or optimization techniques.
- In comparison to conventional encryption techniques, HE generates more voluminous ciphertexts, increasing communication overheads and potentially intensifying bandwidth limitations.

2.10.2.3 DP

DP, as defined by [58], is a powerful approach for protecting data privacy during the analysis or sharing of large and sensitive datasets. Its fundamental strategy involves adding a random noise function to the original data, which obscures specific details while preserving the overall utility of the aggregated data. Within the FL context, where clients collectively train a model without centralizing their data, DP is applied by introducing random noise to the model's parameters. This can involve employing mechanisms such as the Laplace technique [64] or the Gaussian method [65]. The parameters, now embedded with noise, are subsequently shared with the aggregator. Incorporating DP into FL significantly enhances the privacy safeguards for devices participating in the network and their data. Nevertheless, integrating DP into FL introduces certain obstacles:

- **Privacy dilemma:** Adding noise to safeguard privacy can lead to slower convergence and increase the variability in model updates. Finding the right balance between privacy protection and the practical usefulness of the data becomes a pivotal concern in the development of privacy-focused FL system.
- **Parameter optimization:** Fine-tuning the parameters for DP, such as noise levels and privacy budgets, poses a substantial challenge. This process demands a thoughtful analysis of the specific FL scenario, the sensitivity of the data involved, and the degree of privacy required.

2.10.3 Malicious model detection

Poisoning attacks in FL can harm the accuracy and security of the global model. Hence, it is crucial to devise robust defense mechanisms to detect and mitigate the impact of malicious clients on the FL system. Several strategies have been proposed in the literature to enhance the security of FL against poisoning attacks. Some approaches involve detecting and filtering malicious updates by carefully analyzing the updates received from clients. It becomes possible to identify and exclude those that contain malicious or poisoned information, which helps to maintain the integrity of the global model. Secure aggregation techniques are another defense mechanism used to enhance FL security. These techniques ensure the aggregation process is resilient to attacks and prevent malicious clients from manipulating the results. By employing secure aggregation, the influence of poisoned models is mitigated, and the overall accuracy of the global model is preserved. Improving the accountability of clients is also crucial in defending against poisoning attacks. Tracking and monitoring clients' behavior makes identifying suspicious or malicious activities easier. Our research focuses on enhancing the defense against Backdoor and Label-flipping attacks. By developing effective mechanisms to detect and counter such attacks, we aim to safeguard the integrity and reliability of FL system. Several recent works in the literature have addressed the challenge of poisoning attacks in FL. These works propose various techniques and algorithms that aim to detect, prevent, or mitigate the impact of malicious clients. By building upon the knowledge and advancements in this field, we contribute to the collective efforts in strengthening the defense against poisoning attacks in FL.

In [66], the authors investigate the characteristics of Backdoor attacks on a large-capacity model in Non-IID federated settings. The study reveals that redundant neurons can be exploited to improve

backdoor persistence and avoid the invalidation of backdoor features after aggregation. They propose a new defense scheme with similarity measurement for convergence-round attacks. The proposed scheme is designed based on the characteristics and vulnerabilities of Backdoor attacks, allowing for targeted defense in FL with lower time complexity and better defense effectiveness. Additionally, they design a defense scheme with backdoor neuron activation to counter early-round attacks. However, it is important to note that their proposed defense schemes may not be effective against more sophisticated Backdoor attacks.

In [67], authors propose a hybrid learning-based method for detecting malicious parameter updates in FL and provide empirical evidence of its effectiveness against a Label-flipping attack on three different image classification tasks. The proposed method comprises two strategies: one involves learning the association between parameters and client data, while the other measures cosine similarity among parameter updates from individual clients. The study reveals that their approach surpasses the performance of a spectral anomaly detection method proposed in prior research.

In [68], the authors introduce a novel approach to balancing privacy, accuracy, and fairness in ML models. It reviews existing fairness protection methods and related literature on fairness and privacy collaborative learning in the FL context. The paper also introduces key technologies and the fairness quantitative mechanism underpinning the study and details the method for applying the DP algorithm in FL. Finally, the paper discusses and summarises the experiments performed to evaluate the performances of DP.

In [69], the authors proposed a defense against the Sybil-based poisoning attacks. Their method is called FoolsGold. They use a variety of client updates during distributed learning to detect these attacks. Their method uses cosine similarity and pardon mechanisms to identify malicious clients from normal ones. Unlike previous approaches in the literature, FoolsGold doesn't limit the number of potential attackers, doesn't rely on outside information, and assumes less about clients and their data. The limitations of FoolsGold in Non-IID settings include its reduced efficacy against individual attackers due to reliance on update dissimilarity, which may falter with varied or overlapping data distributions. Additionally, while computational costs are generally manageable, they increase with the number of clients, potentially impacting scalability in larger FL environments.

In [70], the authors introduced a novel defense technique that identifies abnormal updates in both IID and Non-IID scenarios. Their method involves cross-validation among clients on the client side, where each update is assessed using the local data of other clients. During aggregation, the server adjusts the update weights based on the evaluation outcomes. To address the imbalanced data distribution in the Non-IID scenario, a dynamic client allocation mechanism assigns detection tasks to the most suitable clients. Additionally, the approach incorporates DP [40] measures to safeguard client-level privacy without compromising detection performance.

In [71], authors present a novel method to enhance FL tasks' dependability in mobile networks. They accomplish this by introducing the concept of reputation as a measurement and suggesting a client selection scheme based on this metric. The scheme employs a consortium blockchain as a decentralized approach to efficiently and securely manage clients' reputations, thereby preventing repudiations and tampering. Through numerical analysis, the authors demonstrate the effectiveness of their proposed approach in improving the reliability of FL tasks in mobile networks.

In [72], the authors proved that data poisoning attacks can significantly decline classification accuracy and Recall, even if only a few of the clients are malicious. Furthermore, they indicate that these attacks can be focused and seriously affect targeted classes. They also investigate how the timing of the attack and the number of malicious clients influence the attack's longevity. They propose a defensive tactic that can help detect and prevent data poisoning attacks by extracting the unique characteristics of malicious clients in FL and identifying them using the most important Principal Component Analysis (PCA).

In [73], the authors proposed a defense mechanism that uses generative adversarial networks to generate auditing data during the training process and identifies and removes adversaries by auditing their model accuracy. Experiments on MNIST [2] and Fashion-MNIST [3] datasets show that FL is vulnerable to poisoning attacks and their defense method can detect and mitigate such attacks.

In [74], authors proposed a new mechanism for detecting malicious models in FL-enabled Artificial Intelligence of Things (AIoT), called D2MIF, which uses an IF algorithm. D2MIF computes an MS

for each model uploaded by clients using the IF and filters out any models with scores higher than a dynamically adjusted threshold, determined through RL. The proposed mechanism is validated using MNIST[2] and Fashion-MNIST[3] datasets, and the results demonstrate that D2MIF efficiently identifies malicious models. and improves the global model's accuracy in FL-enabled AIoT.

In [75], the authors introduced DifFense, a defense approach aimed at safeguarding FL systems from Backdoor attacks. This approach employs differential testing and a two-step MAD [8] outlier detection approach to identify attacks without prior knowledge of attack scenarios or access to local model parameters. Experimental findings illustrate that this method effectively thwarts various potential attackers while preserving a convergence level similar to that of the global model achieved using FedAvg [14]. Furthermore, the DifFense surpasses previous defense approaches, such as MKrum and coordinate-wise median aggregation. The detection method utilized significantly reduces the average backdoor accuracy of the global model to under 4% and boasts a zero false negative rate, validating its efficacy and versatility.

In [76], the authors propose a two-phase defense mechanism called Defending Poisoning Attacks in FL (DPA-FL) for intrusion detection systems. It addresses the problem of poisoning attacks in FL-based IDS (FL-IDS) where attackers generate malicious local models to pollute the global model. The first phase compares weights between clients to identify potential attackers, while the second phase tests the aggregated model's accuracy in detecting attackers. Experimental results show that DPA-FL achieves high accuracy in defending against poisoning attacks and improves the F1-score under Backdoor attacks. The proposed defense mechanism enhances classification accuracy and detection efficiency in FL-IDS.

In [77], the authors discuss using FL and edge computing for network intrusion detection in the IoT. They focus on Label-flipping attacks and propose a lightweight detection mechanism to mitigate their impact. The mechanism filters out anomalous clients by evaluating their local model's loss and training dataset size. clients with similar characteristics are identified using clustering algorithms. Experimental results demonstrate the effectiveness of the proposed method in defending against Label-flipping attacks and improving the accuracy of intrusion detection models in IoT networks.

In [78], the authors propose a novel defense scheme for protecting FL models from Backdoor attacks. The proposed approach, called ADFL, utilizes adversarial distillation. It generates fake samples with backdoor features using a Generative Adversarial Network (GAN) on the server side and relabels them to create a distillation dataset. Knowledge distillation is then performed using the clean model as a teacher and the global model as a student, which helps revise the global model and eliminate the influence of backdoored neurons. This effectively defends against Backdoor attacks while maintaining model performance. Experimental results demonstrate that ADFL reduces the success rates of Backdoor attacks by 95% while keeping the main task accuracy above 90%. The approach shows promise in enhancing the security of FL against Backdoor attacks. We summarize various defenses against poisoning attacks in the FL system in Table 2.2, offering a comparative overview that highlights their effectiveness and application within our analysis.

Overall, several techniques are proposed in the literature for detecting poisoning attacks in FL. However, we notice that these works still have limitations and potential drawbacks that must be carefully considered and addressed in future research. Firstly, in the D2MIF algorithm [74], the continuous adjustment of a threshold value can lead to instability and slow learning. In addition, using RL [4] is computationally expensive and time-consuming to train, as RL[4] algorithms require many interactions with the environment to learn an optimal policy, which can be impractical for real-world problems with complex environments or time constraints. We also notice in the D2MIF [74] approach that the effectiveness of the IF algorithm is influenced by the malicious clients' numbers, data quality, and tree construction parameters. Incorporating the IF algorithm in the D2MIF approach [74] can increase computational expenses and potentially affect detection speed and system efficiency. Secondly, in the DifFense [75], [66], and [67] approaches, the authors focus solely on the Backdoor attack. In DifFense [75], they assume that the number of adversaries in the FL system is known and remains constant. In practice, it may be challenging to determine the number of adversaries, and the proposed defense mechanism may not be effective if the number of adversaries changes over time. Moreover, the defense strategy assumes normal clients are the majority, which may not always be true. Thirdly, the effectiveness of DPA-FL algorithm [76] may be reduced when there is a small

Work	***Main Contribution***	***Performance Metrics***	***Dataset***	***Deficiency***	***Techniques and Environment***
[76]	Two-phase defense mechanism called DPA-FL, defense against Label-flipping and Backdoor attacks	F1-Score	CICIDS2017[79]	Limited poisoned data reduces DPA-FL's effectiveness, lowering accuracy and raising false negatives. Low poisoning ratios hinder Backdoor attack detection, increasing false negatives	Convolutional Neural Network (CNN)[80] models in FL with 12 clients: 2 CNN layers, fully connected layer, 3x3 kernels, 2x2 max pooling, stability threshold α 3-5
[74]	D2MIF utilize the IF algorithm and RL, defense against Backdoor attacks	Accuracy, Recall, F1-score	MNIST[2], Fashion-MNIST[3]	More malicious clients decrease effectiveness, which relies on data quality and parameters. Increased IF in D2MIF raises computational costs, possibly slowing detection and reducing efficiency.	FL implemented with Tensorflow, scikit-Learn, Windows 10 Server, Intel Core i5-7500 CPU, 8GB RAM
[75]	DifFense approach, employing differential testing and two-step MAD [8] outlier detection, defense against Backdoor attacks	False Negative Rate, accuracy	Fashion-MNIST[3], CIFAR10[9]	Relies on an assumption that most agents are benign	Uses custom CNN the datasets, 50 clients, 20% participation, 1 adversary (default)
[69]	FoolsGold employs an adaptive learning rate based on inter-client contribution similarity, defense against Sybil-based poisoning attacks	Attack success rate, accuracy	MNIST, VGGFace2[81], KDDCup[82], Amazon[83]	Check similarity of updates from clients. Ineffective against solo attackers. Less effective if attackers know design	FL prototype in Python with scikit-learn for cosine similarity, using Non-IID data for training and FoolsGold [69] for shared classifier aggregation
[84]	FedGuard Introduces a secure FL approach that doesn't require pre-training or auxiliary datasets. Utilizes CVAE (Conditional Variational AutoEncoder) to generate validation data for auditing	Reconstruction Error, accuracy	MNIST[2]	Discusses potential overhead related to the size of the autoencoder and the balance between performance and security	Experiments conducted on GRID'5000[85] with advanced computing resources, demonstrating the practical deployment of the approach.
[86]	ShieldFL protects with HE encrypted models from poisoning using secure cosine similarity and Byzantine-tolerant aggregation	Accuracy	MNIST[2], KDD-Cup99[82], Amazon[83]	HE adds privacy-preserving computational overhead	Privacy-Preserving FL with HE defense
[87]	CONTRA utilize Reputation-based aggregation to identify suspicious clients based on cosine similarity between model updates	Attack success rate, accuracy	MNIST[2], CIFAR-10[9], Loan[88]	Less effective with only one attacker; Susceptible to intelligent perturbation attacks	FL prototype, built in Python/PyTorch, uses Stochastic Gradient Descent (SGD)[89] and Dirichlet [90] distribution for data allocation among 100 clients
[91]	PEFL offers a private, secure FL system using HE for poison detection and robust adaptive aggregation.	Attack success rate, accuracy	MNIST[2], CIFAR-10[3]	Computational overhead from encryption and aggregation techniques	Conceptual approach for secure, robust FL to mitigate poisoning attacks, without specific implementation details.
[92]	FAA-DL introduces an approach combining federated analytics and functional encryption for defending against local model poisoning in distributed learning.	Accuracy	MNIST[2], Fashion-MNIST[3]	Potential computational and communication overhead due to anomaly detection and functional encryption processes.	FL prototype, anomaly detection, and verification modules in PyTorch on MacOS with Intel Core i7 and 16GB RAM for 20 clients.

Table 2.2: Comparative table of related works for defense mechanisms opposite poisoning attacks in FL system

amount of poisoned data, leading to potential false negatives. Fourthly, the FoolsGold [69] needs to verify the similarity of updates from different clients and is less effective against attacks by a single person.

The shortcomings of previous studies have guided us in our proposed enhanced solutions with respect to the requirements of the optimal malicious model detection technique. In this thesis, we aim to mitigate poisoning attacks in FL systems deployed in IoT networks. We address the limitations of prior research by introducing and evaluating novel defense strategies to enhance FL system security. By examining these strategies, we seek to ensure the integrity and security of FL system against advanced threats. In the following section, we present our methodology for evaluating these defense mechanisms, emphasizing their effectiveness and implementation in practical IoT scenarios.

2.11 Evaluation of FL defenses in IoT networks

Building on the discussion of defenses within IoT networks, it's crucial to assess the effectiveness of FL defense mechanisms through a variety of metrics. These metrics, grouped into distinct categories, illuminate various dimensions of FL defense efficacy. Key categories include the efficiency of communication, evaluation of participant contributions, performance of the defense model, and the thoroughness of the defense process assessment. Within these realms, metrics such as ACC, LR, ASR, Recall, Precision, F1-Score, aggregation, and CPU run-time, provide a detailed view of FL defense capabilities. The following section offers an integrated summary and outlines the equations typically employed for calculating each of these metrics, thereby connecting the theoretical underpinnings of defenses in IoT networks with practical evaluation tools.

1. **ACC:** This metric quantifies how accurately the defense model predicts true labels, serving as a primary indicator of defense performance. Introducing the ACC metric through equation (2.11.1), it's essential to recognize that a high ACC signals a model's successful differentiation between honest and malicious behaviors:

$$ACC = (TP + TN)/(TP + TN + FP + FN) \tag{2.11.1}$$

   The notation in equation (2.11.1) is explained as follows: True Positives (TP) represent malicious models correctly identified, False Positives (FP) denote honest models misclassified as malicious, True Negatives (TN) indicate honest models correctly identified, and False Negatives (FN) represent malicious models misclassified as honest.

2. **Recall (Sensitivity):** The Recall metric, shown in equation (2.11.2), is key for making sure the model finds all key instances. It's crucial for avoiding missed detections in critical scenarios.

$$Recall = TP/(TP + FN) \tag{2.11.2}$$

3. **Precision:** This evaluates the proportion of true positive predictions in all positive predictions made by the defense model, important for scenarios where the cost of a false positive is high. The Precision metric can be defined as in equation (2.11.3):

$$Precision = TP/(TP + FP) \tag{2.11.3}$$

4. **F1-Score:** The F1-Score combines Precision and Recall into one metric by taking their harmonic mean. It helps balance the two, especially important when both identifying all relevant cases (Recall) and being correct in those identifications (Precision) are key. The F1-Score is defined by equation (2.11.4):

$$F1score = TP/[TP + \frac{1}{2} * (FP + FN)] \tag{2.11.4}$$

5. **LR:** Typically, inversely related to defense model performance, indicating the discrepancy between predictions, and expected outcomes. Lower loss indicates better defense performance.

The LR can be calculated as in equation (2.11.5) by comparing the global model's accuracy before and after training with a new set of clients.

$$LR = (ACC\_initial - ACC\_final)/P \tag{2.11.5}$$

In equation (2.11.5), the notation can be clarified as follows: ACC_inital denotes the accuracy of the global model before incorporating the new set of clients, while ACC_final represents the accuracy of the global model after including these clients in the training process. The variable P stands for the number of participants or client devices actively participating in the training procedure.

6. **ASR:** Measures how well the defense model can resist or fall victim to attacks. It's crucial for checking how tough and secure the model is against threats in IoT networks. You can calculate ASR using a specific formula, which helps determine the model's ability to defend against attacks. The ASR metric can be defined as in equation (2.11.6):

$$ASR = TP/(TP + FN) \tag{2.11.6}$$

7. **CPU aggregation run-time:** Measures the computational effort required for aggregating updates from various IoT devices, key for assessing the computational efficiency and scalability of FL defense in IoT networks. The CPU aggregation run-time is typically measured in seconds or milliseconds, depending on the aggregation process's complexity and computational resources.

    - **Impact of defense mechanisms on CPU aggregation run-time:** Calculating the CPU aggregation run-time in FL helps assess how defense mechanisms, while not altering the aggregation phase directly, affect it indirectly. These mechanisms might introduce preprocessing or postprocessing overheads, influencing data/model readiness and timing for aggregation. They can also alter system dynamics by modifying data or model characteristics, potentially affecting input volumes and aggregation speed. Additionally, defense mechanisms consume computational resources, which might lead to resource contention and influence overall system performance, including aggregation. Analyzing the CPU aggregation runtime in this context is vital for evaluating defense strategies comprehensively, considering their impacts on both system efficiency and security.

8. **CPU run-time:** Within the scope of our analysis, denotes the amount of time the Central Processing Unit (CPU) is engaged in executing the operations of a specific method during one FL iteration. This metric is crucial for evaluating the efficiency of our approach in an FL environment, as it allows us to compare the computational time required by our method with that of other methods during each iteration. Specifically, the CPU runtime is quantified in seconds for each method and provides key insights into the processing efficiency and speed of the method when executed on the server side during a single FL iteration.

These metrics provide a thorough foundation for assessing the effectiveness, efficiency, and security of FL defense models in IoT networks. They address both the performance of the defense model on its tasks and the operational aspects of the FL defense system.

2.12 Datasets in FL within IoT networks

As previously mentioned, FL is designed to store private data locally. Therefore, most FL researchers test and verify their algorithms using benchmark datasets. Even if a real FL research use case is found, the datasets cannot be shared due to the confidentiality of the participating clients (e.g. patients, and financial clients). This is why many researchers and industry professionals create benchmark datasets to experiment with FL. This is supported by state-of-the-art studies. Highlighted in Table 2.3 is a comparative analysis of their features, while Figure 2.6 visually represents these datasets, offering a clear perspective on the tools available to FL researchers. This section aims to illustrate the contribution of each dataset in advancing FL research.

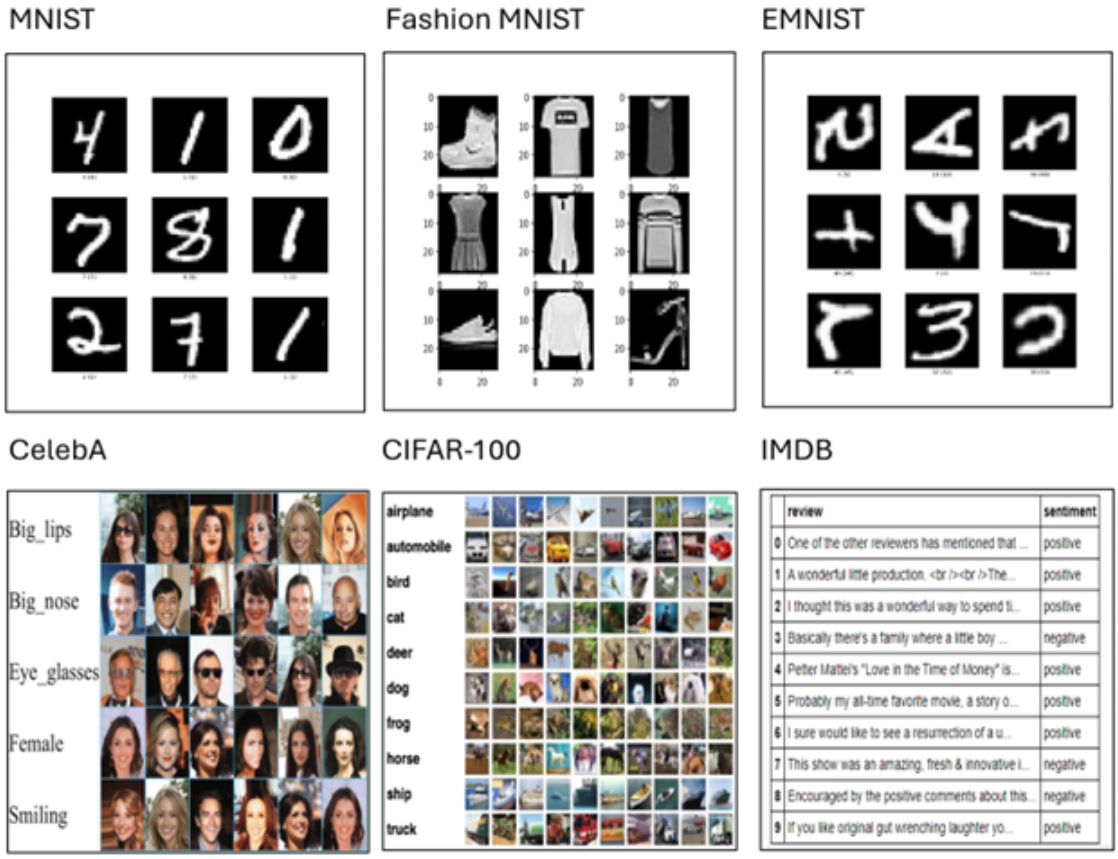


Figure 2.6: Different dataset types overview

2.12.1 MNIST (Modified National Institute of Standards and Technology dataset)

MNIST [2] dataset is pivotal in training and evaluating ML and image recognition algorithms. It features size-normalized and centered handwritten digits within 28 by 28 pixel frames, providing a uniform format for algorithmic input. These grayscale images, depicting digits from 0 to 9, are utilized in models to facilitate the recognition of handwritten characters. With a comprehensive collection of 60,000 images for training and an additional 10,000 for testing, the dataset enables developers to rigorously train and benchmark their algorithms against a standardized set of data.

2.12.2 Fashion-MNIST

Fashion-MNIST dataset [3] serves as an advanced alternative to the traditional MNIST [2] dataset by featuring images of various fashion items, including clothing and accessories, rather than handwritten digits. This dataset comprises 28×28 pixel grayscale images, each depicting a single fashion article, like shirts, purses, or shoes. With a total of 60,000 images for training and an additional 10,000 for testing, Fashion-MNIST [3] offers a more challenging dataset due to its diverse patterns and shapes, making it an excellent resource for training and evaluating ML models. The complexity and variability of the images in Fashion-MNIST provide a robust platform for algorithms to learn the nuanced differences between various types of fashion items, enhancing their capability to recognize and differentiating more complex visual patterns.

2.12.3 EMNIST (Extended Modified National Institute of Standards and Technology)

EMNIST [93] dataset, represents an expansion of the original MNIST [2] dataset, offering greater diversity and a larger volume of data. Comprising over 800,000 images, each 28 × 28 pixels, EMNIST includes a wide array of handwritten characters such as letters (in both uppercase and lowercase), numbers, and punctuation marks. This augmentation not only increases the complexity of the dataset but also broadens its applicability beyond the scope of the original MNIST's digit-focused collection. EMNIST's comprehensive range of handwritten characters makes it an invaluable resource for more intricate and varied ML tasks, including handwriting recognition and the development of models that can understand a broader spectrum of textual inputs.

2.12.4 FEMNIST (Federated Extended MNIST)

FEMNIST [93] dataset, an extension of the EMNIST dataset, incorporates 28 × 28 pixel grayscale images that depict not only handwritten digits but also both uppercase and lowercase alphabetic characters. The distinctive feature of FEMNIST is its distribution across numerous clients, thereby mirroring a real-world FL environment. This aspect of FEMNIST makes it an invaluable resource for training models in a decentralized manner, where data remains on the devices generating it, enhancing privacy and data security. It provides a practical scenario for testing and improving FL algorithms, offering a diverse range of handwriting styles and characters for comprehensive learning and evaluation.

2.12.5 CelebA (Celebrity Attributes)

CelebA [94] is a comprehensive dataset focusing on face attributes, featuring more than 202,599 images of celebrities. Each image is annotated with 40 distinct attributes, making this dataset a rich resource for facial attribute recognition tasks including categorization of faces, prediction of attributes, and modifications of facial features. Specifically designed to advance the field of facial analysis, CelebA facilitates a wide range of research and development activities by providing a vast and varied collection of annotated facial images. This dataset plays a crucial role in training models to accurately identify and analyze facial characteristics, thereby pushing the boundaries of what's possible in facial recognition technology and related applications.

2.12.6 Sentiment140

The Sentiment140 [95] dataset, stands as a pivotal resource for sentiment analysis and opinion mining research. Containing 1.6 million tweets, each is annotated with emoticons to signify sentiments as positive, negative, or neutral. These tweets are primarily in English and encompass a diverse array of subjects such as politics, sports, entertainment, and technology. This extensive coverage makes the Sentiment140 dataset an invaluable tool for training algorithms to detect and interpret sentiments expressed in social media, offering insights into public opinion across various domains. The use of emoticons as indicators of sentiment provides a unique approach to understanding the emotional tone behind the text, facilitating more nuanced and accurate sentiment analysis models.

2.12.7 CIFAR-100 and CIFAR-10

The CIFAR-100 [9] dataset, created under the auspices of the Canadian Institute For Advanced Research (CIFAR), is a cornerstone in the field of image classification and computer vision. As a carefully labeled segment of the extensive 80 million tiny images collection, it features images categorized into 100 detailed classes, which are further aggregated into 20 broader categories. This organization facilitates a nuanced approach to image classification tasks. Alongside CIFAR-100 [9], the CIFAR-10 [9] dataset provides a complementary resource, focusing on a subset of 10 classes. This extension includes 60,000 color training images and 10,000 test images, all in a compact 32 × 32

pixel format, making both datasets integral for developing and testing advanced image recognition algorithms.

2.12.8 IMDB (Internet Movie Database)

IMDB large movie review dataset [5], is specifically designed for binary sentiment classification tasks. It encompasses a compilation of 50,000 movie reviews, each tagged with a binary sentiment label indicating either a positive or negative sentiment. This dataset has been partitioned into two segments: 40,000 reviews for training and 10,000 for testing purposes. For analyzing this dataset, a Bidirectional Long/Short-Term Memory (BiLSTM) [96] model is employed, incorporating an embedding layer that assigns a 100-dimensional vector to each word. The architecture culminates in a fully connected layer, which, along with a sigmoid activation function, is responsible for generating the predicted sentiment of a given movie review. This structured approach enables a nuanced understanding and processing of natural language, offering insights into the sentiments expressed in the movie reviews.

2.12.9 EDGE-IIOTSET

Edge-IIoTset [97] is a comprehensive cybersecurity dataset designed for IoT and IIoT applications, aimed at enhancing ML-based intrusion detection systems in centralized and FL modes. The dataset is structured into seven layers, including Cloud Computing, Network Functions Virtualization, Blockchain networks, Fog Computing, Software-Defined Networking, Edge Computing, and IoT and Industrial Internet of Things (IIoT) perception layers. It incorporates advanced technologies like ThingsBoard IoT platform, Hyperledger Sawtooth, and Mosquitto MQTT (Message Queuing Telemetry Transport) brokers to address the requirements of IoT and IIoT systems. Data are generated from various IoT devices, covering more than 10 types, such as temperature and humidity sensors, ultrasonic, and heart rate sensors. The dataset identifies fourteen attacks across five threats: DoS/DDoS, information gathering, man-in-the-middle, injection, and malware attacks. Through analysis, it provides insights into the performance of ML approaches in detecting cybersecurity threats. In our exploration of various datasets including MNIST, Fashion-MNIST, CIFAR-10, CIFAR-100, IMDB, and others, it's imperative to distinguish between IID and Non-IID data configurations. IID datasets ensure that data samples are evenly distributed across classes, facilitating straightforward model training by providing a balanced representation. On the other hand, Non-IID datasets feature uneven distribution of classes or labels, posing challenges for model generalization and necessitating specialized approaches to handle the skewed data distribution. This distinction is crucial for the development of algorithms and the selection of models that are robust and adaptable to the diverse characteristics inherent in each dataset.

In the preceding sections, we examined FL in IoT, covering potential attacks, defense strategies, evaluation metrics, and datasets' pivotal role. Next, we'll focus on essential FL frameworks, crucial for IoT's secure, collaborative learning, integrating previous insights to grasp their application, and assessing frameworks and selection criteria.

Dataset	*Usage*	*Classes*	*Data type*	*Size*	*Dimensions*	*Remarks*
MNIST [2]	Handwritten digit recognition	10	Grayscale images	60,000 training, 10,000 testing	28x28	Benchmark for ML algorithms
Fashion MNIST [3]	Clothing item recognition	10	Grayscale images	60,000 training, 10,000 testing	28x28	More complex than MNIST, for algorithm testing
EMNIST [93]	Handwritten digits letters recognition	62 (10 digits + 26*2 letters)	Grayscale images	800,000	28x28	Extended MNIST, includes digits and letters
FEMNIST [98]	Handwritten digits letters recognition	62 (10 digits + 26*2 letters)	Grayscale images	3,550,000	28x28	FL setting, diverse and decentralized
CelebA [94]	Face attribute recognition	40 attributes	Color images	202,599	178x218	Large-scale face attributes dataset
Sentiment140 [95]	Sentiment analysis	2 (positive, negative)	Text	1,600,000 tweets	Variable	Tweets dataset, for sentiment analysis
CIFAR-100 [9]	Object recognition	100	Color images	80 million Images	32 × 32	Fine-grained object recognition differentiates between very similar subcategories of objects
CIFAR-10 [9]	Object recognition	10	Color images	50,000 training, 10,000 testing	32x32	Subset of CIFAR-100, less fine-grained
IMDB [5]	Sentiment analysis	2 (positive, negative)	Text	50,000 reviews	N/A	Movie reviews, for binary sentiment classification

Table 2.3: Comparative analysis of benchmark datasets in simulations.

2.13 Common FL frameworks

In the field of FL, numerous specialized frameworks have been created, integrating various FL algorithms to enable smooth real-world use. These frameworks are crucial for progressing the practical deployment and continuous advancement of FL. When selecting an FL framework, it is crucial to consider important features like client-side training, server-side aggregation, communication capabilities, and the presence of a local simulation mode. This segment introduces these practical frameworks, highlighting their fundamental features and the criteria that set them apart, to assist in choosing the most suitable tool for FL applications.

2.13.1 FL frameworks criteria

We explore the essential criteria guiding the selection of the FL frameworks:

- **Model compatibility:** ML models that FL frameworks can accommodate varies significantly. While neural networks are universally supported across these frameworks, support for tree-based models, such as gradient-boosted decision trees and random forests, is less common. Furthermore, certain frameworks offer functionalities for clustering and delivering insights on aggregated data sets.
- **Support for ML libraries:** Practically, the compatibility of ML libraries with FL frameworks is crucial. Most FL frameworks are designed to work with TensorFlow [99] and PyTorch [100] libraries. However, there are also universal FL frameworks that can integrate with any ML library, offering broader flexibility.
- **Scalability:** Most FL frameworks come equipped with pre-built FL models and aggregation algorithms, providing a foundational base for customizing ML models for federated environments. The ease of incorporating your own model or aggregation algorithm varies based on the model's specifics and the chosen FL framework.

- **Aggregation techniques:** The FedAvg [14] algorithm, known for its widespread use, is commonly supported by FL frameworks. Yet, the range of specific aggregation techniques each framework supports can vary.

- **Privacy approaches:** FL naturally enhances privacy by sharing model updates such as gradients or parameters, rather than raw data. However, it's possible to infer data information from these updates. For enhanced privacy, FL frameworks incorporate extra measures such as DP [40], HE [61], and SMPC [39] or secure aggregation.

- **Compatibility with devices and operating systems:** FL frameworks cater to a diverse range of client devices, from personal computers to smartphones and IoT devices. Although not all frameworks are compatible with every device type, when it comes to operating systems, many either support Linux/Unix, MacOS, and Windows directly or offer a containerized solution.

2.13.2 Overview of FL frameworks

We outline key FL frameworks, beginning with:

1. **Flower:** Flower, created by the University of Oxford's Systems and Networking Group [101] is a versatile, open-source FL framework. It offers compatibility with several ML libraries, such as TensorFlow, PyTorch, and Keras, catering to a wide range of FL applications and extending its usability to mobile platforms. For secure data aggregation, Flower employs the SecAgg and protocol [102], with a focus on supporting Horizontal Federated Learning (HFL)[13].

2. **pfl:** Python Framework for Private Federated Learning Simulations developed by Apple, pfl is an innovative, open-source Python framework designed specifically for private FL simulations, targeting the research community to enhance the efficiency and dissemination of FL research. Not intended for third-party FL deployments, pfl is invaluable for researchers validating FL methodologies through simulations. Key features include ease of initiation with existing models and data, rapid iteration across various computing resources, and flexibility in API design to accommodate new FL concepts. pfl supports large-scale, scalable experiments, is compatible with PyTorch and TensorFlow, and offers benchmarks for both. Support other models in addition to neural networks, e.g. Gradient Boosted Decision Trees(GBDT) [103]. Switching between types of models is seamless, and integrates comprehensive privacy mechanisms, demonstrating Apple's dedication to pioneering FL research and facilitating secure, innovative advancements in the field.

3. **PySyft:** PySyft is an open-source FL framework developed by the distributed computing initiative OpenMined. PySyft [104] is a Python-based DL library with a focus on privacy, built on top of the PyTorch framework. It inherits some of PyTorch's FL capabilities like data partitioning across devices. As an open-source framework by OpenMined, PySyft aims to enable privacy-preserving collaborative ML. To help protect sensitive data, PySyft incorporates encrypted computation protocols, DP mechanisms, dynamic and static computational graphs, and HE. However, as PySyft is primarily designed for simulations, it may lack some flexibility and support for large-scale collaboration offered by other FL frameworks tailored for real-world deployment. Still, PySyft provides appeal for researchers experimenting with and testing novel privacy-preserving DL techniques in a simulated federated environment.

4. **Federated AI technology Enabler (FATE):** FATE [105] is the world's first open-source industrial-grade framework for FL, launched by AI department of Webank. It facilitates secure data collaboration between institutions through DP, secure multiparty computation, and HE. As one of the earliest options ready for commercial use, FATE supports horizontal and vertical FL across various algorithms like logistic regression, tree-based methods, deep learning, and transfer learning. This enables enterprises and organizations to jointly analyze data while preserving data privacy and security.

5. **OpenFL:** OpenFL [106] is an open-source framework by Intel supporting horizontal FL. It has two components: Collaborators on end devices and an Aggregator. OpenFL enables user interaction through Python APIs or command line interface. Security is provided by mutual Transport Layer Security (TLS), a cryptographic protocol that provides communication security over a computer network, for mutual authentication between nodes. Data transmission optimization is available via lossless and lossy compression. OpenFL also uses Docker for containerization, improving security and reproducibility.

6. **IBM FL:** The IBM FL [107] framework, released in 2020 by IBM, is a flexible Python-based platform designed for FL. It offers essential tools for collaborative model development without being tied to a specific ML framework, enabling support for diverse learning strategies. The framework is compatible with DL and conventional ML approaches, including supervised, unsupervised, and RL [4] techniques. It HFL for data distribution, while SMPC [39] and DP [40] are integrated for enhanced security measures. IBM FL's adaptability and security features make it suitable for deployment on mobile platforms and by large corporations, ensuring its applicability in practical settings.

7. **NVFlare:** NVFlare [108], developed by Nvidia, is a comprehensive FL framework tailored for business applications, prominently supporting Nvidia Clara, a suite of AI healthcare products, and made open source in 2021. It accommodates a wide range of models, from neural networks to tree-based and statistical models, and is designed to be framework-agnostic, facilitating the migration of almost any ML model to a federated environment. NVFlare enables the setup of FL projects connecting servers, clients, and users with varied roles for training, evaluation, and monitoring, supporting job scheduling and parallel execution primarily through docker-compose. It emphasizes security with server-client authentication and privacy-enhancing technologies. While NVFlare aims to simplify transitioning from centralized to federated ML models, offering templates and extendable APIs for various models, it requires some effort to master its architecture, though it is noted for its flexibility compared to simpler frameworks like Flower.

8. **TensorFlow Federated TFF:** is a Google-developed [99], open-source Python framework for FL across decentralized servers. It enables the use of built-in FL algorithms and the distribution of a global model to clients. TFF consists of two main layers: the FL API, for integrating TensorFlow models into FL environments, and the Federated Core API, which allows for the creation of new federated algorithms by combining TensorFlow with distributed communication techniques.

9. **Federated Machine Learning (FedML):** FedML is an open-source framework developed to support the research and development community in exploring a wide array of FL algorithms and benchmarking their performance. It provides an extensive collection of experimental datasets and models, along with tools for performance measurement and analysis. FedML facilitates data partitioning through HFL, Vertical Federated Learning (VFL), and Federated Transfer Learning (FTL), and enhances security with DP [40] and SMPC [39]. It also includes a variety of state-of-the-art aggregation algorithms and supports different computing paradigms such as distributed computing, standalone simulation, and mobile on-device training, making it a versatile tool for FL experiments [109].

2.14 FLSecLAB: Our implemented FL framework for IoT networks

In our prior discussion, we examined the susceptibility of FL to poisoning attacks within IoT networks. To address this concern, we developed FLSecLAB, a dedicated FL simulation framework for IoT networks. Developed using the powerful capabilities of PyTorch and Scikit-learn, FLSecLAB aims to enhance the security of the FL system against these attacks. It introduces our innovative defense strategies, detailed in subsequent chapters, and incorporates proven defenses from the state of the art,

as mentioned in section 2.10. Moreover, FLSecLAB provides numerous customizable tools, allowing users to create various simulation scenarios to rigorously test and validate the resilience of FL system. This comprehensive approach establishes FLSecLAB as a vital tool for improving and evaluating FL security in IoT networks. The distinctive features of our framework are presented as follows:

2.14.1 Architecture of FLSecLAB: Server and Client entities

The architecture of FLSecLAB is organized around two primary roles within the FL environment: the Server and the client. Each of these roles is further divided into two main components, which are the CommunicationManager and the ComputationHandler/ModelTrainer, ensuring a clear and structured approach to FL in IoT networks. The architecture, is detailed as follows and demonstrated in Figure 2.7.

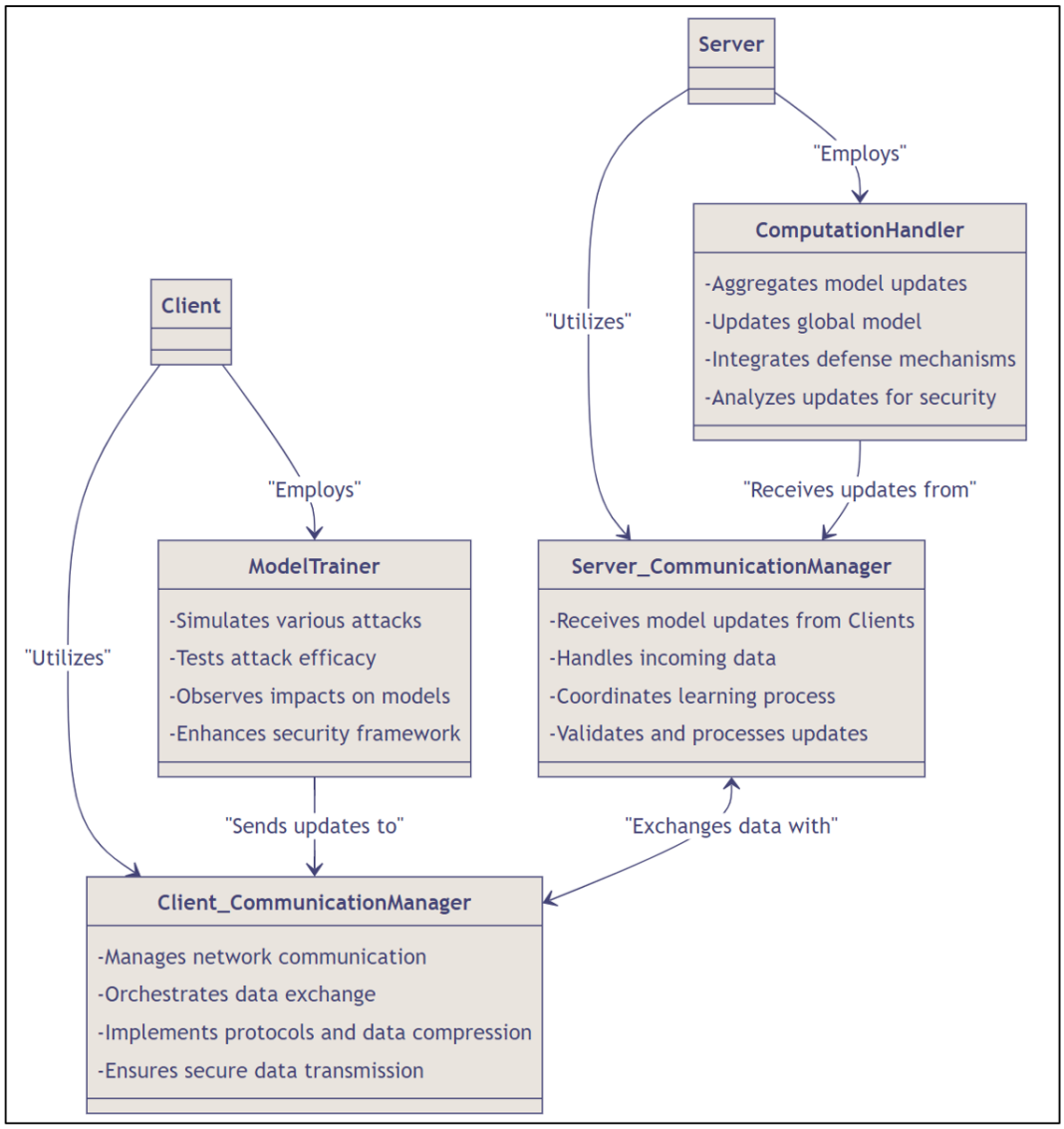


Figure 2.7: FLSecLAB architecture - Interactive roles and components in FL for IoT networks

2.14.1.1 CommunicationManager

The CommunicationManager module is central to both Server and Client entities, managing all aspects of network communication. This module establishes and maintains the network connections, orchestrates data exchange, and implements communication protocols and data compression schemes to optimize network usage and ensure secure data transmission.

- **Server's CommunicationManager:** Within the Server, the CommunicationManager is responsible for receiving model updates from multiple clients, handling incoming data efficiently, and coordinating the distributed learning process. It validates and processes the received updates, facilitating a synchronized and secure FL operation.

2.14.1.2 ComputationHandler and ModelTrainer

The ComputationHandler and ModelTrainer are essential components within the FLSecLAB framework, serving pivotal roles in computational tasks at the Server and Client levels, respectively:

- **ComputationHandler in server:** The ComputationHandler in the Server is pivotal for the core computational activities, including the aggregation of model updates from Clients and the subsequent updating of the global model. Beyond these fundamental tasks, it is instrumental for integrating and deploying new defense mechanisms within the Server. As it receives model updates via the CommunicationManager, the ComputationHandler meticulously assimilates this information to refine the global model. Concurrently, it conducts thorough analyses of the updates to identify and neutralize potential security threats such as data and model poisoning attacks. This dual role ensures that the ComputationHandler not only consolidates learning across the network but also fortifies the Server's defenses, actively contributing to the system's resilience and the reliability of the FL process.

- **ModelTrainer in client:**

  Within the client, the ModelTrainer is designed to facilitate the simulation and understanding of various attack methodologies. By incorporating the logic for different types of attacks directly within the local training process, this module allows for a hands-on examination of how specific attack vectors can influence the learning model. It provides a sandbox environment to test the efficacy of attacks and observe their impacts, which is crucial for developing and validating new defensive measures. The ModelTrainer's adaptability to simulate attacks makes it an invaluable component for enhancing the overall security framework of the FLSecLAB, ensuring that defense mechanisms are robust and effective against an array of potential threats. The delineated architecture of FLSecLAB, with its Server and Client roles coupled with their respective CommunicationManager and ComputationHandler/ModelTrainer components, establishes a robust and transparent framework. This structured architecture is designed to facilitate the deployment of secure and efficient FL within IoT networks, offering a clear methodology for integrating diverse IoT devices and ensuring the scalability and security of the FL process.

2.14.2 Customization of poisoning attacks in FLSecLAB

In addressing the imperative for stringent security in FL system, particularly those embedded within IoT networks, FLSecLAB introduces an elaborate array of customization options for simulating two predominant poisoning attacks: Label-flipping and Backdoor attacks. These configurations enable users to conduct nuanced attack simulations, facilitating an in-depth vulnerability analysis and the formulation of effective countermeasures.

2.14.2.1 Detailed configuration of Label-flipping attack

Label-flipping attacks, where adversaries invert the labels on a dataset to confound the learning algorithm, are a critical concern. FLSecLAB provides a sophisticated toolkit for customizing this attack:

- **Attack rate adjustment:** Users can manipulate the attack rate, varying from minimal to substantial interference, enabling the examination of the model's resilience under different levels of attack severity. For example, altering 30% of labels in a sensitive dataset could demonstrate the potential impact on diagnostic accuracy in a healthcare FL application.
- **Targeted label manipulation:** This functionality permits the strategic selection of labels for alteration, facilitating targeted attacks on specific dataset segments or classes. An illustrative scenario might involve selectively flipping labels on a subset of data critical to decision-making processes, such as altering traffic sign recognition classes in an autonomous vehicle dataset.
- **Comprehensive attack impact evaluation:** The platform encompasses advanced analytical capabilities to scrutinize the effects of Label-flipping, offering insights through various metrics and visualizations, like model ACC under attack and ASR, to ascertain the attack's effectiveness and scope.

2.14.2.2 Customization for Backdoor attack

Backdoor attacks implant covert functionalities within a model, activated by predetermined triggers. FLSecLAB's customization suite for this attack includes:

- **Simplified trigger pattern design:** This feature allows users to craft basic yet effective trigger patterns for activating the backdoor mechanism. For instance, a straightforward example could be the addition of a single, distinct color pixel in a corner of image data. When the model identifies this pixel during inference, it is programmed to misclassify the input deliberately. Such a trigger, despite its simplicity, can effectively test the model's vulnerability to Backdoor attacks and the robustness of its detection mechanisms.
- **Explicit payload effect definition:** The platform allows you to specify the desired outcome when the backdoor is triggered, assessing the model's vulnerability to different levels and types of malicious outputs. For example, in a sentiment analysis model, the payload might be designed to switch the sentiment of an input when a specific trigger phrase is included, challenging the model's integrity.

These advanced and flexible customization features empower FLSecLAB users to simulate, analyze, and understand a vast array of poisoning attack scenarios, significantly contributing to the development of robust and secure FL system within IoT contexts.

2.14.3 Integration of evaluation metrics in FLSecLAB

Building upon the foundational metrics discussed in Section [**sec:eval**], FLSecLAB offers a user-friendly interface that allows for the straightforward selection and application of various evaluation metrics to suit different scenarios. This flexibility ensures that researchers and practitioners can tailor their analysis framework to the specific requirements of each study or assessment, leveraging the predefined metrics to obtain relevant and insightful results.

- Users can effortlessly choose from a range of metrics such as ACC, LR, ASR, Recall, Precision and CPU run-time and CPU aggregation run-time, depending on the scenario at hand.
- This selection process is designed to be intuitive, allowing users to apply different metrics to different scenarios without the need to delve into the underlying complexities, as these are comprehensively covered in section 2.11.
- FLSecLAB's architecture supports this flexibility, enabling a dynamic approach to the evaluation of the FL system's robustness and effectiveness in various operational contexts.

By providing this capability, FLSecLAB enhances the adaptability and usability of the framework, promoting an efficient and targeted approach to the evaluation of FL security in IoT networks.

2.14.4 Advantages of PyTorch and Scikit-learn integration in FLSecLAB

The decision to base FLSecLAB on PyTorch [100] and Scikit-learn [110] was driven by a strategic alignment with the core principles of flexibility, efficiency, and community-driven innovation that these frameworks offer. This section delineates the multifaceted benefits stemming from this choice, underpinning the enhanced capabilities of our FL framework in addressing the security needs of IoT networks.

2.14.4.1 Enhanced flexibility and rapid prototyping

- PyTorch's dynamic computation graph enables more intuitive development and debugging of complex models, a feature particularly beneficial in the rapidly evolving domain of FL security. This flexibility facilitates quick experimentation and prototyping of new defense mechanisms against poisoning attacks, allowing our framework to stay at the forefront of FL security research.
- Scikit-learn's comprehensive suite of simple and efficient tools for data mining and data analysis empowers our framework with a wide array of algorithms and utilities for preprocessing, reducing the time from concept to deployment. This integration enhances FLSecLAB's adaptability to different datasets and security scenarios, ensuring robust performance across diverse IoT environments.

2.14.4.2 Scalability and performance

- Leveraging PyTorch's optimized performance for both CPU and GPU computations, FLSecLAB is capable of efficiently handling the computational demands of FL in IoT networks, which often entail processing large volumes of data across diverse devices. This ensures that the framework remains scalable and performant, even in resource-constrained IoT settings.
- The efficient implementation of ML algorithms in Scikit-learn [110] contributes to the overall performance of FLSecLAB, enabling rapid evaluation and iteration of FL models. This is crucial for assessing the effectiveness of various defense strategies against poisoning attacks under different conditions.

2.14.4.3 Community support and ecosystem synergy

- By aligning with PyTorch [100] and Scikit-learn [110], FLSecLAB benefits from the extensive ecosystems of these libraries, including a wealth of documentation, community forums, and pre-existing codebases. This ecosystem synergy facilitates the integration of new features and the latest advancements in ML and cybersecurity, keeping FLSecLAB at the cutting edge.
- The strong community support surrounding PyTorch and Scikit-learn accelerates the troubleshooting process and fosters collaboration, enabling the FLSecLAB team to leverage collective knowledge and best practices in FL and IoT security. This collaborative environment enhances the framework's reliability and effectiveness, ensuring it remains responsive to emerging threats and technological shifts.

2.14.5 Key features of FLSecLAB

a. **IoT-focused custom implementation:** Recognizing the distinct characteristics of IoT networks, FLSecLAB is engineered from the ground up to cater to this domain. Its design is optimized for the heterogeneous nature of IoT devices, ensuring efficient and secure FL participation across a broad spectrum of device capabilities.

b. **Integration of defense approaches:** FLSecLAB utilizes a range of defense techniques specifically designed to prevent poisoning attacks. These include innovative strategies we developed and refined during our thesis, which we will detail in the upcoming chapters. Alongside our new approaches, We have carefully redeveloped or integrated selected defenses from state-of-the-art

research to enable a detailed comparison with our proprietary defense strategies, which are explained in the subsequent chapters. This comparison is crucial in illustrating the effectiveness of our methods and placing them within the broader context of existing research, thereby significantly enhancing the security framework of FL system. By juxtaposing our innovations against established benchmarks, we ensure that our defenses are both effective and at the forefront of the latest advancements in the field. By evaluating our new defenses alongside proven strategies, we aim to establish a comprehensive defense framework that effectively addresses various vulnerabilities and attack methods, thus strengthening the security of the FL system against deliberate attacks.

c. **Wide dataset compatibility:** FLSecLAB's flexibility is demonstrated by its ability to accommodate a wide range of datasets, such as MNIST [2], Fashion MNIST [3], CIFAR10 [9], IMDB [5], and Sentiment140 [95] and others, including both IID and Non-IID data distributions. This guarantees that the platform can effectively tackle FL challenges across various data types and application domains, ensuring its adaptability to real-world scenarios where data distribution among clients can be highly heterogeneous.

d. **Customizable FL settings:** The framework offers extensive customization options for FL operations, including detailed settings for managing poisoning attack scenarios (such as Label-flipping and Backdoor attacks), controlling client participation, and selecting aggregation algorithms. This level of control enables precise tailoring of FL deployments to specific security and performance requirements.

e. **Comprehensive evaluation metrics:** FLSecLAB utilizes a wide range of metrics to assess the efficiency and security of FL system, such as ACC, LR, ASR, F1-Score, Recall, Precision, CPU run-time, and CPU aggregation run-time. These metrics offer a comprehensive perspective on the framework's performance across different conditions and attack scenarios.

f. **Modularity and extensibility:** Designed with modularity and extensibility in mind, FLSecLAB allows for the easy integration of new datasets, defense mechanisms, attacks, and evaluation metrics. This architectural choice ensures that the framework can evolve in response to new threats and advances in FL research.

Our work provide a substantial contribution to the enhancement of security in FL within IoT networks through the development of FLSecLAB. This platform is designed for the thorough evaluation of FL system against poisoning attacks, incorporating a variety of defense mechanisms. It supports multiple datasets and offers extensive customization capabilities, marking a significant step forward in the quest for secure and efficient FL implementations in the IoT domain. The aim of FLSecLAB is to deepen the understanding of security challenges in FL and to set the stage for future research and advancements in this vital field of security and privacy.

2.14.6 FLSecLAB setup overview

Navigating through the FLSecLAB framework, we explain an example of its configuration process; an intuitive sequence designed to facilitate the systematic setup of a FL scenario as demonstrated in Figure 2.8. As we progress through the interface, we detail the crucial steps required to tailor the parameters for conducting robust FL experiments and simulations. The framework is designed to be highly adaptable, allowing for the easy integration of any new attacks, defense strategies, datasets, and other options, thus enhancing the flexibility and capability of our framework to create and evaluate various simulations to address evolving challenges in FL system .

a. **FL details**: In Figure 2.9, this initial setup involves selecting the ML framework, such as PyTorch [100] or Scikit-learn [110]. We configure key aspects of the FL setup, such as the number of local epochs, total rounds, batch size, learning rate, attack ratio, and the aggregation algorithm (e.g., FedAvg [14]).

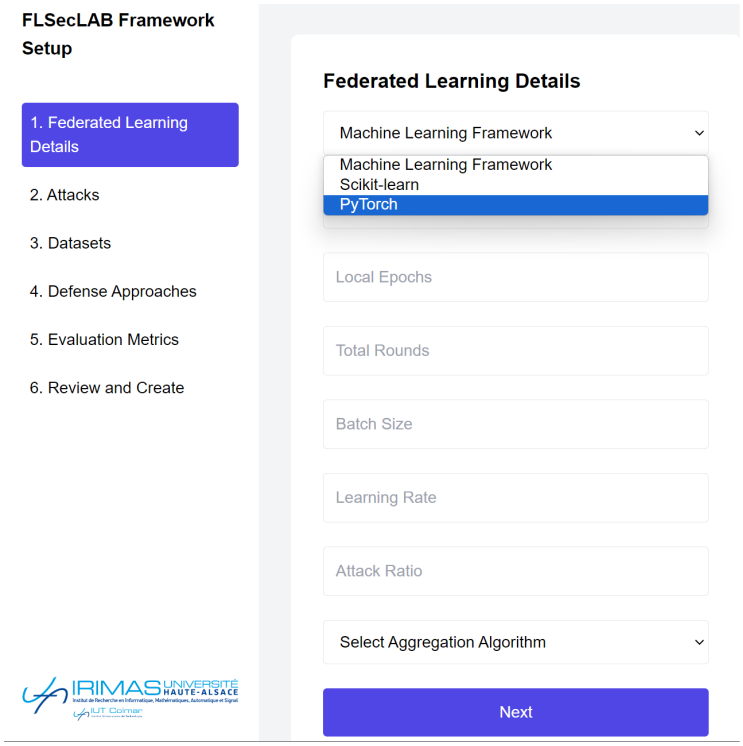


Figure 2.8: FLSecLAB setup overview

b. **Attacks**: In Figure 2.10, we outline the adversarial attacks to simulate in the FL scenario, such as "Backdoor" and "Label-flipping" attacks. Multiple attacks can be selected simultaneously to represent various strategies that malicious clients might employ to compromise the FL process. Moreover, attacks can be customized over the global configurations by choosing parameters such as the number of classes, the source class, and the target class in Label-flipping attacks.

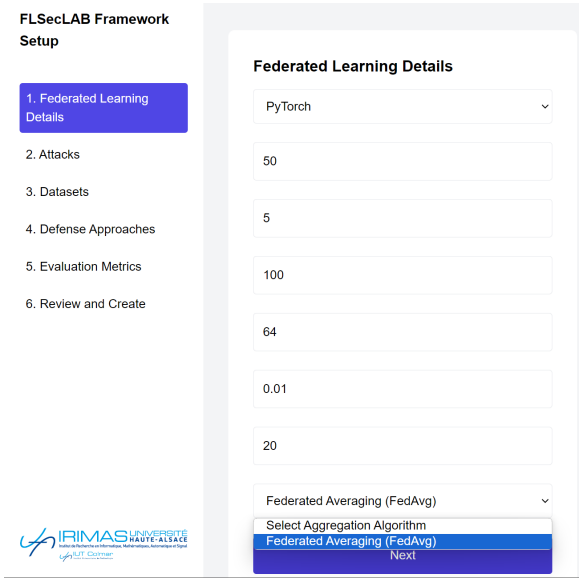


Figure 2.9: FLSecLAB: FL configuration

Figure 2.10: FLSecLAB: FL attacks selection

c. **Datasets**: In Figure 2.11, multiple datasets can be chosen for the current FL scenario, like MNIST [2], Fashion-MNIST [3], and CIFAR10 [9], or Non-IID datasets like MNIST-Non-IID and CIFAR10-Non-IID. These datasets can be modified over the global configuration such as batch size and the number of local epochs.

d. **Defense approaches**: In Figure 2.12, multiple defense approaches can be selected to safeguard the FL process against specific attacks, including the existing approaches from the literature, such as D2MIF [74] and FoolsGold [69], among others. Additionally, our unique proposed defense strategies are EMDG-FL, M3D-FL, ERD-FL, and NAM2D-FL, or our potential future approaches.

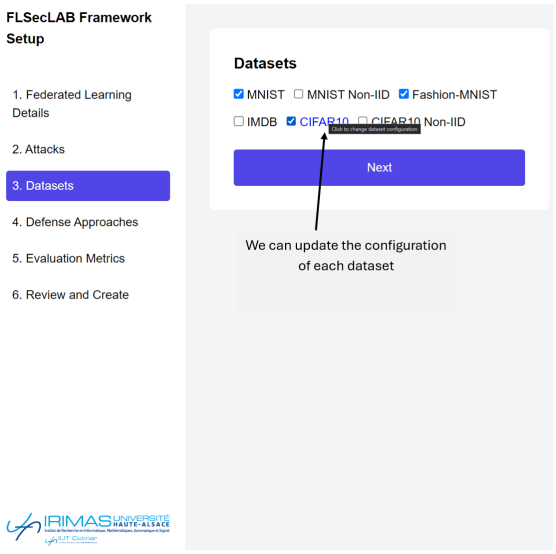


Figure 2.11: FLSecLAB: Datasets selection

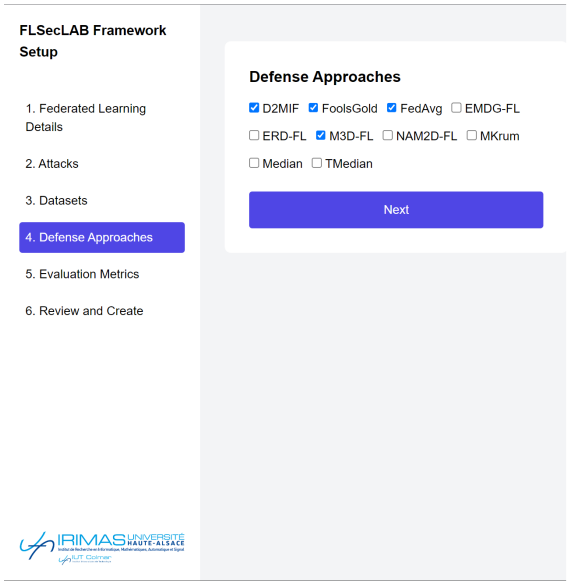


Figure 2.12: FLSecLAB: Defense approaches selection

e. **Evaluation metrics**: In Figure 2.13, we can select multiple metrics to evaluate the FL scenario's performance. Metrics such as ACC, ASR, LR, CPU aggregation run-time, CPU run-time, Precision, and Recall are critical for assessing the robustness of the FL system.

f. **Review and create**: Finally, in Figure 2.14, we review all the configurations made in the previous steps. This overview includes the FL details, selected attacks, datasets, defense approaches, evaluation metrics, and the aggregation algorithm. Once everything is verified, we proceed with the "Create and Go" option to initiate and evaluate the FL scenario with the specified parameters.

Each of these steps guides the user through configuring a comprehensive experiment or simulation in FL, considering various aspects that affect the performance and security of the FL system.

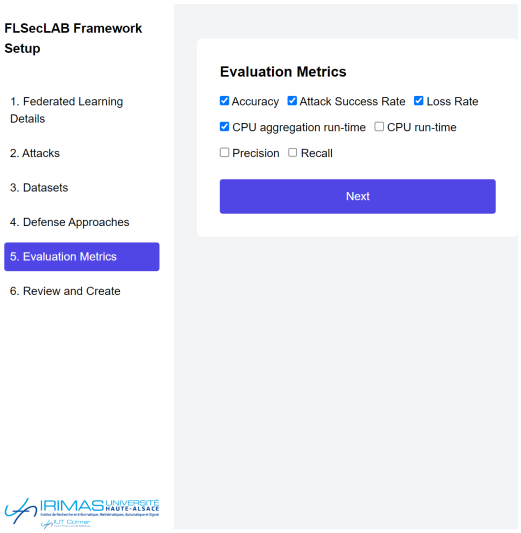


Figure 2.13: FLSecLAB: Evaluation metrics selection

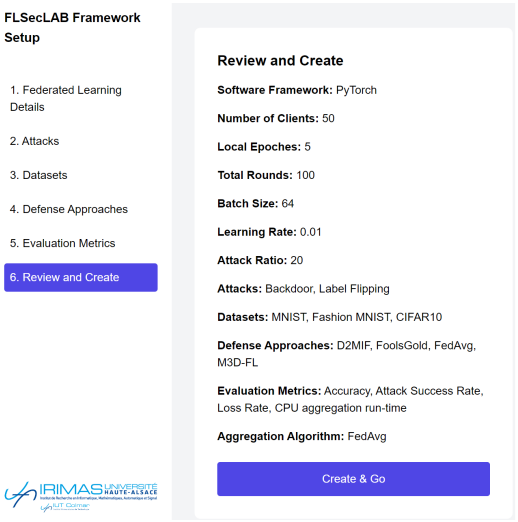


Figure 2.14: FLSecLAB: FL setup report

2.15 Conclusion

In this chapter, we explored the role of FL within IoT networks, emphasizing its potential to enhance collaborative model training while ensuring data privacy. We highlighted FL's ability to harness the vast data from IoT devices, improving learning outcomes without risking data privacy or security. Despite its benefits, FL's integration into IoT networks brings challenges, notably its vulnerability to poisoning attacks which threaten model integrity and security. After that, We provide a comparative study to evaluate the existing defenses against such attacks, specifically the malicious model detection techniques. Then, we introduce the FLSecLAB a robust and adaptable platform that integrates both our newly proposed defenses and those previously presented in this chapter, effectively safeguarding against poisoning attacks, and supporting diverse datasets and performance metrics for comprehensive FL system evaluation. FLSecLAB marks a key advancement in securing FL for IoT networks, enhancing system evaluation and defense capabilities. Our upcoming chapters will delve into our novel defense strategies, expanding on this foundation to enhance FL system security and efficiency. In the next chapter, we propose our first approach to detect malicious clients against Label-flipping attacks in FL system within IoT networks.

Chapter 3

Enhanced malicious model detection based on genetic algorithm for federated learning in IoT networks

Contents

3.1 Introduction

In this chapter, we introduce EMDG-FL, a novel approach to enhance the detection of malicious models in the FL process using an optimal threshold calculation method based on GA [1] against Label-flipping attacks. We utilize a GA to select the optimal threshold value that maximizes the accuracy of the server global model.

Based on the literature review performed in chapter 2, it can be concluded that several techniques can be employed to protect against poisoning attacks for FL in IoT networks. However, we notice that these works still have some limitations and potential drawbacks that need to be carefully considered and addressed in future research. One of the main limitations identified in the existing works discussed in section 2.10, is that some of them like [74], [69], and [70] have high computational cost. Also, for the work presented in [70], the client-side detection and testing updates can be computationally expensive. If the client devices have limited computational resources, this can limit the effectiveness and efficiency of the scheme. The FoolsGold [69] work, introduces additional computational overhead for computing similarity between client update vectors. This overhead is more significant for shallow ML models trained on CPUs compared to DL models trained on GPUs. For the D2MIF [74], the continuous adjustment of a threshold value can lead to instability and slow learning. In addition, RL [4] is computationally demanding and time-consuming to train, necessitating extensive interactions with the environment to learn an optimal policy, which can be impractical for real-world problems with complex environments or time constraints. Hence, the constraints identified in previous studies serve as the driving force behind our present research efforts. In this study, we employ a GA to improve

the detection accuracy of malicious models while simultaneously mitigating the computational burden associated with the proposed approach. We conduct simulations in Python to assess the performance of our approach using the MNIST [2] and Fashion-MNIST [3] datasets. We compare our approach with three other works [74], [69] and [14] from the literature in terms of ACC, ASR, LR, and CPU aggregation run-time. In the following sections, we provide a detailed description of our new approach. Next, we present our experiments setup. Subsequently, we present the outcomes of our simulations.

3.2 EMDG-FL proposed model

Our EMDG-FL method enhances FL security by introducing a threshold calculation using GA[1] for identifying malicious clients against Label-flipping attacks. This approach assesses the integrity of each client's model submission by computing a MS with the IForest [7] algorithm, ensuring accurate detection of malevolent entities. The models received by the server are represented as set $A = \{m^0, m^1, ..., m^i, ..., m^{N-1}\}$, where m^i and m'^i are the normal and malicious local models.The model parameters, originally in multi-dimensional matrix form, are unsuitable for direct application in IForest anomaly detection. To address this, we transform these matrices into 1D vectors. This transformation achieves two key objectives: it allows each matrix position to be considered as an independent feature, and it simplifies the integration of the IForest algorithm.

The IForest algorithm works by building isolation trees based on the clients' number N and the list of clients' models A received by the server, where instances falling in leaf nodes closer to the root are considered more anomalous. The path length to reach a leaf node determines the $MS(m^i)$, which is normalized by the average path length $Avg(m^i)$ to produce the MS. The MS is determined using the formula expressed in equation 3.2.1.

$$\mathrm{MS}(m^i) = 2^{-\frac{Avg(m^i)}{F(N)}} \tag{3.2.1}$$

The m^i represents the local model uploaded by a participant, and the $Avg(m^i)$ represents the average path length from the root to the leaf nodes, where each leaf node represents a client's local model. The $F(N)$ represents the normalization term as presented in equation 3.2.2, and N represents the number of clients.

$$F(N) = 2G(N-1) - (2(N-1)/N) \tag{3.2.2}$$

The $G(i)$ signifies the harmonic number and can be approximated as ln(i) + β (Euler's constant). As $F(N)$ signifies the average $G(i)$ for a given N, we employ it to normalize $G(i)$.

In Fig. 3.1, we present a detailed flow chart of our proposed system, where the process of our EMDG-FL approach commences by: First, The server calculates the initial accuracy of the global model and sends it to all clients. Afterward, the clients train their local models with their data. Next, the clients upload their trained models to the server. The server calculates the *pre-accuracy* of all the currently received local models. To proceed with aggregation, the server filters malicious local models by checking if their pre-accuracy is greater than the initial accuracy of the global model. If so, the server aggregates these trained models. Otherwise, the server calculates its MS using the IForest algorithm [7] and applies the GA[1] to find the optimal threshold combination. This GA[1] process includes several steps:

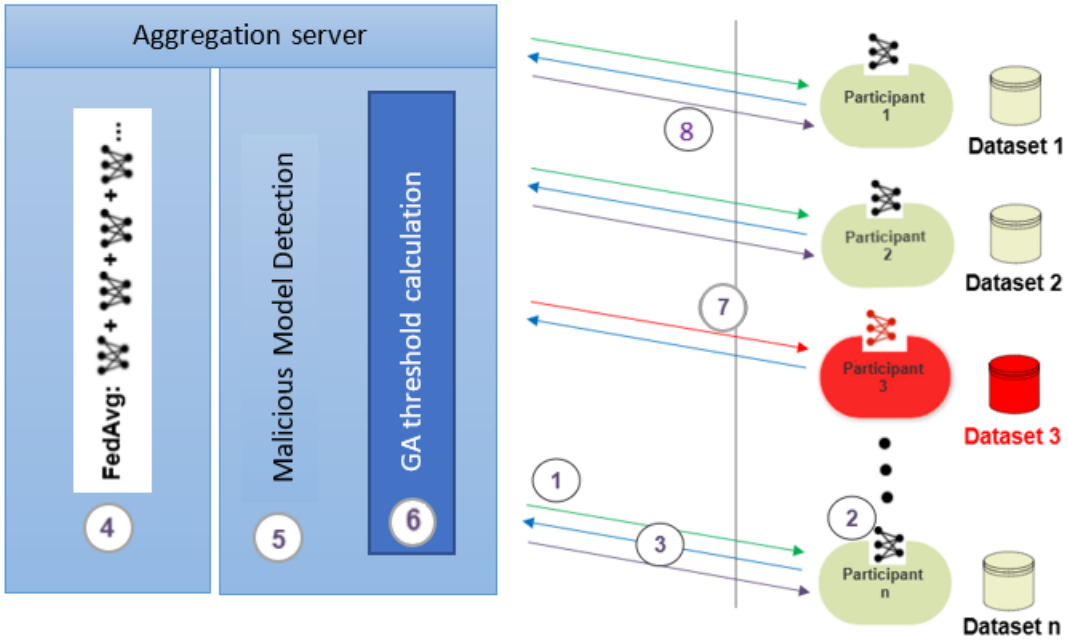
a **Encoding:** Each client's prediction accuracy (P) is encoded as a gene, and all clients' genes represent the chromosome.

b **Decoding:** Decode the chromosome to determine the range of values for P and the threshold T. P and T are in the range [0.8, 0.95].

c **Fitness calculation:** The fitness function F is equal to the sum of $|P - 100\%|$ for all clients, a higher fitness value indicates a better solution.

d **Selection:** Select the chromosomes with higher fitness as parents for the next generation.

e **Crossover:** Combine genes from two parent chromosomes to create new offspring chromosomes.



(1) Send the global model to participants **(2)** Local model training by participants using own data **(3)** Participant upload own model to the server **(4)** Local models updates aggregation **(5)** Detect malicious participants **(6)** Apply the optimal threshold calculation **(7)** Eliminate malicious participants **(8)** send the new global model aggregated to participants.

Figure 3.1: EMDG-FL approach against Label-flipping attacks

f **Mutation:** Randomly change some genes in the offspring chromosomes to maintain diversity in the population.

g **R**epeat: steps from c to f for multiple generations until an optimal solution is achieved.

The individual with the maximum fitness represents the optimal threshold T and associated subset of clients used to determine threshold T. We opted for this fitness function due to its capacity to measure the absolute deviation between accuracy and the ideal accuracy (100%). By maximizing the fitness score, we effectively minimize this deviation. This minimization is most pronounced when accuracy closely approaches 100%. Consequently, this fitness function serves as a robust tool for pinpointing the optimal accuracy threshold T. The process ends if a combination's fitness is no longer improved over a certain or a maximum number of iterations $MaxIteration$ is reached.

The maximum value of the fitness function calculated in Algorithm 2 is subsequently employed as the optimal threshold T in the malicious model detection process outlined in Algorithm 1. After that, the server detects the malicious clients by comparing the MS of each model with a predefined threshold T calculated using the GA [1]. Lastly, if the MS of a model is greater than the threshold T, The server marks the model as malicious and rejects it from the aggregation. This action signifies that the model is considered to be intentionally malicious or potentially harmful. Therefore, if the client's marks are greater than or equal to 3, then this client is definitively removed from the FL process. After aggregating the local model updates from normal clients and eliminating any updates from malicious clients, the server saves the new global model that has been aggregated from the remaining normal clients. The server then sends this latest version of the global model back to the clients to update the parameters in their local models. The EMDG-FL approach involves several steps to ensure the integrity and reliability of the local models submitted by clients and prevent malicious actors from compromising the FL process against Label-flipping attacks. The MS is used to determine if a local model is malicious or not, and a threshold is set to filter out any models with

scores above it. Our approach unfolds in two detailed algorithms: First, Algorithm 1 outlines our technique for detecting malicious models, emphasizing accuracy and efficiency. Then, Algorithm 2 guides us through calculating the optimal threshold, ensuring our detection is both precise and reliable. These steps form the core of our method, blending simplicity with effectiveness to tackle sophisticated challenges.

Algorithm 1: EMDG-FL malicious model detection

```
m_list = 0.          // A list keeps track of how often each client submits a model identified as malicious.
The server calculates the initial model's accuracy and sends the global model to clients.
Every client trains its local model using its own data.
The server Receives the local models uploaded by clients.
The server calculates local models pre-accuracy.
if pre-accuracy > initial model's accuracy then
    The server aggregates the local models of clients.
else
    The server calculates MS using the IForest algorithm [7] for each local model (m^i)
    if MS(m^i) > threshold (T) then
                                                    // T is calculated in algorithm ??
        The server marks (m^i) as malicious and rejects it from the EMDG-FL system.
        m_list[m^i] = m_list[m^i] +1.    // The server labels the participant who submitted the malicious model as sensitive.
        if m_list[m^i] >= 3 then
            The server removes this client.
    else
                                                    // Local model is not deemed malicious
        The server aggregates this model with other normal models using the FedAvg [14] algorithm.
```

Algorithm 2: Optimal threshold calculation

Input: *pre-accuracy*, $MaxIteration$ // *pre-accuracy* is calculated in algorithm 3, $MaxIteration$ is the maximum number of iterations.

Output: Threshold T

1 Encoding: Generate an initial population $G0$. // random group of clients.

2 Decoding: determine a range for T in $[80\%, 95\%]$.

3 Fitness estimation: // Find the maximum value of fitness function.

4 $F = \sum |ACC_i - 100\%|$

// ACC_i is the accuracy of each client in the population, the value of *pre-accuracy* used in $G0$ is the *pre-accuracy* calculated in algorithm 3.

5 Selection: Select the client with a higher fitness value.

6 Cross and mutation.

7 Repeat 5,6 steps until the optimal fitness value is obtained or the $MaxIteration$ is reached.

8 $T = F$.

3.3 Experiments setup

In this section, we provide a description of the hardware environment, the datasets' models, and the attack scenarios implemented in our simulations for our novel defenses. We employ a range of datasets described in Section 2.12, enabling a thorough and varied evaluation.

3.3.1 Hardware environment

We use a client-server model, and all experiments are performed using PyTorch $V1.13$ [100] and Scikit-learn $V1.2.2$ [110] on an Ubuntu Server environment which is Intel Core i7-10700 CPU, 16GB RAM. Furthermore, it is assumed that the FL server is secure and not compromised.

3.3.1.1 Datasets's models

In this simulation, we utilized the MNIST [2] and Fashion-MNIST [3], all the datasets' model presented as follows:

- **MNIST:** Implemented a two-layer CNN [80] with two fully connected layers, totaling approximately 22K parameters.
- **MNIST-Non-IID:** Applied a Dirichlet [90] distribution ($\alpha = 1$) for Non-IID data, conducting 200 training iterations with 3 local epochs and a batch size of 64, the clients utilized the cross-entropy loss function and applied the SGD [89] optimizer, configuring a learning rate of 0.01 and a momentum of 0.9, for training their models
- **CIFAR10:** Evenly allocated CIFAR10 data and employed ResNet18 CNN model [111] for 100 iterations with 3 local epochs and a batch size of 32, utilizing cross-entropy loss and SGD [89] (learning rate 0.01, momentum 0.9).
- **CIFAR10-Non-IID:** Adopted a Dirichlet [90] distribution ($\alpha = 1$) for Non-IID client data, with training parameters consistent with CIFAR10 [9] to generate Non-IID data for the clients.
- **IMDB:** We employed a BiLSTM [96] model with an embedding layer that assigns every word to a 100-dimensional vector. The model concludes with a fully connected layer and a sigmoid function to generate the ultimate predicted sentiment for a review input (approximately $12M$ parameters).

3.3.1.2 Attack scenarios across datasets

This section describes two primary attack strategies applied to various datasets: Label-flipping and Backdoor attacks, each demonstrated with a clear example.

- **CIFAR-10 and CIFAR-10-Non-IID:** Attackers change 'Dog' images to 'Cat' labels in Label-flipping attacks. In Backdoor attacks, they place a 3×3 white pixel square in 'Car' images at the bottom-right and wrongly label them as 'Plane'.
- **MNIST and MNIST-Non-IID:** A simple Label-flipping attack might mislabel a '3' as an '8'. Backdoor attacks involve placing a 3×3 white pixel square on '9' images, reclassifying them as '0'.
- **Fashion-MNIST:** In Label-flipping, a 'Dress' might be incorrectly tagged as a 'T-shirt'. For Backdoor attacks, adding a 3×3 pixel pattern to a 'Bag' image could lead to it being labeled as 'Shoe'.
- **IMDB:** In Label-flipping, reviews marked as 'positive' become 'negative'. For Backdoor attacks, inserting a unique word or phrase in reviews and changing their sentiment illustrates the approach for text data.

3.4 Simulation results analysis

In this section, we detail our simulation results, evaluating the impact of the Label-flipping attacks introduced in section 2.9.1, within a 30-client FL system in IoT networks. We point out that we have implemented the D2MIF [74] and the FoolsGold [69] approaches based on the author's description using our FLSecLAB framework presented in section 2.7, assessing them according to ACC, ASR, LR, and the CPU aggregation run-time. We have analyzed our approach with two image-based classification datasets, MNIST [2] and Fashion-MNIST [3], that explained with details in section 2.12 and their model in section 3.3. In this simulation, we randomize the attackers' proportion, ensuring it does not exceed 20% of the participating clients in the FL process. We adjust the value of $MaxIteration$ from 10 to 100, increasing in increments of 10. This decision is based on preliminary experiments that indicated that increasing the number of GA generations beyond 100 does not significantly improve accuracy but does lead to a considerable increase in the time taken for execution. Therefore, we have selected this specific range (10 to 100) to find an optimal balance between the potential for improved accuracy and the need to manage computational expenses effectively.

In the first experiment, we evaluated our approach regarding average CPU aggregation run-time demonstrated in section 2.11 and compared it with the studied approaches discussed in section 2.10 D2MIF [74] and FoolsGold [69] methods. As we show in Table 3.1, our approach has the lowest CPU aggregation run-time compared with the other approaches. The D2MIF entails a computational expense in their methodology due to the utilization of a costly RL [4] process for threshold selection.

Table 3.1: Average CPU aggregation run-time

	D2MIF [74]	***EMDG-FL***	***FoolsGold [69]***
MNIST	0.06 s	0.02 s	0.07 s
Fashion-MNIST	0.10 s	0.05 s	0.08 s

In the second experiment, we assessed our EMDG-FL approach using the ACC performance metric, as described in section 2.11. The EMDG-FL approach compared to other studied approaches illustrated in section 2.10: D2MIF [74], FoolsGold [69] and FedAvg [14] approaches under the existing of the Label-flipping attacks for MNIST [2] and Fashion-MNIST [3] datasets. We show in Figure 3.2 that, the FedAvg [14] method has the lowest ACC as it lacks defence against attacks. Concerning our approach, it shows the highest ACC, outperforming FedAvg, D2MIF, and FoolsGold methods.

In the third experiment, we evaluated our EMDG-FL approach using the ASR metric presented in section 2.11. which refers to the percentage of attempts by malicious clients to undermine the FL process. Results in Figure 3.3 confirm that FedAvg [14] with no defenses exhibits the highest attack vulnerability, with nearly 100 percent of poisoning attempts succeeding in degrading accuracy. In comparison, EMDG-FL demonstrates significantly enhanced attack resilience, allowing the fewest poisoning attempts to influence the global model. This substantial reduction in ASR can be attributed to EMDG-FL's genetically optimized threshold mechanism for detecting and filtering poisoned model updates. By tuning the threshold more precisely using GA [1], EMDG-FL blocks a greater proportion of malicious updates from FedAvg [14], D2MIF [74], and FoolsGold [69]. EMDG-FL shows the best defense against Label-flipping attacks by reducing ASR through our proposed threshold optimization technique.

In the fourth experiment, we evaluated the EMDG-FL approach in terms of the LR performance metric, as detailed in section 2.11. with different iterations' numbers. Figure 3.4 shows the LR for our EMDG-FL, D2MIF [74] and FoolsGold [69] approaches, with the presence of Label-flipping attacks, using the MNIST [2] dataset with varying numbers of iterations. The LR starts to decrease from the first iteration for all the studied approaches, the LR of our EMDG-FL system drops sharply in the first 10 iterations, then maintains a low level. The D2MIF and FoolsGold approaches require up to thirty iterations to achieve a significant decrease. Our results indicate clearly that the EMDG-FL approach surpasses the other studied approaches, with lower LR values.

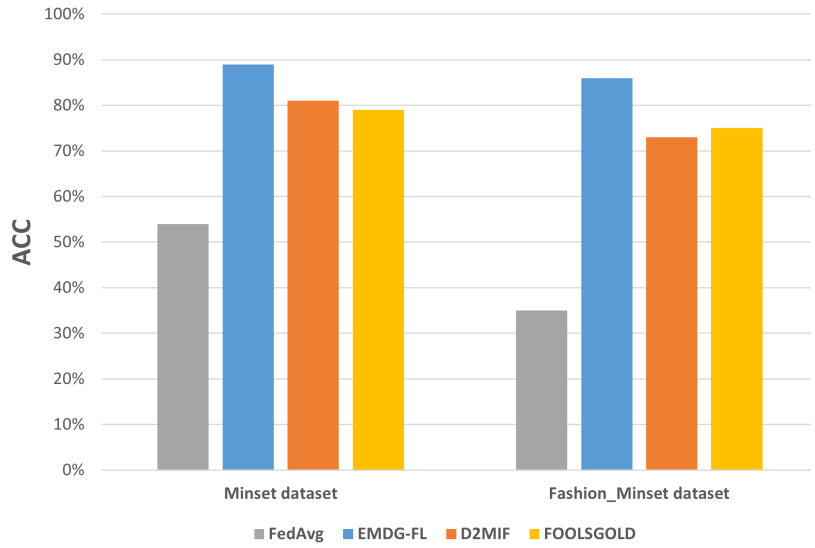


Figure 3.2: ACC of the four approaches using two different datasets against Label-flipping attacks

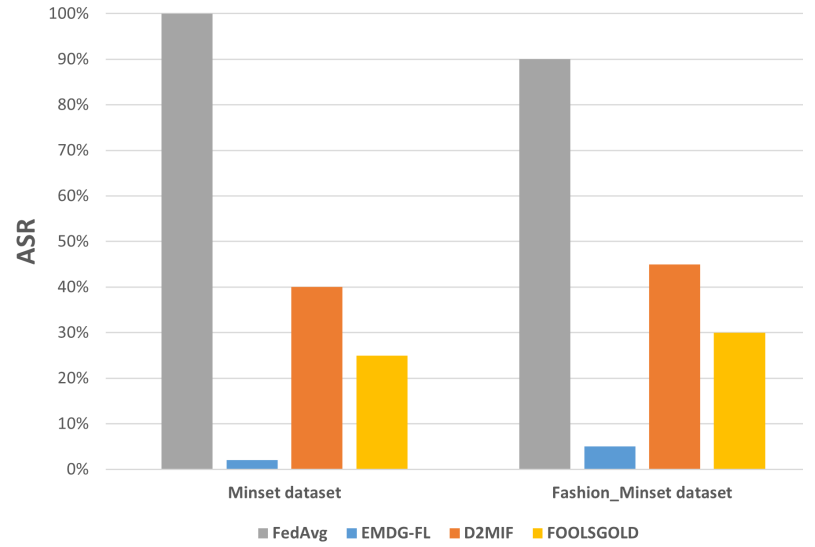


Figure 3.3: ASR of the four approaches using two different datasets against Label-flipping attacks

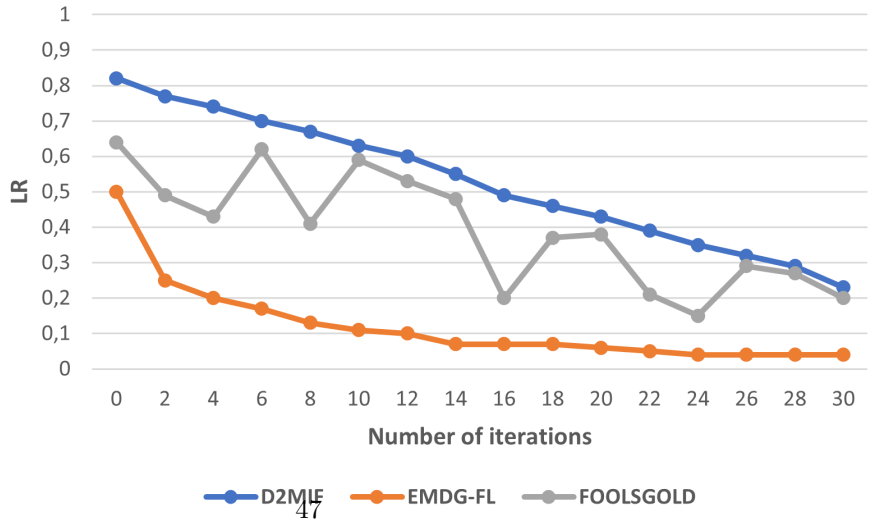


Figure 3.4: LR of three approaches against Label-flipping attacks using MNIST dataset with different iterations

Figure 3.5 compares the LR of our newly proposed EMDG-FL, D2MIF [74] and FoolsGold [69] approaches under the influence of Label-flipping attacks, using the Fashion-MNIST [3] dataset with varying numbers of iterations. Specifically, the EMDG-FL achieves a very low LR after ten iterations, while the D2MIF and FoolsGold require up to twenty-five iterations to reach a low level of LR that gradually increases over time. Our results indicate clearly that the EMDG-FL approach outperforms the studied approaches, with lower LR values.

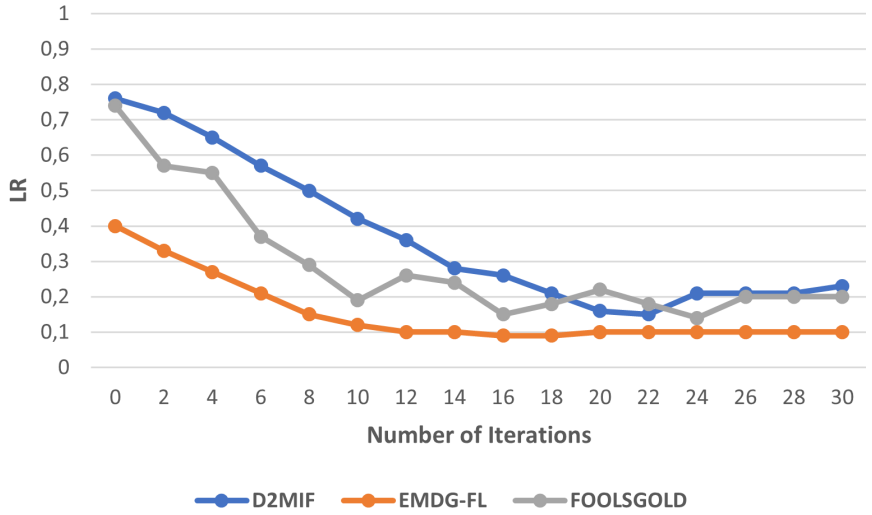


Figure 3.5: LR of three approaches against Label-flipping attacks using Fashion-MNIST dataset with different iterations

3.5 Conclusion

In this chapter, we have introduced EMDG-FL, a novel approach designed to enhance the malicious model detection of FL system against Label-flipping attacks in IoT networks. By leveraging the capabilities of GA for the determination of an optimal threshold, our work seeks to not only protect the integrity of FL system but also to improve its operational efficiency. This research was grounded in thorough simulations and comparative analyses, utilizing the MNIST and Fashion-MNIST datasets as benchmarks. Through these detailed evaluations, EMDG-FL has proven to be superior, surpassing other existing approaches introduced in section 2.10: D2MIF, FoolsGold, and FedAvg across key performance metrics including ACC, ASR, LR, and aggregation computational cost. The next chapter presents a novel scalable approach to address the limitations of the FL system. This approach aims to broaden our understanding and enhance the FL performance across a wider range of clients and dataset types, advancing FL research by overcoming existing challenges and expanding its applicability.

Chapter 4

Entropy-driven robust defense for federated learning in IoT networks

Contents

4.1 Introduction

The IForest [7] used in the EMDG-FL approach, presented in section 3, exhibits significant sensitivity to parameter settings refers to how its effectiveness in detecting anomalies can vary significantly based on the chosen values for its parameters, such as the number of trees or the sample size, impacting the accuracy of malicious update detection. In addressing this within the new ERD-FL approach, the focus is not on modifying the IForest itself but on compensating for its limitations in the context of EMDG-FL. This chapter introduces ERD-FL, an innovative scalable defense mechanism for FL systems in IoT networks. It signifies the first case in the field where entropy information and an adaptive threshold are employed, aiming to enhance detection robustness and improve accuracy against Label-flipping attacks. To validate the efficacy of ERD-FL, we carried out thorough simulations in Python, using different datasets to assess the system's performance. These include image classification datasets such as MNIST [2] and Fashion-MNIST [3], as well as the text-based IMDB dataset [5], enabling us to evaluate our approach across different data types and client numbers. Our analysis extends to various client configurations to examine scalability and operational effectiveness thoroughly. Furthermore, ERD-FL's performance was benchmarked against three other advanced defense strategies in the domain EMDG-FL [112], D2MIF [74], and FoolsGold [69]. Our comparison focuses on several critical metrics, including the ACC, ASR, LR, and CPU aggregation run-time shown in section 2.11. Through this comparative study, we aim to demonstrate ERD-FL's superior adaptability and efficiency, underscoring its potential as a scalable and robust solution against Label-flipping attacks in diverse FL environments.

In the comprehensive review outlined in chapter 2, we identified that while numerous detection methods for poisoning attacks in FL system have been introduced, they predominantly suffer from scalability issues, particularly when considering the deployment across varying numbers of clients and when applied to different types of datasets, notably image classification datasets. The methods highlighted in [74] and [69], for example, are resource-intensive and not sufficiently flexible or efficient for scaling up to larger, more diverse federated networks. The FoolsGold [69] incurs significant computational costs in analyzing client update similarities, a process that becomes increasingly complex as the

number of participants grows. Similarly, the D2MIF [74] is hindered by the need for constant threshold adjustments, complicating its application to varied and dynamic client environments. Even more, the reliance on RL [4] for policy optimization, while innovative, demands extensive trial-and-error interactions, limiting its practicality in scenarios where rapid adaptation to new and diverse datasets is required.

The EMDG-FL approach [112], despite its novel use of the IF algorithm for scoring and the GA [1] algorithm for threshold optimization, remains untested across a spectrum of client numbers and has not been validated with different types of image classification datasets. This gap underscores a critical scalability challenge: the existing methods lack comprehensive evaluation of how they perform when scaled across different sizes of federated networks and applied to various categories of data, especially image classification datasets where nuances in data can significantly impact performance. To address these scalability and flexibility concerns head-on, our proposed solution focuses on leveraging entropy information to streamline the detection of malicious models against Label-flipping attacks, aiming to drastically lower computational overhead. This approach not only promises enhanced performance in identifying threats but also ensures adaptability and efficiency when tested against an expanded array of client numbers and both image classification and text-based datasets. Through this, we aim to deliver a solution that is not only robust in handling the complexities of varied federated environments but also capable of accommodating the specific challenges posed by different type of data, marking a significant step forward in scalable and effective FL security. The following sections will offer an elaborate description of our novel approach. Subsequent to this, we will present the results obtained from our simulations.

4.2 ERD-FL proposed model

Our defense approach is designed to detect and mitigate Label-flipping attacks in the FL system by analyzing the entropy of the model updates provided by each client denoted as H(X). This serves as a metric of uncertainty and randomness within a dataset. Its computation is defined by the following formula:

$$H(X) = -\sum_{i=1}^{n} P(X = x_i) \log P(X = x_i) \tag{4.2.1}$$

In this equation, X represents a random variable characterizing model updates, while x_i signifies the potential values X can assume. Meanwhile, $\mathrm{P}(X = x_i)$ reflects the likelihood of X taking the value x_i. Essentially, entropy gauges the degree of unpredictability within the data. The term $\log P(X = x_i)$ quantifies the information content associated with each outcome x_i. Higher probability equates to lower information content. The presence of the negative sign inverts the scale, linking increased entropy to heightened uncertainty. This concept grants entropy its apt role as a metric for assessing randomness. In Figure 4.1, we present a detailed flowchart of our new approach. $C1$ to Cn represent normal clients, while $C3$ is the malicious client with poisoned data. First, the clients train local models received from the server using their own local data. Second, the clients send their new locally trained models to the server for entropy analysis to detect malicious models. Third, the entropy of suspicious clients' updates is analyzed and flagged as poisoned, they are rejected from aggregation. Fourth, the server aggregates updates from non-malicious clients after passing them through the malicious model detection process based on entropy. Fifth, The server sends the new global model aggregated to clients. Sixth, clients exceeding the entropy threshold twice are eliminated as malicious.

The steps of our approach, described in Algorithm 3, are as follows: Initially, the server computes the global model's initial accuracy. First, the server sends the global model to all clients. During the training phase, the clients train on their local models using their own data. As they train, clients' local model updates, denoted as ΔW, are collected. Simultaneously, clients calculate their pre-accuracy and upload their updated models to the server. Second, the server compares its global model's initial accuracy and the pre-accuracy of each local model. Third, it checks if the received client's pre-accuracy is less than the server's initial accuracy. Any client whose pre-accuracy falls below the initial accuracy of the server is marked as suspicious. If this is the case, the server will activate the entropy model detection process.

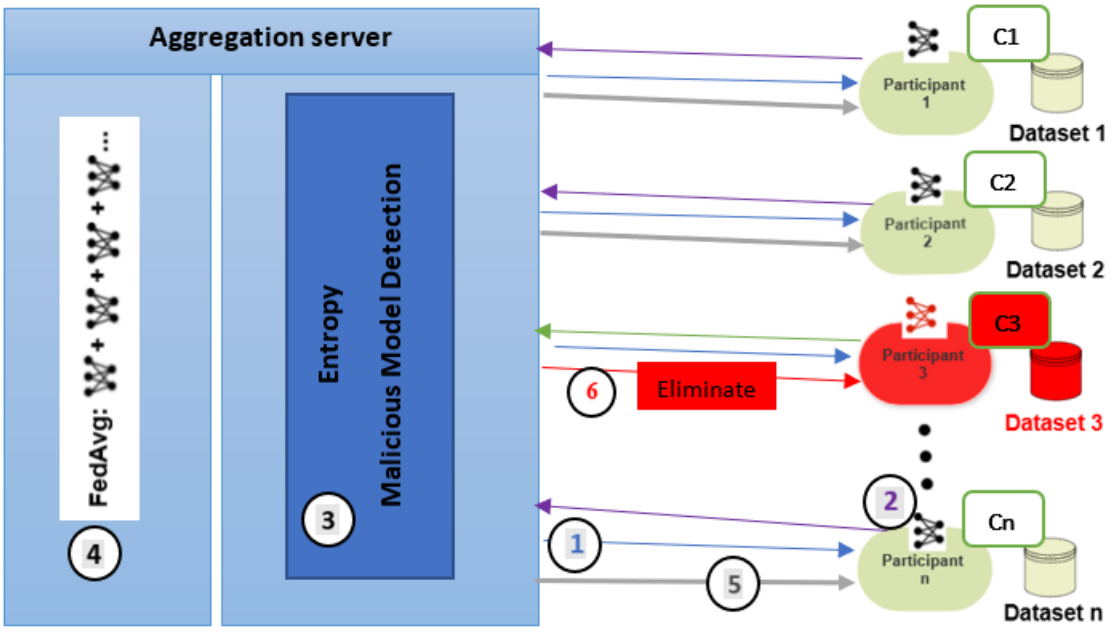


Figure 4.1: ERD-FL approach against Label-flipping attacks

For malicious model detection, the algorithm delves deeper into the updates from these suspicious clients. It determines the entropy of their weight updates using the normalized histogram of weight changes in ΔW as presented in Algorithm 3 at line 10. The average entropy of all suspicious clients' updates is set as an initial threshold γ and it is calculated in Algorithm 3 at line 11. Suspicious clients with an entropy value greater than this threshold are considered potentially malicious. These are not only rejected from the current FL aggregation but are also flagged for closer monitoring. The threshold is dynamic and undergoes re-calibration in subsequent local training rounds. It is adjusted based on the average entropy of the suspicious clients' updates. For heightened security, clients flagged as malicious are subject to further examination. If a client exceeds the current entropy threshold twice, it's permanently excluded from the FL process.

Finally, the local models that passed the entropy model detection checks are aggregated to update the server global model, which is then ready for another FL iteration. The process continues until no active clients remain, marking the end of the FL cycle. The benefit of updating the threshold is that it can improve the efficiency of the malicious model detection mechanism. If the threshold is not updated, then the threshold may become too high or too low over time. For example, If a new client joins the FL process and turns out to be malicious, the entropy of the clients' weight updates may increase, the threshold becomes too high, and the malicious client may not be flagged. By updating the threshold after each local training round, we can ensure that the threshold is always accurate and effective at flagging potentially malicious clients. The initial and updated thresholds are determined using the average entropy of weight updates from suspicious clients. This method prioritizes flagging potential malicious clients. For example, if 10 clients are marked suspicious in an FL process with 100 clients, the initial threshold is based on their entropy. In a subsequent round, if another 10 clients are flagged, the threshold is adjusted based on the combined 20 suspicious clients. In this way, the threshold remains adept at detecting malicious clients despite varying average entropy across all clients in different rounds. Our proposed method's novelty lies in utilizing entropy information as a defense mechanism. By leveraging entropy analysis, we can detect and filter out potentially poisoned data, thereby improving the robustness of FL against Label-flipping attacks. This approach provides a complementary defense strategy to existing methods and enhances the integrity and reliability of the FL process. In the next section, we will present the experimental setup and results to evaluate the effectiveness of our proposed defense mechanism.

Algorithm 3: Defense mechanism using entropy information.

// Initialize Global Model:

The server calculates the global model's initial-accuracy.
The server distributes the global model to all clients for local training.

// Local Training Phase:

The server receives clients' local model updates (ΔW).
The server evaluates the pre-training accuracy of each local model.

// Initial Aggregation Check:

if *pre-accuracy* > *initial-accuracy* **then**
    Aggregate the acceptable local models (those that passed the pre-accuracy check).

// Anomaly detection and entropy calculation:

else
    for *each client that could be suspicious* **do**
        Calculate the entropy (H) of their model weights update (ΔW):
        $H(\Delta W) = -\sum p(w) * \log p(w)$
            // p(w) is derived by normalizing the histogram of weight updates in ΔW.

// Establish Entropy Threshold (γ):

    Calculate the average entropy $H(\Delta W)$ across all suspicious clients to set an initial entropy threshold (γ): $\gamma = (1 / M) * \sum_{i=1}^{M} H(\Delta W_i)$
      // M is the number of suspicious clients and $H(\Delta W_i)$ is the calculated entropy of the i-th client's update.

// Client Evaluation:

    for *each suspicious client* **do**
        Compute $H(\Delta W)$ for the client's update.
        if $H(\Delta W) > \gamma$ **then**
            Flag the client as potentially malicious.
            Exclude the client updates from the current aggregation.
            Add the client to a watch list for further observation.

// Threshold Adaptation:

    After each local training round, reassess the average entropy from the remaining client contributions and adjust γ accordingly.

// Client Monitoring and Elimination:

    Continuously monitor the clients on the watchlist.
    If any client's entropy value exceeds γ consistently in subsequent rounds, indicating persistent questionable behavior, remove the client from further FL processes.

// Model Aggregation:

    Combine the updates from all non-eliminated clients to refine the global model.

// Share the new global model to all clients:

    Send the enhanced global model back to all remaining clients for additional rounds of local training.
    Repeat the process starting from step 2 for each new round of FL.

4.3 Simulation results analysis

Our simulations were conducted following the same experimental setup described in section 3.3 to examine the effects of Label-flipping attacks, as outlined in section 2.9.1, on an FL system with varying client numbers (30, 50, 100, 150). We employed the D2MIF [74], FoolsGold [69] and EMDG-FL [112] implemented using our FLSecLAB framework presented in section 2.7, as benchmarks to gauge the efficacy of our innovative approach evaluating them based on ACC, ASR, LR, and CPU aggregation runtime. We utilized the same three datasets mentioned in section 2.12 and their respective models discussed in section 3.3. These included the MNIST [2] and Fashion-MNIST [3] datasets for image classification and the IMDB [5] dataset for text-based sentiment analysis. Adding a text dataset to our analysis broadens the study, offering insights into the method's adaptability to different data types and enhancing its generalizability. During this simulation, we assign a random value to the attackers' proportion, capping it at a maximum of 20% of the clients participating in the FL process.

In the first experiment, we evaluated our ERD-FL approach in terms of the ACC performance metric that is explained in section 2.11. A high ACC indicates effective differentiation between honest and malicious models. Our study compares the efficacy of ERD-FL with EMDG-FL [112], D2MIF [74], and FoolsGold [69] under Label-flipping attacks for different client counts (30, 50, 100, 150). Figure 4.2 illustrates that ERD-FL consistently outperforms in accuracy across multiple datasets, including MNIST, Fashion-MNIST, and IMDB. Notably, EMDG-FL is the closest competitor, especially under the MNIST and Fashion-MNIST datasets, except for the case with 30 clients. ERD-FL showcases its robust scalability and effectiveness, transitioning seamlessly from image-based to text-based datasets. This versatility and performance superiority are visually summarized in Figure 4.2, highlighting ERD-FL's adaptability to varied client numbers and dataset types.

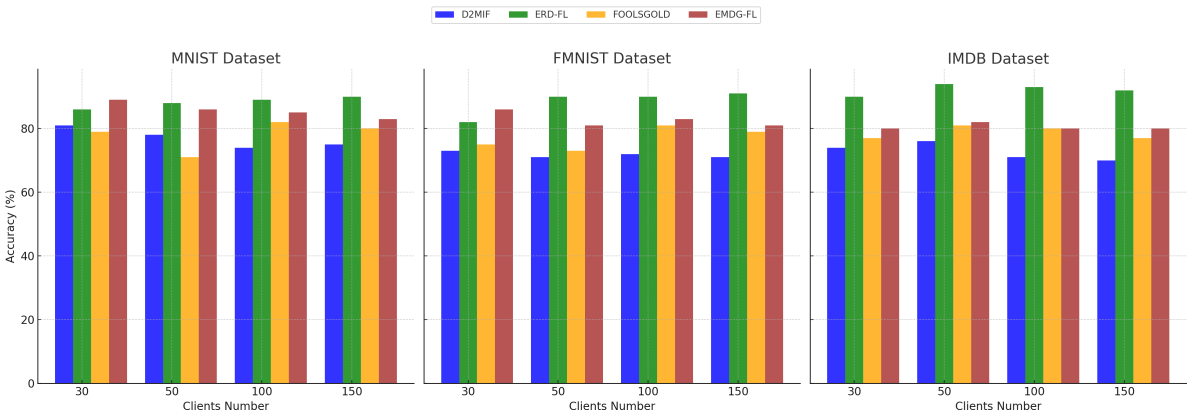


Figure 4.2: ACC of the four approaches using MNIST, Fashion-MNIST, and IMDB datasets against Label-flipping attacks with different client numbers

In the second experiment, we evaluated our ERD-FL approach in terms of the ASR performance metric as presented in section 2.11, which refers to the degree of correctness or the ability of the model to make correct predictions on a given set of data from a distributed set of clients. The goal is to compare the performance of our ERD-FL approach with EMDG-FL [112], D2MIF [74], and FoolsGold [69] in the presence of Label-flipping attacks on MNIST [2], Fashion-MNIST [3], and IMDB [5] datasets with 30 clients. As shown in Figure 4.3, ERD-FL demonstrates the lowest ASR among all the studied approaches due to its superior detection sensitivity. This enables ERD-FL to more accurately filter poisoned updates, thereby maintaining higher global model accuracy during Label-flipping attacks.

In the third experiment, we evaluated the ERD-FL approach regarding LR performance metrics using MNIST, Fashion-MNIST, and IMDB datasets with varying different iterations' numbers and 30 clients. The LR represents the rate at which the model's accuracy decreases over time as more clients are added to the FL system.

Figure 4.4 illustrates the LR computation for our ERD-FL, alongside EMDG-FL [112], D2MIF

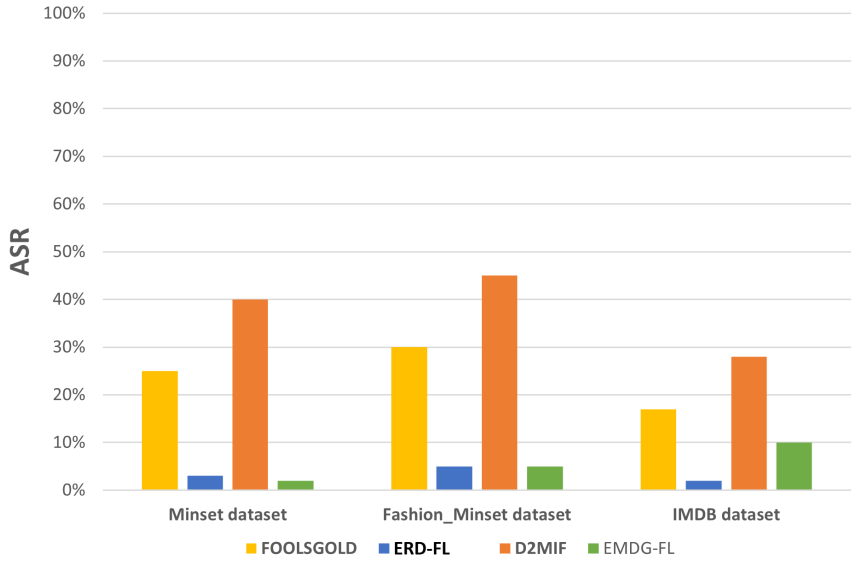


Figure 4.3: ASR of the four approaches using three different datasets against Label-flipping attacks

[74], and FoolsGold [69] approaches, in the presence of a Label-flipping attack on the MNIST [2] dataset with varying numbers of iterations. The LR exhibits a noticeable decrease from the initial iteration for all approaches. The ERD-FL and EMDG-FL approaches demonstrate a sharp drop in LR within the first 14 iterations, maintaining a consistently low level thereafter. In contrast, D2MIF and FoolsGold require up to thirty iterations to achieve a significant LR reduction. Importantly, this extended iteration period for D2MIF and FoolsGold corresponds to a higher communication cost between clients and the central server.

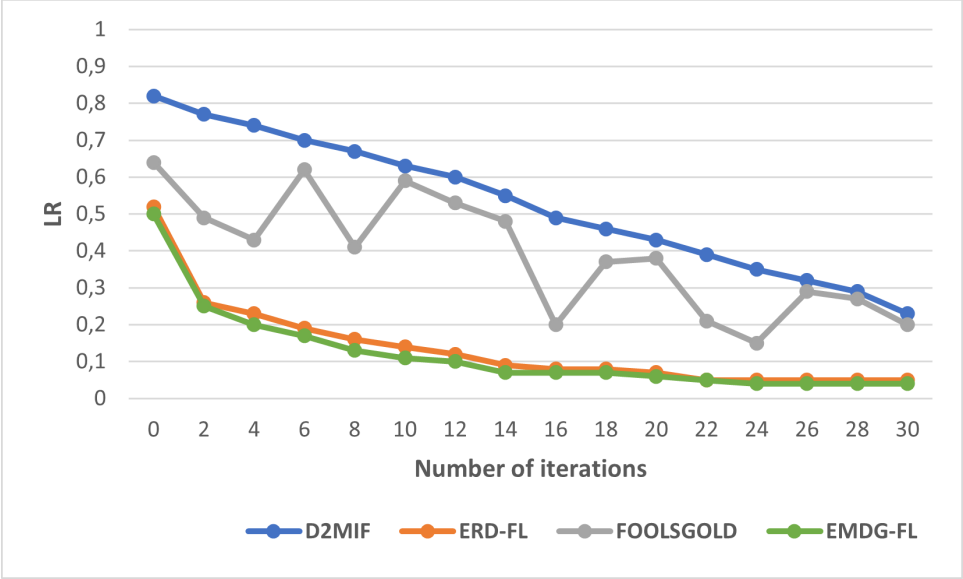


Figure 4.4: LR of four approaches against Label-flipping attacks using MNIST dataset with different iterations

Figure 4.5 presents a comparison of the LR among our proposed ERD-FL, EMDG-FL [112], D2MIF [74], and FoolsGold [69] approaches under the influence of Label-flipping attack using the Fashion-

MNIST [3] dataset with varying numbers of iterations. The results clearly show the superior performance of the ERD-FL and EMDG-FL approaches, characterized by consistently lower LR values compared to D2MIF and FoolsGold. Notably, ERD-FL and EMDG-FL achieve a significantly low LR after 20 iterations, while D2MIF and FoolsGold require up to 30 iterations to reach a comparable low LR, which gradually increases over time.

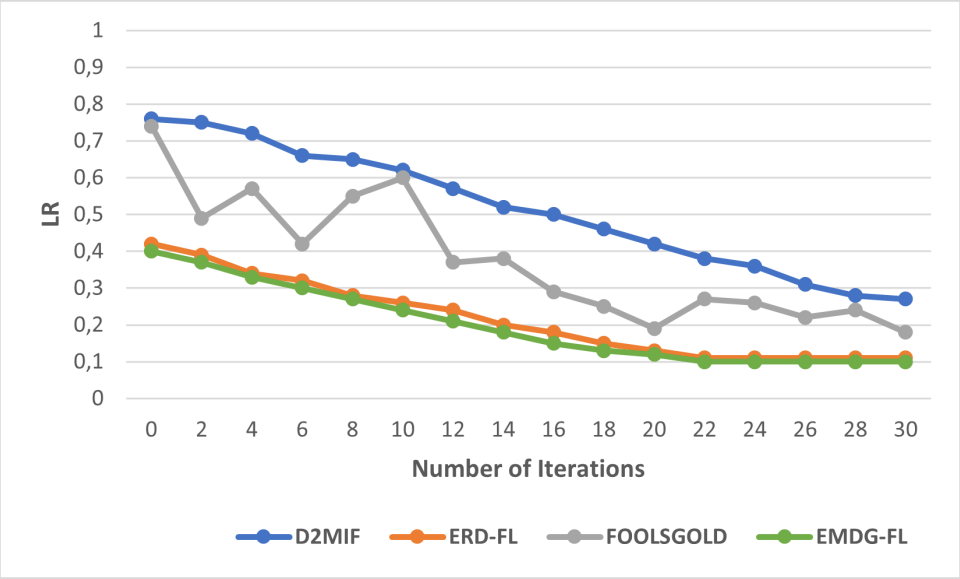


Figure 4.5: LR of four approaches against Label-flipping attacks using Fashion-MNIST dataset with different iterations

Figure 4.6 presents a comparison of the LR among our newly proposed ERD-FL, EMDG-FL, D2MIF, and FoolsGold approaches under Label-flipping attacks using the IMDB dataset with varying numbers of iterations. The results demonstrate the superiority of the ERD-FL approach, characterized by consistently lower LR values compared to EMDG-FL, D2MIF and FoolsGold. Particularly noteworthy is that the ERD-FL approach achieves a remarkably low LR from the first iteration, whereas EMDG-FL, D2MIF and FoolsGold require up to 28 iterations to reach a comparable low LR, which gradually increases over time.

In the fourth experiment, we evaluated our approach in terms of the average CPU aggregation run-time for varying client numbers (30, 50, 100, 150) and compared it with EMDG-FL, D2MIF and FoolsGold approaches using the MNIST , Fashion-MNIST and IMDB datasets. As we show in Table 4.1, the server run-time overhead per iteration, counted in seconds, the result of our ERD-FL approach is lower than the three studied works from the literature. Furthermore, considering the remarkable effectiveness and scalability of our method in countering targeted attacks, the minimal run-time overhead becomes a highly valuable investment.

Table 4.1: Average CPU aggregation run-time per iteration of the server using MNIST, Fashion-MNIST, and IMDB datasets

	CPU run-time per iteration			
	ERD-FL	***EMDG-FL***	***D2MIF***	***FOOLSGOLD***
MNIST	0.01	0.02	0.06	0.07
Fashion-MNIST	0.02	0.05	0.10	0.08
IMDB	0.08	0.15	0.22	0.17

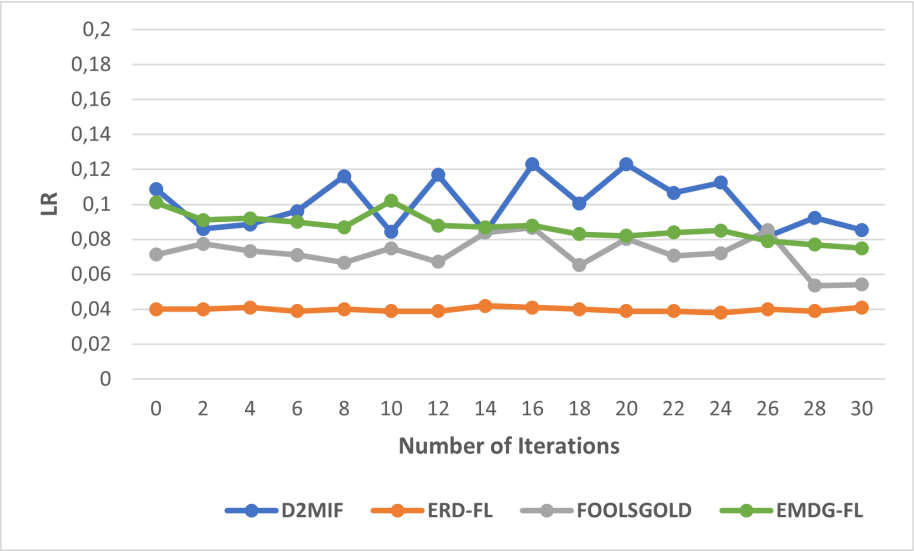


Figure 4.6: LR of four approaches against Label-flipping attacks using IMDB dataset with different iterations

4.4 Conclusion

In this chapter, we have introduced ERD-FL, a novel defense approach designed to enhance the security of FL system within IoT networks against Label-flipping attacks. Our approach stands out as the first to utilize entropy information for detecting such attacks, establishing a novel contribution to the field of FL security. By integrating an entropy information analysis of entropy in model updates, ERD-FL not only identifies but also mitigates the impact of poisoned clients, thereby significantly bolstering the integrity and resilience of FL systems. The effectiveness of ERD-FL was thoroughly evaluated through extensive simulations using a wide range of datasets, including MNIST, Fashion-MNIST for image classification, and IMDB for text-based, covering various scenarios. This comprehensive evaluation, which extended to different client numbers, provided a solid foundation for assessing the scalability and operational effectiveness of our proposed solution. Our findings revealed that ERD-FL exhibits superior adaptability and efficiency, outperforming other defense approaches presented in section 2.10: EMDG-FL, D2MIF, and FoolsGold, across several critical performance metrics. These metrics include ACC, ASR, LR, and CPU aggregation run-time, thereby underscoring its potential as a scalable and robust solution against Label-flipping attacks in diverse FL environments. Looking ahead, in the next chapter, we present a new defense concept for the FL system based on multi-layer protection. This concept addresses more complex scenarios with multiple simultaneous attacks and a larger number of clients.

Chapter 5

Multi-layer malicious model detection for federated learning in IoT networks

Contents

5.1 Introduction

While the ERD-FL approach leverages entropy for enhanced detection of malicious clients in FL, its dependency on historical entropy data may not adapt swiftly to sudden or novel attack strategies, potentially affecting its effectiveness. Future investigations should focus on enhancing the approach's adaptability and responsiveness to diverse and evolving attack patterns, ensuring robust defense across all FL scenarios. In this chapter, we explore a novel and advanced M3D-FL approach, adeptly tackling complex FL scenarios with a multi-layered defense against both Label-flipping and Backdoor attacks. The approach commences with the first layer, where the LOF [6] algorithm calculates MS for FL clients, facilitating their initial exclusion from the aggregation process. Subsequently, the second layer intensifies scrutiny using the MAD [8] algorithm for precise outlier detection, aimed at permanently removing these flagged clients, enhancing the system's resilience in multifaceted environments. By applying the M3D-FL approach within such intricate and challenging environments, demonstrated through extensive evaluations of datasets: CIFAR10 [9], MNIST [2], and Fashion-MNIST [3]. We showcase its efficacy in notably enhancing key metrics like ACC and CPU aggregation runtime, when compared to five other studies discussed in section 2.10: [74], [69], EMDG-FL [112], DifFense [75], and FedAvg [14] in the literature, thereby bolstering the resilience of FL system against a diverse range of threats in such intricate scenarios.

Below, we present our innovative approach. Following this, we present the results obtained from our simulations.

5.2 M3D-FL proposed model

Our novel malicious model detection process involves pre-aggregating received local models to calculate a *pre-accuracy*. If this *pre-accuracy* is lower than the initialization model's accuracy, the server will

execute our malicious model detection using the LOF [6] algorithm to calculate an MS for each local model uploaded by clients. Models with an MS below the set threshold are considered normal and are aggregated formally. However, any malicious models uploaded by clients are not used for formal model aggregation, and those clients are marked as sensitive. These clients will not be removed from the system immediately, but if they are marked as sensitive more than twice, the MAD [8] outlier detection method will be invoked, which will be explained below.

The MAD is a robust statistical method used to detect outliers in a dataset. In the context of FL, MAD can be used as a defense mechanism to detect and remove malicious or faulty updates from participating clients. One of the main advantages of using MAD in FL defense is its ability to handle non-normal distributions and outliers. This is particularly important in FL, where the data distribution across clients can be highly heterogeneous and non-normal. By using MAD, we can identify updates that deviate significantly from the median, which can help improve the overall accuracy and reliability of the FL model. Another advantage of using MAD in FL defense is its simplicity and computational efficiency. MAD can be easily computed using basic arithmetic operations and does not require complex modeling or training procedures. This makes it a practical, robust, and scalable defense mechanism for FL system [8].

To use MAD [8] for outlier detection in our approach, we first calculate the median of the data. Then, we calculate the absolute deviation of each data point from the median and take the median of these absolute deviations, which gives us the MAD. Next, we define a threshold for outlier detection. A common threshold is to define an outlier as any data point that is more than a certain number, named (C_MAD), of MADs away from the median. A common value for this constant factor (C_MAD) is three, which means that any data point more than three MADs away from the median is considered an outlier. To summarise, the steps for outlier detection using MAD are as follows: Let $X = \{x_1, x_2, ..., x_n\}$ be the dataset of n observations.

- **Step 1:** Calculate the median of X:
  Median(X) = median$(x_1, x_2, ..., x_n)$
- **Step 2:** Calculate the absolute deviation of each data point from the median:
  MADi = $|xi - Median(X)|$, where: $(i = 1, 2, ..., n)$
- **Step 3:** Calculate the median of the absolute deviations:
  MAD = median$(MAD_1, MAD_2, ..., MAD_n)$
- **Step 4:** Define a threshold (T) for outlier detection:
  T = (C_MAD$*MAD$)
- **Step 5:** Identify any data points that are above the threshold as outliers:
  Outliers = $\{x_i || x_i - Median(X)| > threshold\}$

To apply the MAD [8] outlier detection method, we introduce the notion of "suspiciousness" with a designated number called L. The MAD requires a larger number of clients to become more reliable and better represent the suspicious characteristics, enabling us to identify outliers more accurately. Our M3D-FL model is designed for detecting malicious models in the FL system. It utilizes the LOF algorithm [6] to calculate the MS and identify anomalies in each locally uploaded model by clients.

The LOF [6] calculates an outlier score based on the local density of local clients' models. The main steps involved in this algorithm are summarised as the following:

- Let $M = \{m_1, m_2, \ldots, m_n\}$ is the set of n local models uploaded by clients.
- For each local client's model (m_i), we calculates the K-Nearest Neighbours (KNN) [113] of m_i and their reachability distances using the Euclidean distance.
- We calculate the reachability distance (rd) of m_i toward m_j (rd) as follow:
  $rd(m_i, m_j)$ = max(Kd(m_i), d(m_i, m_j))

where Kd(m_i) represents the distance to the KNN [113] of a given local model, d(m_i, m_j) is the distance between m_i and m_j, and rd is calculated based on the maximum of Kd and the Euclidean distance to another model. The rd of m_i measures how far it is from its nearest neighbors.

- Let $NK(m_i)$ is the set of KNN of m_i. The LOF [6] calculates the Local Reachability Density (LRD) of each m_i as in formula (6.2.3), which is based on the inverse of the average reachability distances from the NK(m_i):

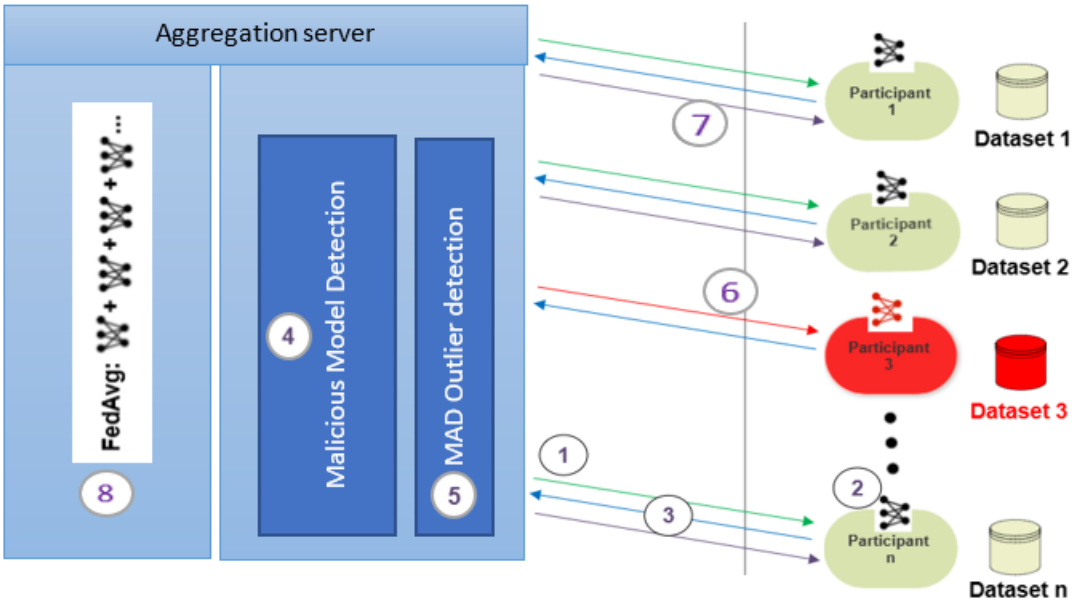
$$\mathrm{LRD}(m_i) = \frac{1}{\sum_{m_j \in \mathrm{NK}(m_i)} \mathrm{rd}(m_i, m_j)/|\mathrm{NK}(m_i)|} \tag{5.2.1}$$

- The NK(m_i) with lower LRD values are indicative of more isolated or anomalous models within the dataset.
- The LOF [6] calculates the score as the ratio of the average LRD of the KNN [113] and the LRD of the current m_i. The higher LOF scores indicate a higher likelihood of being an outlier.
- Calculate LOF score for each mi:

$$\mathrm{LOF}(m_i) = \frac{\sum_{m_j \in \mathrm{NK}(m_i)} \mathrm{LRD}(m_j)}{|\mathrm{NK}(m_i)| \cdot \mathrm{LRD}(m_i)} \tag{5.2.2}$$

- The MS for m_i is:
  MS(m_i) = LOF(m_i)

In Figure 5.1, we present a detailed flow chart of our new approach, which will be further explained in Algorithm 4.



(1) The Server sends the global model to clients **(2)** Local model training by clients using own data **(3)** The client uploads own model to the server **(4)** The server detects malicious clients **(5)** MAD Outlier detection process **(6)** The server eliminates malicious clients **(7)** Local models updates aggregation **(8)** The Server sends the new global model aggregated to clients.

Figure 5.1: M3D-FL approach against Label-flipping and Backdoor attacks

The process of the M3D-FL method is structured as follows:

1. The server distributes the global model to all clients.
2. clients train their local models using their own data.
3. clients upload their trained models to the server.
4. The server identifies malicious clients by comparing the MS of each model with a predefined threshold, T, determined in step 4 of MAD outlier detection algorithm.
5. If a client is labeled as malicious twice, it is labeled as suspicious. The goal is to reach a sufficient number of suspicious clients, noted L, to invoke MAD.
6. If the MAD result, applied to the L suspicious clients, indicates an outlier, the client is removed. Otherwise, the degree of maliciousness in the list is reduced by one, and the algorithm returns to the malicious model detection process.
7. If a client is not deemed malicious, the server aggregates their model with other normal clients using the FedAvg [14] algorithm.
8. After aggregating the local models, the server saves the new global model and sends it back to the clients to update their local model's parameters.

In summary, the M3D-FL approach incorporates various steps to detect and mitigate malicious behavior in an FL scenario. By iteratively evaluating clients' models and leveraging MAD [8], it maintains the integrity of the collaborative learning process while updating and distributing the global model to improve overall performance. The M3D-FL process involves multiple steps to ensure the integrity and reliability of the local models submitted by clients. These steps are explained in algorithm 4. The main objective of these steps is to prevent any malicious actors from compromising the FL process.

5.3 Simulation results analysis

We conducted our simulations following the same experimental setup described in section 3.3 to examine the impact of Backdoor and Label-flipping attacks presented in 2.9 on an FL system involving different numbers of clients (50, 100, 150, 200). We employed the D2MIF [74] and the FoolsGold [69] implemented using our FLSecLAB framework presented in Section 2.7 as benchmarks to gauge the efficacy of our innovative approach. We used three datasets described in section 2.12 and their corresponding models discussed in section 3.3. These included the CIFAR10 [9], MNIST [2], and Fashion-MNIST [3]. Throughout this simulation, the ratio of attackers to participating clients in the FL process is varied randomly, constrained to not exceed 20% of the total clients.

It's important to note that the accuracy results for the DifFense [75] approach are not implemented in our thesis, except with 50 clients in the presence of a Backdoor attack, where we implemented our new approach and the other studied approaches. We took the accuracy results for the DifFense approach as they are reported in their paper.

In the first experiment, we assess our M3D-FL approach in terms of the ACC performance metric which explained in section 2.11, compared to other studied approaches presented in section 2.10: FedAvg [14], EMDG-FL [112] , D2MIF [74], FoolsGold [69] and DifFense [75] in the presence of Backdoor attack and using three different datasets with varying numbers of clients. From Figure 5.2, it is evident that M3D-FL consistently achieves the highest and the most stable accuracy across different numbers of clients and outperforms the studied approaches regarding the ACC when using the CIFAR10 dataset. The ACC of M3D-FL from 93% to 90%. EMDG-FL, D2MIF, and FoolsGold also demonstrate competitive performance, with ACC from 88% to 86%, 85% to 81%, and 86% to 81%, respectively. The ACC of DifFense is calculated for 50 clients only and it has a competitive performance equal to 89 %.

Algorithm 4: M3D-FL process

Require: L

// Number of suspicious clients to be reached for MAD algorithm

$m\_list = [0] * N$ // List to track MSs per client

$suspicious = []$ // List to track suspicious clients

Receive the local model set uploaded by clients
Aggregate the local models in the set together
Calculate local models' pre-accuracy
if *pre-accuracy* > *initial model's accuracy* **then**
    initial model's accuracy = pre-accuracy
    Aggregate the local models in the set // Only if new model improves accuracy
else
    Calculate an $MS(i)$ for each local model m_i using the LOF algorithm
    if $MS(i) > T$ **then**
        Mark the model m_i as malicious and reject it from aggregation
        $m\_list[i] = m\_list[i] + 1$ // Marking the client who submitted malicious model as sensitive
        if $m\_list[i] \geq 2$ **then**
            $suspicious$.append(i) // Add suspicious client to the list
            if *len*($suspicious$) $\geq L$ **then**
                Invoke the MAD detection process
                if *MAD result is an outlier* **then**
                    Remove this client
                else
                    $m\_list[i] = m\_list[i] - 1$
    else
        Aggregate this model with other normal models using the FedAvg algorithm

Figure 5.2: ACC using CIFAR10 dataset under Backdoor attack with different clients number

Next, we evaluate our M3D-FL with the ACC compared to the other approaches: FedAvg [14], EMDG-FL [112] , D2MIF [74], FoolsGold [69] and DifFense [75], against Backdoor attack and using the Fashion-MNIST [3] with wide numbers of client. Figure 5.3 shows the outperforming ACC of M3D-FL compared to the studied approaches. In this scenario, M3D-FL consistently achieves the highest and most stable accuracy across different numbers of clients, ranging from 91% to 88%. EMDG-FL and FoolsGold also demonstrate competitive performance, with ACC from 89% to 86% and from 86% to 81%, respectively. D2MIF achieves ACC from 86% to 79%, while FedAvg consistently has lower ACC, from 76% to 60%.

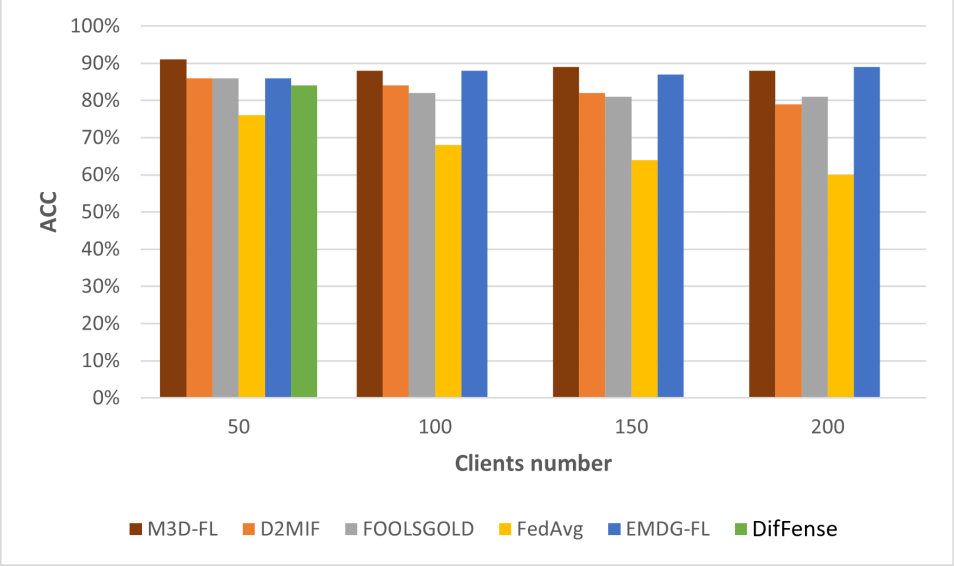


Figure 5.3: ACC using Fashion-MNIST dataset under Backdoor attack with different clients number

We assess our M3D-FL approach's ACC compared with FedAvg [14], EMDG-FL [112], D2MIF [74] and FoolsGold [69] under a Backdoor attack using the MNIST dataset [2] with varying numbers of clients. From Figure 5.4, it can be observed that M3D-FL consistently achieves the highest and the most stable accuracy across different numbers of clients. Its ACC from 92% to 87%. EMDG-FL

and FoolsGold also show competitive performance, with ACC from 88% to 87% and 83% to 79%, respectively. D2MIF and FedAvg have slightly lower accuracies, from 82% to 77% and 78% to 70%, respectively.

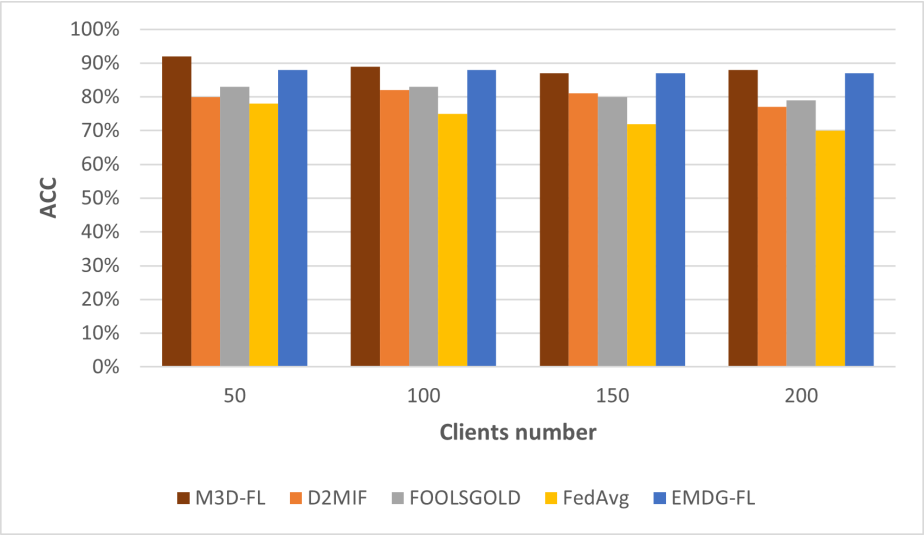


Figure 5.4: ACC using MNIST dataset under Backdoor attack with different clients number

In the second experiment, we evaluate the ACC of our M3D-FL approach against other studied approaches: FedAvg [14], EMDG-FL [112], D2MIF [74] and FoolsGold [69] from the literature under a Label-flipping attack, using three different datasets and different numbers of client. From Figure 5.5, it is evident that M3D-FL outperforms the studied approaches with the highest and the most stable ACC using the CIFAR10 [9] dataset with varying numbers of clients. In this scenario, M3D-FL consistently achieves high accuracy across different numbers of clients, from 91% to 87%. EMDG-FL [112] and FoolsGold [69] also demonstrate a competitive performance, with ACC from 89% to 85% and 89% to 79%, respectively. D2MIF [74] achieves ACC from 86% to 79%, while FedAvg [14] consistently has lower ACC, from 87% to 65%.

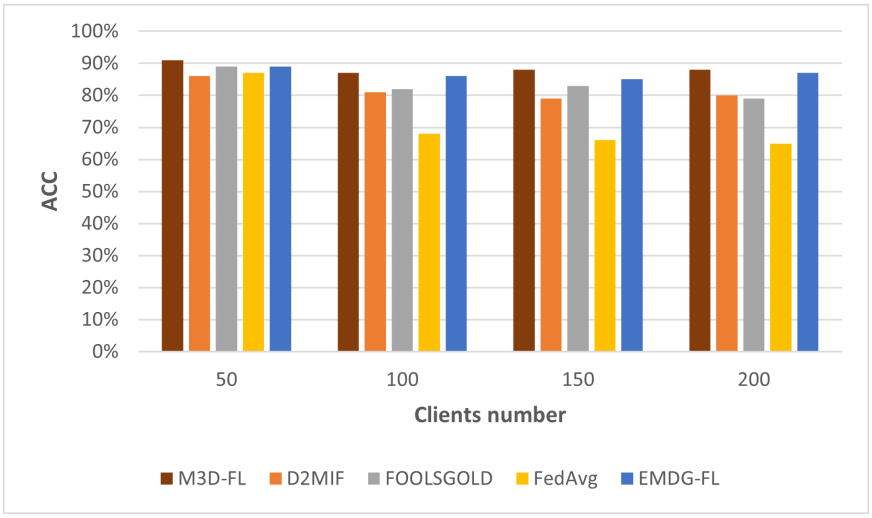


Figure 5.5: ACC using CIFAR10 dataset under Label-flipping attack with different clients number

We also evaluate the performance of our M3D-FL approach with ACC, contrasting it with existing studies' approaches against Label-flipping attack, utilizing the Fashion-MNIST [3] dataset. This

evaluation includes a comparison across different client numbers. Figure 5.6 illustrates the degree of ACC improvement our approach offers over other approaches. Figure 5.6 shows the enhancement level of ACC of M3D-FL approach compared to the studied approaches. M3D-FL achieves the highest and the most stable ACC across different numbers of clients, ranging from 91% to 88%. EMDG-FL, FoolsGold [69] and D2MIF [74] also demonstrate competitive performance, with ACC from 88% to 85% and 88% to 79% and 85% to 81%, respectively. FedAvg [14] consistently has lower accuracies, from 79% to 64%.

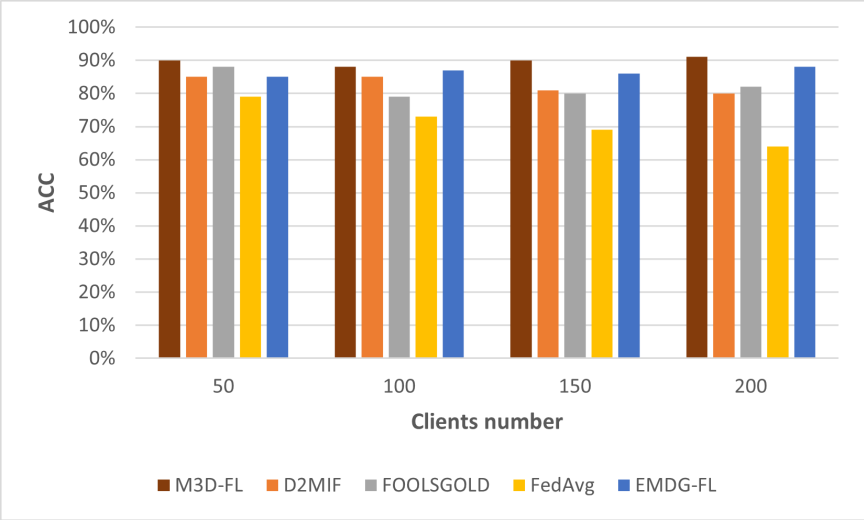


Figure 5.6: ACC using Fashion-MNIST dataset under Label-flipping attack with different clients number

Subsequently, we examine our M3D-FL approach in terms of the ACC, contrasting it with other studied approaches: FedAvg [14], EMDG-FL [112], D2MIF [74] and FoolsGold [69] under the Label-flipping attack, using the MNIST [2] dataset with different client numbers. Figure 5.7. M3D-FL maintains the highest and most stable levels of accuracy with differing numbers of clients, varing from 94% to 92%. EMDG-FL and FoolsGold also demonstrate competitive performance, with ACC ranging from 89% to 82% and 82% to 80%, respectively. D2MIF achieves ACC from 84% to 80%, while FedAvg consistently has lower ACC, from 75% to 65%.

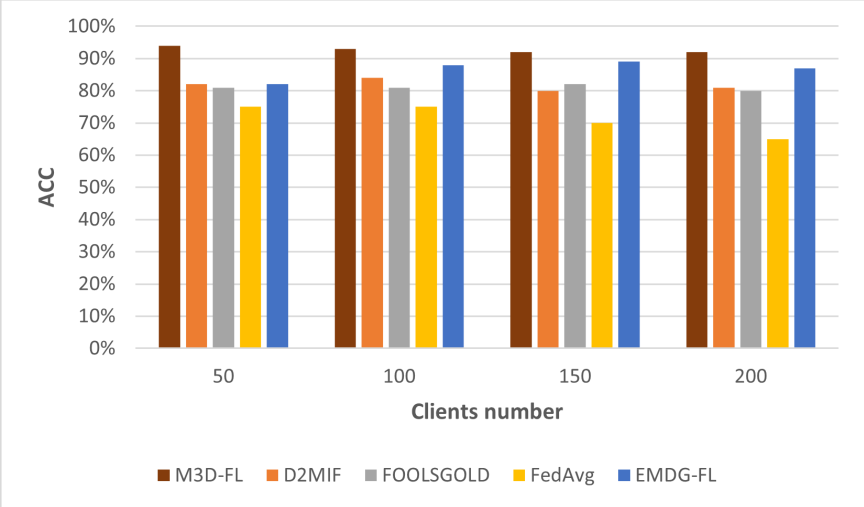


Figure 5.7: ACC using MNIST dataset under Label-flipping attack with different clients number

To validate the optimal selection of the threshold value for our methodology, the third experiment was designed to evaluate the effectiveness of different threshold levels in enhancing the ACC by applying three distinct threshold values for the MAD [8], utilizing the CIFAR10 [9] dataset. The results, depicted in Figure 5.8, indicate that a threshold value of 3 is optimal. Consequently, we integrated this value into our approach.

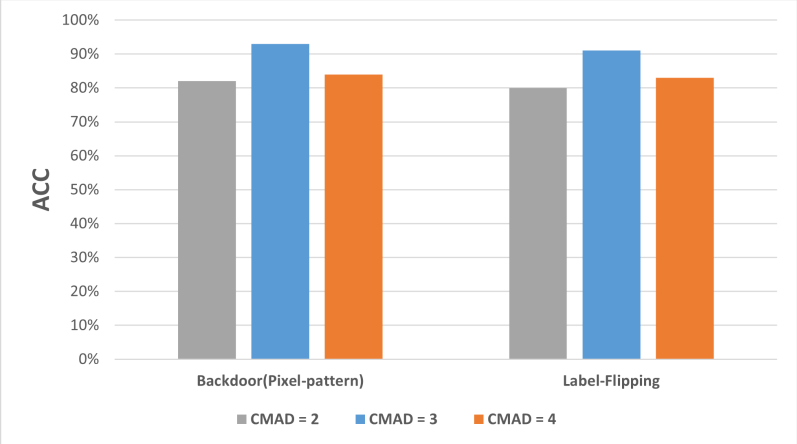


Figure 5.8: ACC of the three choices of C_MAD using CIFAR10 dataset

After that, we evaluated M3D-FL in terms of ACC performance metric while using three different threshold values of MAD [8] outlier detection using the MNIST [2] dataset. From Figure 5.9, we show that the most appropriate value equals 3, which we used in our M3D-FL approach.

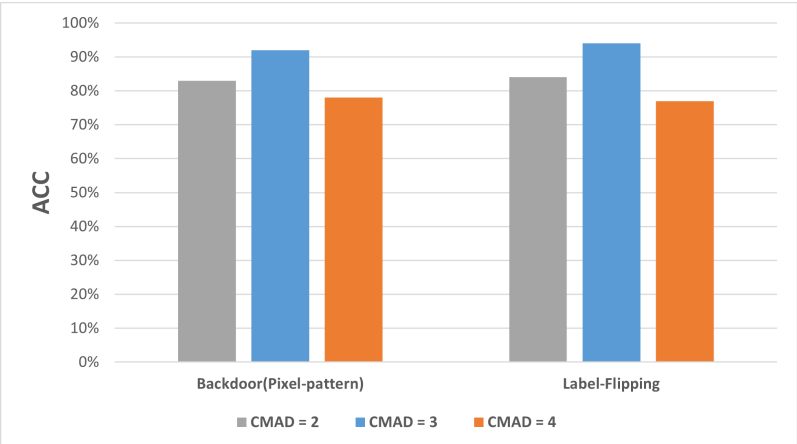


Figure 5.9: ACC of the three choices of C_MAD using MNIST dataset

We also evaluated M3D-FL in terms of ACC while using three different threshold values of MAD [8] using the Fashion-MNIST [3] dataset. From Figure 5.10, We demonstrate that the optimal value is equal to 3, which we employed in our novel approach.

The previous results indicate that M3D-FL is particularly effective in mitigating the impact of the Backdoor and Label-flipping scenarios in FL system using the MNIST, Fashion-MNIST, and CIFAR10 datasets.

In the fourth experiment, we evaluated M3D-FL in terms of the average CPU aggregation run-time and compared it with other approaches from the literature using the various datasets. As we show in Table 5.1, the CPU aggregation run-time overhead per iteration, counted by the server in seconds, the result of M3D-FL is lower than three other approaches using CIFAR10 [9] dataset, except for FedAvg

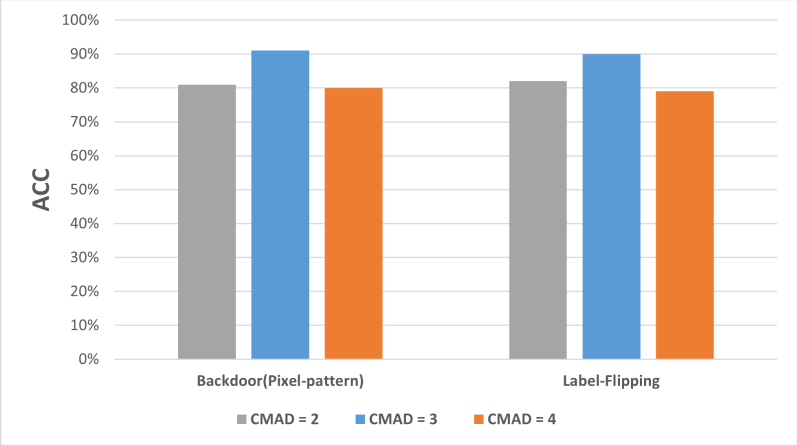


Figure 5.10: ACC of the three choices of C_MAD using Fashion-MNIST dataset

[14] where it primarily focuses on averaging updates and does not have a specific design to counter attacks. Additionally, given the impressive effectiveness of M3D-FL in combating targeted attacks, the minimal run-time overhead becomes a highly worthwhile investment.

Table 5.1: Average CPU aggregation run-time per iteration of the server using CIFAR10 dataset

	Average CPU run-time				
	M3D-FL	***EMDG-FL***	***D2MIF***	***FoolsGold***	***FedAvg***
Backdoor	0.200	0.290	0.534	0.294	0.197
Label-flipping	0.205	0.257	0.437	0.255	0.179

Table 5.2 shows the average CPU aggregation run-time for M3D-FL and other approaches from literature using the Fashion-MNIST [2] dataset. M3D-FL has the lowest CPU aggregation run-time compared to the other studied approaches.

Table 5.2: Average CPU aggregation run-time per iteration of the server using Fashion-MNIST dataset

	Average CPU run-time per iteration				
	M3D-FL	***EMDG-FL***	***D2MIF***	***FoolsGold***	***FedAvg***
Backdoor	0.016	0.028	0.079	0.205	0.007
Label-flipping	0.025	0.031	0.099	0.256	0.009

We show in Table 5.3, the average CPU aggregation run-time for M3D-FL and other approaches from literature using the MNIST [2] dataset. M3D-FL has the lowest CPU run-time compared to the other studied approaches.

Table 5.3: Average CPU aggregation run-time per iteration of the server using MNIST dataset

	Average CPU run-time per iteration				
	M3D-FL	***EMDG-FL***	***D2MIF***	***FoolsGold***	***FedAvg***
Backdoor	0.070	0.083	0.105	0.230	0.010
Label-flipping	0.086	0.092	0.090	0.234	0.060

5.4 Conclusion

We conclude that M3D-FL exhibits superior and more stable performance relative to other evaluated approaches in the literature, by implementing a multi-layered protection strategy. The utilization of the LOF and MAD outlier detection algorithms has proven to be more efficacious compared to the studied approaches, yielding better ACC metrics and acceptable CPU aggregation run-time. The outcomes achieved demonstrate a notable improvement and are considered satisfactory. In previous approaches, our discussion and analysis were primarily focused on environments where data is IID. However, the next chapter will expand our exploration into more complex FL scenarios that incorporate both IID and Non-IID data settings in the FL system. Here, we aim to address the additional challenges posed by Non-IID data, a scenario that is more representative of real-world applications, which were not tackled by our approaches discussed previously.

Chapter 6

Non-IID adaptive malicious model detection in federated learning in IoT networks

Contents

6.1 Introduction

The LOF method used in chapter 5 is effective for identifying malicious clients, yet it faces difficulties with Non-IID data in the FL system due to its inability to handle diverse data distributions. This limitation introduces vulnerabilities and affects its performance in high-dimensional data scenarios. In such contexts, traditional algorithms, including LOF, struggle to cope with the intricacies presented by a multitude of features, consequently diminishing their accuracy in detecting outliers. This issue underscores the need for advanced approaches to navigate the challenges of diverse and complex data landscapes within FL systems.

In this chapter, we present a novel server-client defense approach, named NAM2D-FL for identifying and removing malicious models in the FL system, to address the intricate challenges of both IID and Non-IID data environments. For the first time in our thesis, we use a dual-sided defense strategy, leveraging both server and client components to combat Label-flipping and Backdoor attacks. This approach is driven by the critical need to tackle the complexities associated with Non-IID data, a common issue in real-world applications not fully addressed by previous approaches. Our innovative use of server-client defenses seeks to significantly enhance the adaptability, computational efficiency, and performance of FL systems, offering robust protection against advanced attacks.

By implementing the NAM2D-FL approach in complex environments and rigorously testing it on datasets CIFAR10 [9], CIFAR10-Non-IID [9], MNIST [2], and MNIST-Non-IID [2]. We compare our approach with five existing methods discussed in section 2.10: Median [54], TMean [54], MKrum [55], and FoolsGold [69] from the literature in terms of ACC, ASR, Precision, Recall, and CPU run-time of each method. This clear evidence establishes NAM2D-FL as the top-performing solution for fortifying FL system against Label-flipping and Backdoor attacks in diverse scenarios. In the next sections, we present our innovative approach, and following that, we detail the outcomes of our simulations.

6.2 NAM2D-FL proposed model

Our novel malicious model detection process is encapsulated within Algorithm 5. This process entails pre-aggregating received local models to calculate a *pre-accuracy*. Based on this value, we initially discern the viability of each local update in enhancing the global model's performance. If a local model's pre-accuracy exceeds the current global model's accuracy, it is accepted for the server's aggregation; otherwise, the malicious model detection process is invoked outright.

Our model comprises of two phases: an adaptive LOF and an optimal GA [114] threshold. First, we adapt the traditional LOF algorithm [6], a choice informed by its reputation for adeptly identifying data point anomalies within a dataset. Second, select the optimal threshold and integrate it within the adaptive LOF scoring using the GA algorithm.

6.2.1 Phase one: Adaptive LOF

This phase introduces significant innovations to the traditional LOF method by adapting it to the context of FL, focusing on two main steps:

- **Contextual distance adaptation:** The contextual distance $d_{\text{contextual}}(x, y)$ between two model updates x and y is defined as:

$$d_{\text{contextual}}(x, y) = \sum_{i=1}^{n} w_i * |x_i - y_i|. \tag{6.2.1}$$

  Here, n represents the total number of parameters in the model updates x_i and y_i, respectively. w_i are weights that adapt based on the variance of parameters across updates, making the distance measure sensitive to the distribution of data (IID or Non-IID). The goal is to mitigate the impact of variance in parameter updates, ensuring malicious model updates are detected based on meaningful deviations rather than mere statistical outliers.

- **LRD adjustment and refined LOF scoring:**

  In refining the Local Reachability Density (LRD) and LOF scoring to more accurately identify malicious model updates, the incorporation of the contextual distance calculated in Equation (6.2.1) is a pivotal enhancement. It plays a crucial role in calculating the reachability distance $rd_k(x, y)$ between model updates, as follows:

$$rd_k(x, y) = \max\{k\text{-distance}(y), d_{\text{contextual}}(x, y)\} \tag{6.2.2}$$

  This calculation is integral to the LRD adjustment, where the contextual distance significantly boosts the model's ability to discern anomalies by accounting for the nuanced differences between model updates. The K-Nearest Neighbours (KNN) algorithm [115] is dynamically adjusted based on the data distribution, affecting the LRD, which is inversely proportional to the average reachability distance of the number of KNN k:

$$LRD_k(x) = \left(\frac{\sum_{y \in N_k(x)} rd_k(x, y)}{|N_k(x)|} \right)^{-1} \tag{6.2.3}$$

  Here, x represents a model update, $N_k(x)$ denotes the k nearest neighbors, and the enhanced reachability distance $rd_k(x, y)$, now including **contextual distance**, enriches how anomalies are evaluated relative to neighboring updates. This ensures that the LRD calculation captures the subtle differences between model updates, improving the model's sensitivity to anomalies.

  Consequently, the LOF score for each update is determined by comparing the LRD values of the update to those of its neighbors, effectively distinguishing between normal and malicious behavior:

$$LOF_k(x) = \frac{\sum_{y \in N_k(x)} \frac{LRD_k(y)}{LRD_k(x)}}{|N_k(x)|}. \tag{6.2.4}$$

Thus, $d_{\text{contextual}}(x, y)$'s integration into the LRD and LOF evaluations facilitates a nuanced analysis of model updates, refining the malicious detection mechanism's accuracy by aligning it closely with the data distribution's specific characteristics.

6.2.2 Phase two: Integration of GA for optimal thresholding

In this Phase, we refine our malicious model detection by identifying the best threshold for flagging malicious model updates, using a GA:

$$\text{Threshold}_{\text{optimal}} = \text{GA}(\text{LOF scores}, \text{validation data}) \tag{6.2.5}$$

This phase leverages the GA to pinpoint the most effective threshold based on LOF scores from Phase One and feedback from validation data selected portion of clients' datasets. The validation data is crucial for assessing the performance of various thresholds proposed by the GA, ensuring privacy as it's processed locally by clients. Here's how the process unfolds:

1. **Proposition of thresholds by GA:** The GA, informed by LOF scores and prior performance data, suggests several potential thresholds.

2. **Local testing by clients:** Each client applies these thresholds to its own validation data. This step ensures that the evaluation respects the privacy of each client's dataset.

3. **Sharing outcomes:** Clients use their validation data to test thresholds for detecting malicious models, for example, Label-flipping attacks in the MNIST dataset, where digits labels might be maliciously altered (e.g., '0' labeled as '9'). They calculate accuracy for these thresholds in identifying such attacks and share the results with the server, without sending actual data. This collaborative effort allows the server to aggregate accuracy scores from all clients, determining the most effective threshold for the entire FL process. By focusing on collective feedback, this method enhances the model's ability to resist Label-flipping attacks, ensuring the integrity of digit recognition tasks across the FL system, all while maintaining data privacy.

4. **Adjusting thresholds:** Based on the aggregated feedback, the GA fine-tunes the threshold values, aiming for optimal performance across the diverse environments of all clients.

The outcome is a precisely tuned threshold, optimized across the clients' federation, enhancing our model's sensitivity to malicious clients without compromising data privacy. This methodical approach ensures that the malicious model detection system is robust, adaptive, and capable of operating effectively within the FL system.

In Figure 6.1, we present a detailed flow chart of our new approach, which will be further explained in Algorithm 5. The NAM2D-FL approach can be concisely summarized as follows:

1. The server sends the initial global model to all clients for local training with their own data.

2. Clients return their trained models to the server.

3. The server checks the pre-accuracy of each model. Updates that surpass the global model's accuracy are approved.

4. For updates not meeting the initial accuracy, the server applies an Adaptive LOF method. This includes calculating distances between updates with adjusted weights for parameter variance and refining LOF scores by dynamically choosing an appropriate number of neighbors, enhancing anomaly detection.

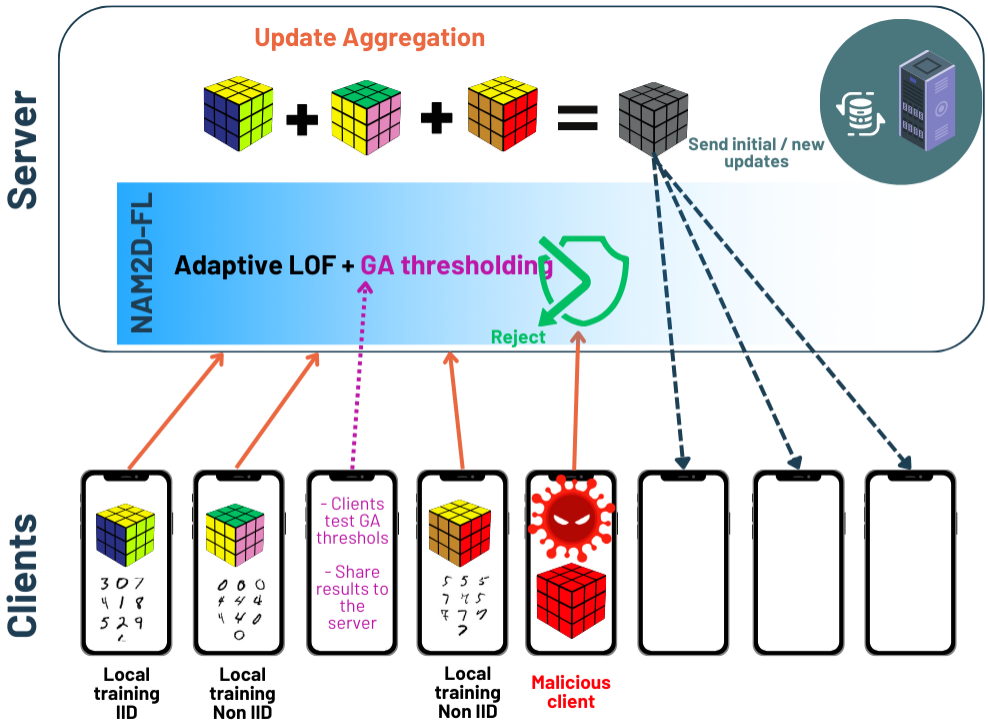


Figure 6.1: NAM2D-FL approach against Label-flipping and Backdoor attacks

5. The server generates thresholds using a GA based on the LOF score and sends them to clients, who evaluate these against their data and report the accuracy and corresponding thresholds back to the server.

6. Based on this collaborative feedback, the server fine-tunes the threshold to best identify malicious activity. Updates identified as malicious lead to the removal of the corresponding clients from the FL system.

7. Updates deemed benign are aggregated into the global model using FedAvg [14], and this updated model is redistributed to clients for further local training.

Algorithm 5: NAM2D-FL process

Require: Set of local models from clients, Initial global model's accuracy.

Receive local models from clients.
Aggregate local models to form a tentative global model.
Calculate the pre-accuracy of this global model. **if** *pre-accuracy* > *initial global model's accuracy* **then**
    Accept this global model update.
else
    Discard this update.
Normalize model updates considering expected parameter variance. **for** *each model update* x **do**
    for *each other model update* y **do**
        Compute the contextual distance $d_{\text{contextual}}$ as in Equation (6.2.1).
        Determine the parameter k of KNN algorithm [115] adaptively based on overall update variability.
        Compute the reachability distance $rd_k(x, y)$ as in Equation (6.2.2) using the adjusted contextual distance.
        Compute the LRD for x as in Equation (6.2.3).
        Compute the LOF score for x as in Equation (6.2.4).
    The server uses GA to calculate thresholds as in phase two of our proposed model, Equation (6.2.5), based on LOF score.
    The server send these thresholds to clients.
    The clients test and return feedback accuracies of corresponding thresholds to the server.
    Using feedback from clients, the server adjusts the threshold for identifying malicious activity.
    for *each client's update* x *do* **do**
        if $LOF_k(x) > GA$ *determined threshold* **then**
            Label the client as malicious and remove it.
        else
            Label the client as normal.
Aggregate updates from normal clients to update the global model using the FedAvg [14] algorithm.

6.3 Simulation results analysis

In this section, we conduct a thorough evaluation of our defense mechanism's effectiveness against Label-flipping and Backdoor attacks, focusing ACC and CPU run-time for each method. This analysis carried out with 50 clients on FL process within IoT networks, systematically compares these crucial metrics of our defense approach against those of renowned defense strategies listed in section 2.10: Median [54], TMean [54], MKrum [55], and FoolsGold [69] implemented using our FLSecLAB framework presented in section 2.7. Notably, this evaluation includes, for the first time, Non-IID distribution data, providing a novel perspective on defense robustness across different data distributions. The datasets examined MNIST [2], MNIST-Non-IID [2], CIFAR10 [9], and CIFAR10-Non-IID [9] with their specific distributions as expounded in section 2.12, utilize the corresponding models outlined in section 3.3. In this simulation, we define the attackers' proportion to be between 10% and 20% of the participating clients in FL process, While certain literature assumes higher attacker percentages, real-world FL scenarios rarely exhibit more than 20% attackers. For instance, having millions of users on Gboard [116], controlling a small percentage of user devices would require the attackers to compromise a large number of devices, necessitating significant effort and resources, making it impractical in most cases.

In the first experiment, we focused on the CIFAR10 [9] dataset under both IID and Non-IID settings. In Table 6.1, we noticed that our proposed NAM2D-FL approach demonstrates unparalleled performance in combating Label-flipping attacks, surpassing the others studied approaches discussed in section 2.10: FedAvg [14], Median [54], TMean [54], MKrum [55], and FoolsGold [69] across multiple evaluation metrics. Specifically, NAM2D-FL achieved an ACC of 89% for CIFAR10 and 91% for CIFAR10-Non-IID, marking significant improvements of at least 12% over the closest competitors. This indicates the robust adaptability and effectiveness of NAM2D-FL under varied data distribution scenarios. Moreover, in terms of adversarial defense, represented by the ASR, we showed in Table 6.1 that, the NAM2D-FL approach showcased outstanding resilience, maintaining an ASR at only 10% for CIFAR10 and further reducing it to 12% for CIFAR10-Non-IID. This demonstrates NAM2D-FL's effectiveness in protecting the model's integrity against Label-flipping attacks, notably outperforming other methods where the lowest ASR observed was 18% and 20%, respectively. The Precision and Recall outcomes further solidify the efficacy of NAM2D-FL. Achieving the highest scores in these metrics across both IID and Non-IID settings for CIFAR10 validates not only the accuracy but also the reliability and consistency of our model, ensuring that it not only correctly identifies true instances but also minimizes false detections.

Table 6.1: Performance metrics under label-lipping attacks using CIFAR10 and CIFAR10-Non-IID

Benchmark / Method	FedAvg	Median	TMean	MKrum	FGold	**NAM2D-FL**
CIFAR10						
ACC%	70%	71%	70.5%	72%	73%	**89%**
ASR%	20%	19.5%	19.8%	18.5%	18%	**10%**
Precision%	77%	77.5%	77.2%	78%	78.5%	**92%**
Recall%	75%	75.5%	75.2%	76%	76.5%	**89%**
CIFAR10-Non-IID						
ACC%	73%	74%	73.5%	75%	79%	**91%**
ASR%	22%	21.5%	21.8%	20.5%	20.0	**12%**
Precision%	78%	78.5%	78.2%	79%	79.5	**86%**
Recall%	76%	76.5%	76.2%	77%	77.5%	**84%**

Transitioning to the second experiment with the MNIST [2] dataset, The NAM2D-FL method keeps succeeding. Within both IID and Non-IID data distributions, as we noticed in Table 6.2, our NAM2D-FL exhibited remarkable performance compared with other studied approaches presented in section 2.10: FedAvg [14], Median [54], TMean [54], MKrum [55], and FoolsGold [69] across multiple evaluation metrics, achieving an accuracy of 89.0% on the MNIST and enhancing it to 93.0% under MNIST-Non-IID. Similarly, NAM2D-FL effectively defends against Label-flipping attacks in both IID and Non-IID MNIST datasets, achieving the lowest ASR and outperforming competing approaches in Precision and Recall. This underscores its superior efficacy and robustness across various metrics, establishing it as a significant advancement in securing and enhancing the reliability of the FL system.

Table 6.3 presents the results of the third experiment, demonstrating the significant superiority of the NAM2D-FL approach in mitigating Backdoor attacks within both IID and Non-IID CIFAR10 [9] dataset. NAM2D-FL achieves remarkable accuracies of 85% and 86% for CIFAR10 and CIFAR10-Non-IID, respectively, outperforming its closest competitors by 7.5%. Additionally, its ASR stands at 15% and 18%, surpassing the nearest rivals by 8% and 10.0% in each scenario. Moreover, NAM2D-FL exhibits higher Precision and Recall, surpassing the closest methods by 6% and 7%, respectively. These results underscore NAM2D-FL's robust efficacy in enhancing both performance and security against Backdoor attacks in the context of FL system.

In the fourth experiment, the NAM2D-FL method significantly outshines its competitors in combating Backdoor attacks on both of IID and Non-IID MNIST [2] dataset as we showed in Table 6.4. It achieves a high accuracy of 91% for MNIST and 90% for MNIST-Non-IID, which is 7.0% higher than the nearest competitor in both cases. Furthermore, the method demonstrates robust defense capabilities, with an ASR of 17% for MNIST and 16% for MNIST-Non-IID, outperforming the closest

Table 6.2: Performance metrics under label-lipping attacks using MNIST and MNIST-Non-IID

Benchmark / Method	FedAvg	Median	TMean	MKrum	FoolsGold	**NAM2D-FL**
MNIST						
ACC%	69%	78%	77.5%	79%	80%	**87%**
ASR%	21%	20.5%	20.8%	19.5%	19%	**11%**
Precision%	74%	79.5%	79.2%	80%	80.5%	**89%**
Recall%	73%	78.5%	78.2%	79%	79.5%	**86%**
MNIST-Non-IID						
ACC%	78%	79%	78.5%	80%	81%	**93%**
ASR%	23%	22.5%	22.8%	21.5%	21%	**13%**
Precision%	80%	80.5%	80.2%	81%	81.5%	**88%**
Recall%	79%	79.5%	79.2%	80%	80.5%	**89%**

Table 6.3: Performance metrics under Backdoor attacks using CIFAR10 and CIFAR10-Non-IID

Benchmark / Method	FedAvg	Median	TMean	MKrum	FoolsGold	**NAM2D-FL**
CIFAR10						
ACC%	70%	76%	76.5%	77%	77.5%	**85%**
ASR%	25%	24.5%	24%	23.5%	23%	**15%**
Precision%	80%	80.5%	81%	81.5%	82%	**88%**
Recall%	78%	78.5%	79%	79.5%	80%	**87%**
CIFAR10-Non-IID						
ACC%	69%	77%	77.5%	78%	78.5%	**86%**
ASR%	30%	29.5%	29%	28.5%	28%	**18%**
Precision%	75%	81.5%	82%	82.5%	83%	**89%**
Recall%	76%	79.5%	80%	80.5%	81%	**88%**

contenders by 9 and 11 percent, respectively. Precision and Recall metrics underscore its efficacy, with NAM2D-FL achieving 93% and 92% for MNIST, and 94% and 91% for MNIST-Non-IID, leading the nearest competitors by significant margins. These results affirm the NAM2D-FL method's outstanding performance against Backdoor attacks in FL scenarios.

Table 6.4: Performance metrics under Backdoor attacks using MNIST and MNIST-Non-IID

Benchmark / Method	FedAvg	Median	TMean	MKrum	FoolsGold	**NAM2D-FL**
MNIST						
ACC%	65%	82.5%	83%	83.5%	84%	**91%**
ASR%	28%	27.5%	27%	26.5%	26%	**17%**
Precision%	70%	85.5%	86%	86.5%	87%	**93%**
Recall%	72%	83.5%	84%	84.5%	85%	**92%**
MNIST-Non-IID						
ACC%	73%	83.5%	84%	84.5%	85%	**90%**
ASR%	29%	28.5%	28%	27.5%	27%	**16%**
Precision%	76%	86.5%	87%	87.5%	88%	**94%**
Recall%	74%	84.5%	85%	85.5%	86%	**91%**

In the fifth experiment, as illustrated in Figure 6.2, we examined the CPU run-time per iteration required by other studied approaches introduced in section 2.10 against Label-flipping attacks across different datasets, excluding FedAvg [14], which simply computes the average of updates without intending to defend against attacks. We found that NAM2D-FL outperformed other approaches in efficiency across different scenarios. On the CIFAR10 [9] dataset, NAM2D-FL required only 1.0 seconds, which was quicker than Median [54], MKrum [55], and FoolsGold [69] approaches. For the CIFAR10-Non-IID variant, NAM2D-FL records a run-time of 1.1 seconds indicating superior

performance against the other approaches. On the MNIST dataset, NAM2D-FL achieved a run-time of 0.071 seconds, highlighting its notable run-time over the other methods. Even in the MNIST-Non-IID context, it maintained superior performance, completing in 0.81 seconds. These findings underscore NAM2D-FL's efficacy and efficiency in countering Label-flipping attacks, establishing it as a preferable defense option with minimal performance impact.

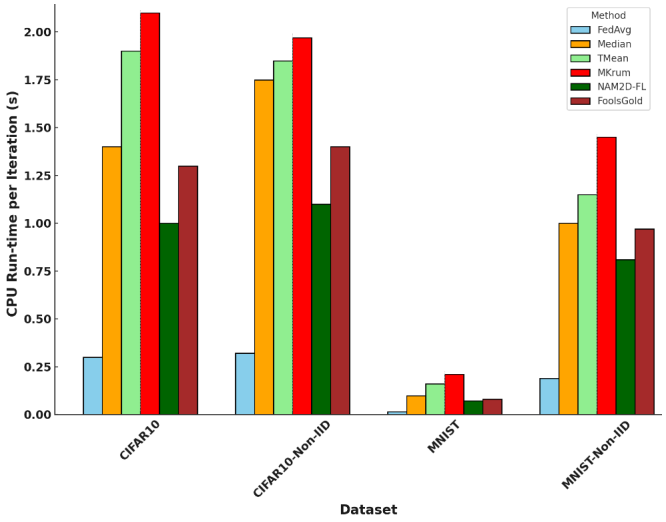


Figure 6.2: CPU run-time per iteration (in seconds) for Label-flipping attacks across datasets

In the sixth experiment, we concentrate on evaluating the CPU run-time per iteration for the NAM2D-FL approach against Backdoor attacks across diverse datasets. As presented in Figure 6.3, in the CIFAR10 [9], NAM2D-FL shows a run-time of 0.9 seconds, outperforming the other approaches: Median [54], TMean [54], and MKrum [55], and FoolsGold [69]. Within the CIFAR10-Non-IID [9], its efficiency is highlighted with a 1.2 second run-time, surpassing the slower performances of other studied approaches. For the MNIST [2] dataset, NAM2D-FL's run-time is notably efficient at 0.045 seconds outperforming the other defenses. In the scenario of MNIST-Non-IID [2], NAM2D-FL's efficiency is again evident at 0.55 seconds, showcasing better performance than the compared approaches. These findings affirm that NAM2D-FL not only provides a strong defense against Backdoor attacks but does so with commendable computational efficiency.

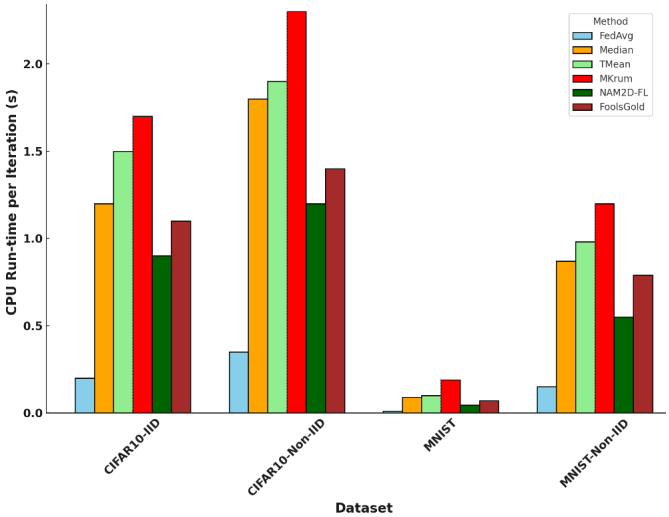


Figure 6.3: CPU run-time per iteration (in seconds) for backdoor attacks across datasets

6.4 Conclusion

In this chapter, we present a groundbreaking advancement in FL with the introduction of the NAM2D-FL approach. Our research was motivated by the imperative to address the complex challenges posed by both IID and Non-IID data settings in the face of Label-flipping and Backdoor attacks, prevalent in real-world implementations. After a detailed review of current strategies and their shortcomings, detailed in section 2.10, our approach incorporates a dual-side server-client mechanism, augmented by the integration of adaptive LOF and GA-based thresholding. This combination aims to enhance the robustness and effectiveness of the FL system in real-world scenarios. Our evaluation, conducted utilizing datasets including the MNIST and CIFAR for both IID and Non-IID data settings, clearly demonstrates the superior performance of the NAM2D-FL approach across several key metrics. This achievement underscores our contribution to enhancing the security and integrity of the FL system by defending against emerging poisoning attacks, thereby fostering increased trust and confidence in collaborative learning paradigms.

Chapter 7

General conclusion and perspectives

Contents

7.1 Conclusion

FL is an innovative ML approach that allows clients to collaboratively train a model while keeping their data private. FL raises important issues regarding data privacy and security. Yet, the transparency of FL system, particularly in the growing field of IoT networks, exposes them to poisoning attacks especially the Label-flipping and Backdoor attacks. These malicious activities, which involve tampering with data to compromise the model's reliability, present significant threats to privacy and security, underscoring the essential requirement for strong defense mechanisms.

The first contribution of this thesis was to provide a thorough examination of recent advancements in FL within IoT networks. This exploration encompasses an extensive literature review of FL's fundamentals, applications, its merits, and the critical challenges and security concerns it faces. Specifically, we delve into the emerging landscape of poisoning attacks targeted at FL within these IoT networks, categorizing the types of datasets employed. After that, we provide a comprehensive literature review of the existing defense approaches in FL system within IoT networks, we adopt a classification of these techniques based on three main categories that cover all previous works in literature while highlighting the advantages and disadvantages of each of them and with a particular focus on malicious model detection techniques. Then we related works for each category and provide a comparative study of malicious model detection approaches. Subsequently, we introduce FLSecLAB, our innovative FL framework designed for IoT networks. This framework stands out for its adaptability, allowing for rigorous testing across a variety of scenarios, thereby marking a significant leap forward in enhancing the security posture of the FL system, through detailed evaluation and comparison. This discussion not only sheds light on the current state of FL security but also sets the groundwork for future advancements in this rapidly evolving field.

Following the literature review, the second contribution was to introduce the EMDG-FL approach, a new approach to enhance the malicious model detection in FL system against Label-flipping attacks within IoT networks. EMDG-FL is built on the IForest algorithm for calculating the malicious score. Then, the GA is used to discover an optimal threshold based on this score to identify malicious clients in the FL system. EMDG-FL aims to safeguard FL integrity, boost efficiency, and reduce computational costs.

After that, the third contribution was to propose the ERD-FL approach, an innovative defense strategy employing entropy analysis for detecting and mitigating the impact of Label-flipping attacks. In the new ERD-FL strategy, the focus shifts from changing IForest to overcoming its weaknesses within EMDG-FL. This part introduces ERD-FL as a novel, scalable defense for FL in IoT networks. It's the first of its kind to use entropy data and a flexible threshold to better detect and counteract Label-flipping attacks, aiming for more accurate results. We use in our simulation scenarios a wide clients' number and different dataset types, including image-based and text-based datasets.

The fourth contribution was then extended to the M3D-FL approach, which introduced a multi-layered protection strategy to secure the FL system. While the ERD-FL method uses entropy data to enhance the detection of malicious clients in the FL system. However, its dependence on historical entropy data may restrict its capacity to promptly adapt to novel or sudden attack strategies, potentially affecting its effectiveness. By integrating the LOF and MAD outlier detection algorithms, M3D-FL demonstrated superior performance in more intricate scenarios involving Label-flipping and Backdoor simultaneous attacks, as well as a larger clients' number.

Finally, the fifth contribution was to propose the NAM2D-FL approach, targeting the nuanced challenges posed by Non-IID data in the presence of both Label-flipping and Backdoor attacks. The LOF method used in M3D-FL is efficient at detecting malicious clients but struggles with Non-IID data in the FL system due to its inability to manage varied data distributions. This constraint not only exposes vulnerabilities but also impacts its efficacy in high-dimensional data scenarios. Through the integration of server-client mechanisms by the integration of refined LOF and adaptive GA-based thresholding, NAM2D-FL significantly enhanced the robustness and effectiveness of the FL system, demonstrating improvements with a wide range of evaluation performance metrics, and comparison with several defense approaches in FL real-world settings.

Our proposed approaches in this thesis showed high performance in terms of several performance metrics such as ACC, LR, ASR, CPU run-time, CPU aggregation run-time, Precision, and Recall compared to other works in the literature. This evidences not only the effectiveness of the proposed solutions in enhancing the security of FL systems but also their potential to facilitate the wider adoption of FL in IoT environments, contributing to the development of more resilient and efficient collaborative learning models.

7.2 Perspectives

The advent of FL in IoT represents a paradigm shift in how we approach ML, particularly with the integration of trendy fields like Large Language Models (LLMs). LLMs, which are advanced AI systems capable of understanding and generating human-like text, enrich FL by offering sophisticated data analysis and decision-making capabilities. This methodology not only promises enhanced privacy and data security but also paves the way for more personalized and efficient computing at the edge. However, this integration introduces complex challenges, including data privacy, model scalability, and computational efficiency. Addressing these challenges is paramount for the advancement of FL in IoT environments.

The following sections provide an in-depth perspective on these critical areas, offering insights into current research directions and future trends.

1. **Privacy-preserving mechanisms for LLMs**:

   Within the integration of FL and LLMs, emphasizing privacy-preserving mechanisms is crucial for IoT. This section will highlight key methodologies that fortify data privacy, detailing advanced solutions to protect user information within decentralized LLM frameworks, as follows:

   - **Enhancing data security in training**:
     The core of FL's appeal lies in its ability to train models on decentralized data, thereby preserving privacy. However, the scale of data required by LLMs necessitates novel privacy-preserving mechanisms. Techniques like DP, HE, and SMPC offer promising pathways. These methods ensure that individual data points cannot be reverse-engineered from the model, thereby safeguarding user privacy.

- **Mitigating data exposure risks**:
  Beyond encryption, there's a growing interest in exploring how to minimize data exposure during the training of LLMs without compromising the model's quality. Advanced data anonymization techniques, combined with secure aggregation protocols, can further reduce the risk of sensitive information leakage. Research into privacy-preserving data augmentation and federated transfer learning could also play a significant role in enhancing model performance under stringent privacy constraints.

2. **Training cost and efficiency**:

   Addressing training cost and efficiency is vital in the convergence of FL and LLMs. This part will focus on optimizing resource utilization and computational strategies, aiming to refine the balance between efficiency and model efficacy in FL environments, as follows:

   - **Addressing computational demands**:
     The computational intensity of training large models in a FL context is a significant hurdle. Innovative approaches to reduce this burden include model compression techniques, such as pruning and quantization, and efficient model training algorithms that reduce communication overhead and computational requirements without significant loss in accuracy.
   - **Cost-effective model training strategies**:
     Developing strategies for cost-effective model training involves not just algorithmic improvements but also infrastructure optimization. Adaptive model training that adjusts model complexity based on the available computational resources and network conditions could offer a path forward. Additionally, leveraging cloud computing resources during off-peak hours for more intensive training tasks could balance the cost and efficiency equation.

3. **Adaptive FL frameworks**:

   Adaptive FL frameworks are essential for the evolving requirements of IoT, ensuring the relevance of LLMs. This section is dedicated to exploring adaptive strategies and personalization techniques that enhance the responsiveness of FL system to variable data and environments, as follows:

   - **Dynamic model adaptation**:
     Adapting FL frameworks to dynamically adjust to changing network environments and data distributions is crucial for the deployment of LLMs in IoT. This involves developing algorithms that can modify their learning strategy based on real-time feedback, ensuring optimal performance across diverse devices and data streams.
   - **Personalization and context-aware learning**:
     Personalization is key in IoT applications adaptive FL frameworks can enable LLMs to tailor their predictions and functionalities to individual users or devices, enhancing the user experience. Context-aware learning mechanisms that incorporate situational data can further refine model accuracy and relevance.

4. **The role of edge computing in FL**:

   Merging edge computing with FL signifies a strategic enhancement for LLMs in IoT. This segment will address how edge computing contributes to scalability, privacy, and latency reduction, emphasizing its role in advancing FL methodologies for improved distributed intelligence, as follows:

   - **Leveraging edge devices for scalability**:
     Integrating edge computing with FL can address scalability and latency issues. By processing data on local devices and only sharing model updates, we can significantly reduce the bandwidth required for model training. This approach also opens up new avenues for real-time learning and inference, crucial for time-sensitive IoT applications.

- **Enhancing privacy and efficiency**:
  Edge computing inherently supports privacy-preserving objectives by keeping data localized. Combining this with LLMs trained via FL can create a powerful paradigm for building intelligent systems that are both privacy-centric and efficient. Research into optimizing edge device capabilities for LLM training and inference is an exciting frontier that could redefine the boundaries of FL.

This perspective has illuminated the dynamic interplay between FL and LLMs within the IoT ecosystem, highlighting the transformative potential and the myriad challenges this integration presents. By focusing on privacy-preserving mechanisms, training cost and efficiency, adaptive FL frameworks, and the pivotal role of edge computing, we pave the way for a more secure, efficient, and personalized future in IoT. As we continue to explore and refine these technologies, the collaborative effort between academia, industry, and regulatory bodies will be crucial in navigating the complexities of data privacy, computational demands, and the seamless integration of AI into our daily lives. The progress in these areas not only promises to enhance the capabilities of IoT devices but also sets a new standard for the responsible and innovative use of AI, ensuring that the benefits of such technologies are accessible to all.

Publications

International Peer-Reviewed Journals

- **O. Ben Atia**, M. A. Samara, I. Bennis, J. Gaber, A. Abouaissa, P. Lorenz, M3D-FL: Multi-layer Malicious Model Detection for Federated Learning in IoT networks, **(2024) Journal of Computer and Security** (Submitted).
- **O. Ben Atia**, M. A. Samara, I. Bennis, J. Gaber, A. Abouaissa, P. Lorenz, A comprehensive literature review of attacks and defense mechanisms in Federated Learning system in IoT networks, **(2024) Journal of Machine Learning Research** (Submitted).

International Peer-Reviewed Conferences

- **O. Ben Atia**, M. Al Samara, I. Bennis, J. Gaber, A. Abouaissa, P. Lorenz, FMT-FL: Forecasting Malicious Threats-Federated Learning via Time-Series" **IEEE WCCS conference (2024)**. (Under construction)
- **O. Ben Atia**, M. Al Samara, I. Bennis, J. Gaber, A. Abouaissa, P. Lorenz, NAM2D-FL: Non-IID Adaptive Malicious Model Detection in Federated Learning system in IoT networks," **IEEE Globecom conference (2024)**. (Submitted)
- **O. Ben Atia**, M. Al Samara, I. Bennis, J. Gaber, A. Abouaissa, P. Lorenz, ERD-FL: Entropy-Driven Robust Defense for Federated Learning , **IEEE IWCMC Conference (2024)**. (Accepted)
- **O. Ben Atia**, M. Al Samara, I. Bennis, J. Gaber, A. Abouaissa, P. Lorenz, EMDG-FL: Enhanced Malicious Model Detection based on Genetic Algorithm for Federated Learning in IoT networks, **IEEE WCNC Conference (2024)**. (Accepted)

Bibliography

1. Alam, T., Qamar, S., Dixit, A. & Benaida, M. Genetic algorithm: Reviews, implementations, and applications. *arXiv preprint arXiv:2007.12673* (2020).
2. LeCun, Y., Bottou, L., Bengio, Y. & Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE* **86,** 2278–2324 (1998).
3. Xiao, H., Rasul, K. & Vollgraf, R. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747* (2017).
4. François-Lavet, V., Henderson, P., Islam, R., Bellemare, M. G., Pineau, J., *et al.* An introduction to deep reinforcement learning. *Foundations and Trends in Machine Learning* **11,** 219–354 (2018).
5. Maas, A. *et al. Learning word vectors for sentiment analysis* in *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies* (2011), 142–150.
6. Breunig, M. M., Kriegel, H.-P., Ng, R. T. & Sander, J. *LOF: identifying density-based local outliers* in *Proceedings of the 2000 ACM SIGMOD international conference on Management of data* (2000), 93–104.
7. Liu, F. T., Ting, K. M. & Zhou, Z.-H. *Isolation forest* in *2008 eighth ieee international conference on data mining* (2008), 413–422.
8. Leys, C., Ley, C., Klein, O., Bernard, P. & Licata, L. Detecting outliers: Do not use standard deviation around the mean, use absolute deviation around the median. *Journal of experimental social psychology* **49,** 764–766 (2013).
9. Krizhevsky, A., Hinton, G., *et al.* Learning multiple layers of features from tiny images (2009).
10. McMahan, B., Moore, E., Ramage, D., Hampson, S. & y Arcas, B. A. *Communication-efficient learning of deep networks from decentralized data* in *Artificial intelligence and statistics* (2017), 1273–1282.
11. Li, T. *et al.* Federated optimization in heterogeneous networks. *Proceedings of Machine learning and systems* **2,** 429–450 (2020).
12. Guendouzi, B. S., Ouchani, S., Assaad, H. E. & Zaher, M. E. A systematic review of federated learning: Challenges, aggregation methods, and development tools. *Journal of Network and Computer Applications,* 103714 (2023).
13. Yang, Q. *et al.* Federated learning: synthesis lectures on artificial intelligence and machine learning. *vol* **13,** 1–207 (2019).
14. McMahan, B., Moore, E., Ramage, D., Hampson, S. & y Arcas, B. A. *Communication-efficient learning of deep networks from decentralized data* in *Artificial intelligence and statistics* (2017), 1273–1282.
15. Bonawitz, K. *et al. Practical secure aggregation for privacy-preserving machine learning* in *proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security* (2017), 1175–1191.

16. Ahn, J.-H., Simeone, O. & Kang, J. *Wireless federated distillation for distributed edge learning with heterogeneous data* in *2019 IEEE 30th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)* (2019), 1–6.

17. Sanislav, T., Mois, G. D., Zeadally, S. & Folea, S. C. Energy harvesting techniques for internet of things (IoT). *IEEE Access* **9,** 39530–39549 (2021).

18. Mohammadi, M., Al-Fuqaha, A., Sorour, S. & Guizani, M. Deep learning for IoT big data and streaming analytics: A survey. *IEEE Communications Surveys & Tutorials* **20,** 2923–2960 (2018).

19. Nguyen, D. C. *et al.* Enabling AI in future wireless networks: A data life cycle perspective. *IEEE Communications Surveys & Tutorials* **23,** 553–595 (2020).

20. Goddard, M. The EU General Data Protection Regulation (GDPR): European regulation that has a global impact. *International Journal of Market Research* **59,** 703–705 (2017).

21. Nguyen, D. C. *et al.* Federated learning meets blockchain in edge computing: Opportunities and challenges. *IEEE Internet of Things Journal* **8,** 12806–12825 (2021).

22. Sheller, M. J., Reina, G. A., Edwards, B., Martin, J. & Bakas, S. *Multi-institutional deep learning modeling without sharing patient data: A feasibility study on brain tumor segmentation* in *Brainlesion: Glioma, Multiple Sclerosis, Stroke and Traumatic Brain Injuries: 4th International Workshop, BrainLes 2018, Held in Conjunction with MICCAI 2018, Granada, Spain, September 16, 2018, Revised Selected Papers, Part I 4* (2019), 92–104.

23. Liu, Y., James, J., Kang, J., Niyato, D. & Zhang, S. Privacy-preserving traffic flow prediction: A federated learning approach. *IEEE Internet of Things Journal* **7,** 7751–7763 (2020).

24. Chen, H. *et al.* Federated learning over wireless IoT networks with optimized communication and resources. *IEEE Internet of Things Journal* **9,** 16592–16605 (2022).

25. Rani, S., Kataria, A., Kumar, S. & Tiwari, P. Federated learning for secure IoMT-applications in smart healthcare systems: A comprehensive review. *Knowledge-Based Systems,* 110658 (2023).

26. Lim, W. Y. B. *et al.* Federated learning in mobile edge networks: A comprehensive survey. *IEEE Communications Surveys & Tutorials* **22,** 2031–2063 (2020).

27. Wu, Q., Chen, X., Zhou, Z. & Zhang, J. Fedhome: Cloud-edge based personalized federated learning for in-home health monitoring. *IEEE Transactions on Mobile Computing* **21,** 2818–2832 (2020).

28. Von Krogh, G. *et al.* The privacy and security implications of open data in healthcare: a contribution from the IMIA Open Source Working Group. *Yearbook of medical informatics* **27,** 041–047 (2018).

29. Konečnỳ, J., McMahan, H. B., Ramage, D. & Richtárik, P. Federated optimization: Distributed machine learning for on-device intelligence. *arXiv preprint arXiv:1610.02527* (2016).

30. Pandya, S. *et al.* Federated learning for smart cities: A comprehensive survey. *Sustainable Energy Technologies and Assessments* **55,** 102987 (2023).

31. Yang, Z., Chen, M., Wong, K.-K., Poor, H. V. & Cui, S. Federated learning for 6G: Applications, challenges, and opportunities. *Engineering* **8,** 33–41 (2022).

32. Nguyen, M.-D. *et al.* Hcfl: A high compression approach for communication-efficient federated learning in very large scale iot networks. *IEEE Transactions on Mobile Computing* (2022).

33. Wang, T. *et al.* Edge-based communication optimization for distributed federated learning. *IEEE Transactions on Network Science and Engineering* **9,** 2015–2024 (2021).

34. Mothukuri, V. *et al.* Federated-learning-based anomaly detection for IoT security attacks. *IEEE Internet of Things Journal* **9,** 2545–2554 (2021).

35. Zhao, Y. *et al.* Federated learning with non-iid data. *arXiv preprint arXiv:1806.00582* (2018).

36. Le, J. *et al.* Federated continuous learning with broad network architecture. *IEEE Transactions on Cybernetics* **51,** 3874–3888 (2021).

37. Xia, J. *et al.* PervasiveFL: Pervasive federated learning for heterogeneous IoT systems. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* **41,** 4100–4111 (2022).

38. Lyu, L., Yu, H. & Yang, Q. Threats to federated learning: A survey. *arXiv preprint arXiv:2003.02133* (2020).

39. Zhao, C. *et al.* Secure Multi-Party Computation: Theory, practice and applications. *Information Sciences* **476,** 357–372. ISSN: 0020-0255 (2019).

40. Sun, L., Qian, J. & Chen, X. LDP-FL: Practical private aggregation in federated learning with local differential privacy. *arXiv preprint arXiv:2007.15789* (2020).

41. Alamri, M., Jhanjhi, N. & Humayun, M. Blockchain for Internet of Things (IoT) research issues challenges & future directions: A review. *Int. J. Comput. Sci. Netw. Secur* **19,** 244–258 (2019).

42. Chen, Y., Qin, X., Wang, J., Yu, C. & Gao, W. Fedhealth: A federated transfer learning framework for wearable healthcare. *IEEE Intelligent Systems* **35,** 83–93 (2020).

43. Lyu, L. *et al.* Towards fair and privacy-preserving federated deep models. *IEEE Transactions on Parallel and Distributed Systems* **31,** 2524–2541 (2020).

44. Chen, Z., Tian, P., Liao, W. & Yu, W. Towards multi-party targeted model poisoning attacks against federated learning systems. *High-Confidence Computing* **1,** 100002 (2021).

45. Sun, J. *et al. FL-WBC: Enhancing Robustness against Model Poisoning Attacks in Federated Learning from a Client Perspective* in *Advances in Neural Information Processing Systems* (eds Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P. & Vaughan, J. W.) **34** (Curran Associates, Inc., 2021), 12613–12624.

46. Sun, G. *et al.* Data Poisoning Attacks on Federated Machine Learning. *IEEE Internet of Things Journal* **9,** 11365–11375 (2022).

47. Tolpegin, V., Truex, S., Gursoy, M. E. & Liu, L. *Data Poisoning Attacks Against Federated Learning Systems* in *Computer Security – ESORICS 2020* (eds Chen, L., Li, N., Liang, K. & Schneider, S.) (Springer International Publishing, Cham, 2020), 480–501. ISBN: 978-3-030-58951-6.

48. Gong, X., Chen, Y., Wang, Q. & Kong, W. Backdoor Attacks and Defenses in Federated Learning: State-of-the-Art, Taxonomy, and Future Directions. *IEEE Wireless Communications* **30,** 114–121 (2023).

49. Zhang, J., Zhang, J., Chen, J. & Yu, S. *GAN Enhanced Membership Inference: A Passive Local Attack in Federated Learning* in *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)* (2020), 1–6.

50. Ceballos, I. *et al.* Splitnn-driven vertical partitioning. *arXiv preprint arXiv:2008.04137* (2020).

51. Cyffers, E. & Bellet, A. *Privacy Amplification by Decentralization* in *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics* (eds Camps-Valls, G., Ruiz, F. J. R. & Valera, I.) **151** (PMLR, 28–30 Mar 2022), 5334–5353.

52. Kadhe, S., Rajaraman, N., Koyluoglu, O. O. & Ramchandran, K. Fastsecagg: Scalable secure aggregation for privacy-preserving federated learning. *arXiv preprint arXiv:2009.11248* (2020).

53. Xu, G., Li, H., Liu, S., Yang, K. & Lin, X. Verifynet: Secure and verifiable federated learning. *IEEE Transactions on Information Forensics and Security* **15,** 911–926 (2019).

54. Yin, D., Chen, Y., Kannan, R. & Bartlett, P. *Byzantine-robust distributed learning: Towards optimal statistical rates* in *International Conference on Machine Learning* (2018), 5650–5659.

55. Blanchard, P., El Mhamdi, E. M., Guerraoui, R. & Stainer, J. Machine learning with adversaries: Byzantine tolerant gradient descent. *Advances in neural information processing systems* **30** (2017).

56. Cao, X., Fang, M., Liu, J. & Gong, N. Z. Fltrust: Byzantine-robust federated learning via trust bootstrapping. *arXiv preprint arXiv:2012.13995* (2020).

57. Kim, M., Song, Y., Wang, S., Xia, Y., Jiang, X., *et al.* Secure logistic regression based on homomorphic encryption: Design and evaluation. *JMIR medical informatics* **6,** e8805 (2018).

58. Dwork, C., Roth, A., *et al.* The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science* **9,** 211–407 (2014).

59. Yao, A. C. *Protocols for secure computations* in *23rd annual symposium on foundations of computer science (sfcs 1982)* (1982), 160–164.

60. El Ouadrhiri, A. & Abdelhadi, A. Differential privacy for deep and federated learning: A survey. *IEEE access* **10,** 22359–22380 (2022).

61. Acar, A., Aksu, H., Uluagac, A. S. & Conti, M. A survey on homomorphic encryption schemes: Theory and implementation. *ACM Computing Surveys (Csur)* **51,** 1–35 (2018).

62. Van Dijk, M., Gentry, C., Halevi, S. & Vaikuntanathan, V. *Fully homomorphic encryption over the integers* in *Advances in Cryptology–EUROCRYPT 2010: 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, French Riviera, May 30–June 3, 2010. Proceedings 29* (2010), 24–43.

63. Bendlin, R., Damg⟩*ard, I.*, Orlandi, C. & Zakarias, S. *Semi-homomorphic encryption and multiparty computation* in *Annual International Conference on the Theory and Applications of Cryptographic Techniques* (2011), 169–188.

64. Holohan, N., Antonatos, S., Braghin, S. & Mac Aonghusa, P. The bounded laplace mechanism in differential privacy. *arXiv preprint arXiv:1808.10410* (2018).

65. Dong, J., Roth, A. & Su, W. J. Gaussian differential privacy. *Journal of the Royal Statistical Society Series B: Statistical Methodology* **84,** 3–37 (2022).

66. Lu, S., Li, R., Liu, W. & Chen, X. Defense against backdoor attack in federated learning. *Computers & Security* **121,** 102819 (2022).

67. Qayyum, A., Janjua, M. U. & Qadir, J. Making federated learning robust to adversarial attacks by learning data and model association. *Computers & Security* **121,** 102827 (2022).

68. Gu, X. *et al.* Privacy, accuracy, and model fairness trade-offs in federated learning. *Computers & Security* **122,** 102907 (2022).

69. Fung, C., Yoon, C. J. & Beschastnikh, I. Mitigating sybils in federated learning poisoning. *arXiv preprint arXiv:1808.04866* (2018).

70. Zhao, L. *et al.* Shielding collaborative learning: Mitigating poisoning attacks through client-side detection. *IEEE Transactions on Dependable and Secure Computing* **18,** 2029–2041 (2020).

71. Kang, J. *et al.* Reliable federated learning for mobile networks. *IEEE Wireless Communications* **27,** 72–80 (2020).

72. Tolpegin, V., Truex, S., Gursoy, M. E. & Liu, L. *Data poisoning attacks against federated learning systems* in *Computer Security–ESORICS 2020: 25th European Symposium on Research in Computer Security, ESORICS 2020, Guildford, UK, September 14–18, 2020, Proceedings, Part I 25* (2020), 480–501.

73. Zhao, Y. *et al.* Detecting and mitigating poisoning attacks in federated learning using generative adversarial networks. *Concurrency and Computation: Practice and Experience* **34,** e5906 (2022).

74. Liu, W. *et al.* D2MIF: A malicious model detection mechanism for federated learning empowered artificial intelligence of things. *IEEE Internet of Things Journal* (2021).

75. Kim, Y., Chen, H. & Koushanfar, F. Backdoor Defense in Federated Learning Using Differential Testing and Outlier Detection. *arXiv preprint arXiv:2202.11196* (2022).

76. Lai, Y.-C. *et al.* Two-phase Defense Against Poisoning Attacks on Federated Learning-based Intrusion Detection. *Computers & Security* **129,** 103205 (2023).

77. Yang, R., He, H., Wang, Y., Qu, Y. & Zhang, W. Dependable federated learning for IoT intrusion detection against poisoning attacks. *Computers & Security* **132,** 103381 (2023).

78. Zhu, C., Zhang, J., Sun, X., Chen, B. & Meng, W. ADFL: Defending Backdoor Attacks in Federated Learning via Adversarial Distillation. *Computers & Security,* 103366 (2023).
79. Panigrahi, R. & Borah, S. A detailed analysis of CICIDS2017 dataset for designing Intrusion Detection Systems. *International Journal of Engineering & Technology* **7,** 479–482 (2018).
80. Yamashita, R., Nishio, M., Do, R. K. G. & Togashi, K. Convolutional neural networks: an overview and application in radiology. *Insights into imaging* **9,** 611–629 (2018).
81. Cao, Q., Shen, L., Xie, W., Parkhi, O. M. & Zisserman, A. *Vggface2: A dataset for recognising faces across pose and age* in *2018 13th IEEE international conference on automatic face & gesture recognition (FG 2018)* (2018), 67–74.
82. Tavallaee, M., Bagheri, E., Lu, W. & Ghorbani, A. A. *A detailed analysis of the KDD CUP 99 data set* in *2009 IEEE symposium on computational intelligence for security and defense applications* (2009), 1–6.
83. Dheeru, D. Karra taniskidou e. *UCI machine learning repository* **12** (2017).
84. Chelli, M. *et al. FedGuard: Selective Parameter Aggregation for Poisoning Attack Mitigation in Federated Learning* in *2023 IEEE International Conference on Cluster Computing (CLUSTER)* (2023), 72–81.
85. Bolze, R. *et al.* Grid'5000: A large scale and highly reconfigurable experimental grid testbed. *The International Journal of High Performance Computing Applications* **20,** 481–494 (2006).
86. Ma, Z., Ma, J., Miao, Y., Li, Y. & Deng, R. H. ShieldFL: Mitigating Model Poisoning Attacks in Privacy-Preserving Federated Learning. *IEEE Transactions on Information Forensics and Security* **17,** 1639–1654 (2022).
87. Awan, S., Luo, B. & Li, F. *Contra: Defending against poisoning attacks in federated learning* in *Computer Security–ESORICS 2021: 26th European Symposium on Research in Computer Security, Darmstadt, Germany, October 4–8, 2021, Proceedings, Part I 26* (2021), 455–475.
88. Chang, S., Kim, S. D. & Kondo, G. Predicting default risk of lending club loans. *Machine Learning,* 1–5 (2015).
89. Richtárik, P. & Takáč, M. On optimal probabilities in stochastic coordinate descent methods. *Optimization Letters* **10,** 1233–1243 (2016).
90. Minka, T. *Estimating a Dirichlet distribution* 2000.
91. Liu, X. *et al.* Privacy-enhanced federated learning against poisoning adversaries. *IEEE Transactions on Information Forensics and Security* **16,** 4574–4588 (2021).
92. Shi, S., Hu, C., Wang, D., Zhu, Y. & Han, Z. Federated anomaly analytics for local model poisoning attack. *IEEE Journal on Selected Areas in Communications* **40,** 596–610 (2021).
93. Ganin, Y. *et al.* Domain-adversarial training of neural networks. *Journal of machine learning research* **17,** 1–35 (2016).
94. Liu, Z., Luo, P., Wang, X. & Tang, X. *Deep learning face attributes in the wild* in *Proceedings of the IEEE international conference on computer vision* (2015), 3730–3738.
95. Go, A., Bhayani, R. & Huang, L. Twitter sentiment classification using distant supervision. *CS224N project report, Stanford* **1,** 2009 (2009).
96. Xu, G., Meng, Y., Qiu, X., Yu, Z. & Wu, X. Sentiment analysis of comment texts based on BiLSTM. *Ieee Access* **7,** 51522–51532 (2019).
97. Ferrag, M. A., Friha, O., Hamouda, D., Maglaras, L. & Janicke, H. *Edge-IIoTset: A New Comprehensive Realistic Cyber Security Dataset of IoT and IIoT Applications: Centralized and Federated Learning* 2022.
98. Caldas, S. *et al.* Leaf: A benchmark for federated settings. *arXiv preprint arXiv:1812.01097* (2018).
99. Solanki, T., Rai, B. K. & Sharma, S. in *Federated Learning for IoT Applications* 157–167 (Springer, 2022).

100. Paszke, A. *et al.* Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems* **32** (2019).

101. Beutel, D. J. *et al.* Flower: A friendly federated learning research framework. *arXiv preprint arXiv:2007.14390* (2020).

102. Li, K. H., de Gusmão, P. P. B., Beutel, D. J. & Lane, N. D. *Secure aggregation for federated learning in flower* in *Proceedings of the 2nd ACM International Workshop on Distributed Machine Learning* (2021), 8–14.

103. Ding, C., Cao, X. (& Næss, P. Applying gradient boosting decision trees to examine non-linear effects of the built environment on driving distance in Oslo. *Transportation Research Part A: Policy and Practice* **110,** 107–117. ISSN: 0965-8564 (2018).

104. Ryffel, T. *et al.* A generic framework for privacy preserving deep learning. *arXiv preprint arXiv:1811.04017* (2018).

105. Liu, Y., Fan, T., Chen, T., Xu, Q. & Yang, Q. Fate: An industrial grade platform for collaborative learning with data protection. *The Journal of Machine Learning Research* **22,** 10320–10325 (2021).

106. Reina, G. *et al.* OpenFL: An open-source framework for Federated Learning. arXiv 2021. *arXiv preprint arXiv:2105.06413.*

107. Ludwig, H. *et al.* Ibm federated learning: an enterprise framework white paper v0. 1. *arXiv preprint arXiv:2007.10987* (2020).

108. Roth, H. R. *et al.* Nvidia flare: Federated learning from simulation to real-world. *arXiv preprint arXiv:2210.13291* (2022).

109. He, C. *et al.* Fedml: A research library and benchmark for federated machine learning. *arXiv preprint arXiv:2007.13518* (2020).

110. Pedregosa, F. *et al.* Scikit-learn: Machine learning in Python. *the Journal of Machine Learning Research* **12,** 2825–2830 (2011).

111. He, K., Zhang, X., Ren, S. & Sun, J. *Deep residual learning for image recognition* in *Proceedings of the IEEE conference on computer vision and pattern recognition* (2016), 770–778.

112. Ben Atia, O., Al Samara, M., Bennis, I., Abouaissa, A. & Lorenz, P. *EMDG-FL: Enhanced Malicious Model Detection based on Genetic Algorithm for Federated Learning* in *In IEEE Wireless Communications and Networking Conference (WCNC)* (2024).

113. Cover, T. & Hart, P. Nearest neighbor pattern classification. *IEEE transactions on information theory* **13,** 21–27 (1967).

114. Ferdush, J. *et al.* An enhanced image encryption technique combining genetic algorithm and particle swarm optimization with chaotic function. *International Journal of Computers and Applications* **43,** 960–967 (2021).

115. Cunningham, P. & Delany, S. J. K-nearest neighbour classifiers-a tutorial. *ACM computing surveys (CSUR)* **54,** 1–25 (2021).

116. Hard, A. *et al.* Federated learning for mobile keyboard prediction. *arXiv preprint arXiv:1811.03604* (2018).