Robust and Privacy-Preserving Federated Learning

Fatima Elhattab

INSTITUT NATIONAL DES SCIENCES APPLIQUÉES LYON

THÈSE DE DOCTORAT DE L'UNIVERSITÉ DE LYON
opérée au sein de
l'Institut National des Sciences Appliquées

École Doctorale 512
InfoMaths

Spécialité de doctorat: Informatique

Présentée et soutenue publiquement
par **Fatima Elhattab**

Robust and Privacy-Preserving Federated Learning

Devant le jury composé de :

Romain Rouvoy	Professeur, Université de Lille	**Rapporteur**
François Taïani	Professeur, Université de Rennes	**Rapporteur**
Aurelien Bellet	Chargé de Recherche, Université de Lille	**Examinateur**
Véronique Eglin	Professeure, INSA Lyon	**Examinatrice**
Sara Bouchenak	Professeure, INSA Lyon	**Directrice de thèse**

Laboratoire LIRIS
Adresse
Bâtiment Nautibus
Campus de la Doua
25 avenue Pierre de Coubertin
69622 Villeurbanne Cedex

École Doctorale InfoMaths
7, avenue Jean Capelle
69621 VILLEURBANNE Cedex

SIGLE	ECOLE DOCTORALE	NOM ET COORDONNEES DU RESPONSABLE
CHIMIE	**CHIMIE DE LYON** https://www.edchimie-lyon.fr Sec. : Renée EL MELHEM Bât. Blaise PASCAL, 3e étage secretariat@edchimie-lyon.fr	**M. Stéphane DANIELE** C2P2-CPE LYON-UMR 5265 Bâtiment F308, BP 2077 43 Boulevard du 11 novembre 1918 69616 Villeurbanne directeur@edchimie-lyon.fr
E.E.A.	**ÉLECTRONIQUE, ÉLECTROTECHNIQUE, AUTOMATIQUE** https://edeea.universite-lyon.fr Sec. : Stéphanie CAUVIN Bâtiment Direction INSA Lyon Tél : 04.72.43.71.70 secretariat.edeea@insa-lyon.fr	**M. Philippe DELACHARTRE** INSA LYON Laboratoire CREATIS Bâtiment Blaise Pascal, 7 avenue Jean Capelle 69621 Villeurbanne CEDEX Tél : 04.72.43.88.63 philippe.delachartre@insa-lyon.fr
E2M2	**ÉVOLUTION, ÉCOSYSTÈME, MICROBIOLOGIE, MODÉLISATION** http://e2m2.universite-lyon.fr Sec. : Bénédicte LANZA Bât. Atrium, UCB Lyon 1 Tél : 04.72.44.83.62 secretariat.e2m2@univ-lyon1.fr	**Mme Sandrine CHARLES** Université Claude Bernard Lyon 1 UFR Biosciences Bâtiment Mendel 43, boulevard du 11 Novembre 1918 69622 Villeurbanne CEDEX sandrine.charles@univ-lyon1.fr
EDISS	**INTERDISCIPLINAIRE SCIENCES-SANTÉ** http://ediss.universite-lyon.fr Sec. : Bénédicte LANZA Bât. Atrium, UCB Lyon 1 Tél : 04.72.44.83.62 secretariat.ediss@univ-lyon1.fr	**Mme Sylvie RICARD-BLUM** Institut de Chimie et Biochimie Moléculaires et Supramoléculaires (ICBMS) - UMR 5246 CNRS - Université Lyon 1 Bâtiment Raulin - 2ème étage Nord 43 Boulevard du 11 novembre 1918 69622 Villeurbanne Cedex Tél : +33(0)4 72 44 82 32 sylvie.ricard-blum@univ-lyon1.fr
INFOMATHS	**INFORMATIQUE ET MATHÉMATIQUES** http://edinfomaths.universite-lyon.fr Sec. : Renée EL MELHEM Bât. Blaise PASCAL, 3e étage Tél : 04.72.43.80.46 infomaths@univ-lyon1.fr	**M. Hamamache KHEDDOUCI** Université Claude Bernard Lyon 1 Bât. Nautibus 43, Boulevard du 11 novembre 1918 69 622 Villeurbanne Cedex France Tél : 04.72.44.83.69 hamamache.kheddouci@univ-lyon1.fr
Matériaux	**MATÉRIAUX DE LYON** http://ed34.universite-lyon.fr Sec. : Yann DE ORDENANA Tél : 04.72.18.62.44 yann.de-ordenana@ec-lyon.fr	**M. Stéphane BENAYOUN** Ecole Centrale de Lyon Laboratoire LTDS 36 avenue Guy de Collongue 69134 Ecully CEDEX Tél : 04.72.18.64.37 stephane.benayoun@ec-lyon.fr
MEGA	**MÉCANIQUE, ÉNERGÉTIQUE, GÉNIE CIVIL, ACOUSTIQUE** http://edmega.universite-lyon.fr Sec. : Stéphanie CAUVIN Tél : 04.72.43.71.70 Bâtiment Direction INSA Lyon mega@insa-lyon.fr	**M. Jocelyn BONJOUR** INSA Lyon Laboratoire CETHIL Bâtiment Sadi-Carnot 9, rue de la Physique 69621 Villeurbanne CEDEX jocelyn.bonjour@insa-lyon.fr
ScSo	**ScSo*** https://edsciencessociales.universite-lyon.fr Sec. : Mélina FAVETON INSA : J.Y. TOUSSAINT Tél : 04.78.69.77.79 melina.faveton@univ-lyon2.fr	**M. Bruno MILLY** Université Lumière Lyon 2 86 Rue Pasteur 69365 Lyon CEDEX 07 bruno.milly@univ-lyon2.fr

*ScSo : Histoire, Géographie, Aménagement, Urbanisme, Archéologie, Science politique, Sociologie, Anthropologie

Abstract

In today's rapidly evolving digital landscape, machine learning has become an indispensable and transformative force, as substantiated by extensive research studies. Its profound impact spans across diverse industries, offering groundbreaking solutions and innovations that have reshaped the way we interact with technology and make decisions. From recommendation systems enhancing content delivery on platforms to the presence of virtual personal assistants like Siri and Alexa, capable of understanding and responding to natural language commands, the applications of machine learning are both diverse and impactful. In domains like healthcare, it aids in disease diagnosis, while in finance, it fortifies fraud detection and risk assessment. This ubiquity of machine learning signifies not just a technological trend but a fundamental shift in problem-solving and decision-making approaches. However, this surge in data-driven innovation has raised a paramount concern - the protection of individuals' privacy and personal data. The General Data Protection Regulation (GDPR) exemplifies the heightened importance of data privacy in our modern era. As machine learning becomes increasingly intertwined with our daily lives, achieving a delicate balance between technological advancements and safeguarding individual privacy has become imperative. Moreover, addressing these concerns has given rise to the concept of privacy-preserving machine learning, with federated learning emerging as a pivotal technique, redefining collaborative machine learning by enabling multiple parties to build a shared model without sharing their raw data.

Federated Learning holds a lot of promise in the world of Machine Learning, allowing decentralized devices in edge computing systems to work together to train models. But, it's not all smooth sailing; there are some serious security and privacy issues to contend with. This research breaks down into two main parts, each dealing with these challenges in Federated Learning.The first part, which constitutes a critical aspect of our research, focuses on thwarting poisoning attacks. These malicious actions occur when unscrupulous clients attempt to sneak harmful tasks into federated models alongside their primary objectives, potentially compromising the integrity of the entire learning process. In response to this looming threat, we have developed *ARMOR*, a novel and sophisticated detection system. *ARMOR* leverages the power of GANs to meticulously scrutinize the data concealed within model updates. The second part of our research delves into safeguarding the privacy of individuals participating in Federated Learning, particularly from membership inference attacks. To address this challenge, we have introduced two mechanisms. The first one, *PASTEL* works tirelessly to enhance the resilience of Federated Learning systems against membership inference attacks. It achieves this by reducing the internal generalization gap, thereby minimizing the risk associated with data leakage between the information used for training and the data that the model has not seen during its training phase. The second privacy-focused approach, DINAR, is an ingenious solution that adds an extra layer of privacy protection to Federated Learning. DINAR operates by obscuring sensitive data within the model itself, effectively rendering it inaccessible to prying eyes. Furthermore, it employs intelligent gradient descent methods to ensure that the model remains not only privacy-conscious but also highly useful. These research objectives collectively aim to address security and privacy challenges and advance the field of federated learning.

Résumé

Dans le monde numérique en perpétuelle mutation d'aujourd'hui, l'apprentissage automatique est désormais une puissance essentielle et révolutionnaire, comme le démontrent de multiples recherches. Son impact profond s'étend à travers diverses industries, offrant des solutions et des innovations révolutionnaires qui ont remodelé la manière dont nous interagissons avec la technologie et prenons des décisions. Des systèmes de recommandation améliorant la diffusion de contenu sur les plateformes à la présence d'assistants personnels virtuels comme Siri et Alexa, capables de comprendre et de répondre à des commandes en langage naturel, les applications de l'apprentissage automatique sont à la fois diverses et impactantes. Dans des domaines tels que la santé, il contribue au diagnostic des maladies, tandis que dans la finance, il renforce la détection de la fraude et l'évaluation des risques. Cette ubiquité de l'apprentissage automatique signifie non seulement une tendance technologique, mais aussi un changement fondamental dans les approches de résolution de problèmes et de prise de décisions. Cependant, cette vague d'innovation axée sur les données a soulevé une préoccupation primordiale : la protection de la vie privée des individus et de leurs données personnelles. Le Règlement général sur la protection des données (RGPD) illustre l'importance accrue de la protection des données à l'ère moderne. À mesure que l'apprentissage automatique s'intègre de plus en plus dans notre vie quotidienne, trouver un équilibre délicat entre les avancées technologiques et la protection de la vie privée individuelle est devenu impératif. De plus, L'attention portée à ces préoccupations a donné naissance au concept de l'apprentissage automatique préservant la vie privée, avec l'apprentissage fédéré émergeant comme une technique cruciale, redéfinissant l'apprentissage automatique collaboratif en permettant à plusieurs parties de construire un modèle partagé sans partager leurs données brutes.

L'apprentissage fédéré suscite de grandes attentes dans le domaine de l'apprentissage automatique, en permettant à des dispositifs décentralisés au sein de systèmes informatiques périphériques de collaborer pour créer des modèles. Cependant, ce n'est pas toujours simple, car des préoccupations de sécurité et de confidentialité doivent être prises en considération. Cette étude se divise en deux parties principales, chacune se penchant sur ces défis dans le contexte de l'apprentissage fédéré. La première partie se concentre sur la prévention des attaques d'empoisonnement. Ces actes malveillants surviennent lorsque des utilisateurs mal intentionnés tentent d'introduire des tâches nuisibles dans les modèles fédérés en plus de leurs objectifs légitimes, compromettant ainsi potentiellement l'intégrité de l'ensemble du processus d'apprentissage. En réponse à cette menace imminente, nous avons développé *ARMOR*, un système de détection innovant et sophistiqué. *ARMOR* tire parti de la puissance des réseaux génératifs antagonistes (GAN) pour scruter minutieusement les données cachées au sein des mises à jour du modèle. La deuxième partie de notre recherche se penche sur la protection de la vie privée des individus participant à l'apprentissage fédéré, en particulier contre les attaques d'inférence d'appartenance. Pour relever ce défi, nous avons introduit deux mécanismes. Le premier, *PASTEL*, s'efforce de renforcer la résilience des systèmes d'apprentissage fédéré contre les attaques visant à déduire l'appartenance des données. Il y parvient

en réduisant l'écart interne de généralisation, minimisant ainsi le risque de divulgation de données entre les informations utilisées pour la formation et celles que le modèle n'a pas vues pendant sa phase d'apprentissage. La deuxième approche axée sur la confidentialité, DINAR, fonctionne en obscurcissant les données des couches sensibles du modèle lui-même, les rendant ainsi inaccessibles. De plus, il utilise des méthodes de descente de gradient pour garantir une précision élevée du modèle. Ces objectifs de recherche visent collectivement à relever les défis en matière de sécurité et de confidentialité et à faire progresser le domaine de l'apprentissage fédéré.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Context and Motivation

In today's fast-paced digital landscape, machine learning has emerged as an indispensable and transformative force, as substantiated by numerous research studies. Its profound impact spans across diverse industries, promising groundbreaking solutions and innovations. Machine learning has permeated virtually every facet of our lives, reshaping the way we interact with technology and make decisions. From recommendation systems fine-tuning content delivery on platforms to the presence of virtual personal assistants like Siri and Alexa [85], capable of understanding and responding to natural language commands, the applications of machine learning are as diverse as they are impactful. In the domain of healthcare, machine learning aids in disease diagnosis [1], while in finance, it fortifies fraud detection and risk assessment [105]. The ubiquity of machine learning is not just a technological trend but a fundamental shift in how we approach problem-solving and decision-making.

However, this surge in data-driven innovation has brought forth a paramount concern - the protection of individuals' privacy and personal data. The General Data Protection Regulation (GDPR) symbolizes how crucial data privacy has become in our contemporary world [42]. With the growing integration of machine learning into our everyday experiences, it is now essential to find a careful equilibrium between technological progress and protecting the personal privacy of individuals. Moreover, addressing these concerns has given rise to the concept of privacy-preserving machine learning. This innovative approach ensures that sensitive data remains confidential while still enabling the development of powerful machine learning models [40]. Within this context, federated learning emerges as a pivotal technique, redefining the landscape of collaborative machine learning by allowing multiple parties to build a shared model without sharing their raw data [71].

1.2 Research Objectives

Federated Learning represents a promising paradigm in Machine Learning, enabling collaborative model training among decentralized devices in edge computing systems. This approach allows multiple participants, or clients, to train a model collectively without the need to share their individual data directly. Instead, clients share their local model parameters with an FL server, which then combines these updates to create a global model. This global model is subsequently shared back

with the clients. FL has found successful applications in various domains, including autonomous driving, speech recognition, smartphone word prediction, activity recognition, and financial fraud detection. Nonetheless, despite its merits, Federated Learning exhibits susceptibility to a range of client-side attacks due to its user-centric nature [119].

Our research is divided into two main thrusts, each addressing critical security and privacy challenges in the context of Federated Learning. In this section, we outline our objectives and contributions for both aspects of our research.

1.2.1 Countering Poisoning Attacks for Robust Federated Learning

This segment of our research focuses primarily on data and model poisoning attacks, specifically targeting the resilience of Federated Learning [7, 104]. Adversaries aim to introduce a harmful task into the federated model alongside its main task. This insidious task assigns arbitrary labels to input data, often triggered by specific criteria. For instance, an attacker could circumvent a facial recognition-based authentication system by mislabeling their images to gain unauthorized access.

Detecting these poisoning attacks within federated learning proves to be an intricate challenge since participants transmit model updates to the FL server, concealing their raw training data. Consequently, the FL server possesses limited insights into user behavior, impeding the detection of malicious participants. A variety of mechanisms have been proposed in contemporary research to enable the identification of such attacks. While these mechanisms adopt diverse approaches to detection, they all hinge on scrutinizing the geometric characteristics of model updates submitted by participants to the FL server.

In the first part of our thesis, we have demonstrated that attackers can still elude these detection methods by crafting model updates that closely mimic benign participants' updates. Consequently, we introduce ARMOR, a novel GAN-based attack detection system that shifts the focus towards analyzing the information embedded in model updates, rather than merely monitoring their geometric attributes.

We assess the performance of ARMOR using well-established image recognition datasets and deep neural network architectures. Our results reveal that ARMOR outperforms existing state-of-the-art mechanisms in mitigating highly aggressive poisoning attack scenarios, underlining its efficacy as a potent defense against such threats.

1.2.2 Countering Membership Inference Attacks for Privacy-Preserving Federated Learning

Despite the strides made in preserving data privacy through decentralized data handling in Federated Learning (FL), recent research has revealed vulnerabilities, rendering FL systems susceptible to privacy attacks. Specifically, our focus centers on membership inference attacks (MIAs), where a malicious participant seeks to ascertain whether a specific data sample was utilized in the FL model training process.

To bolster privacy in FL, several defense mechanisms have been proposed, including techniques based on cryptography, secure multiparty computation (SMC), and differential privacy (DP). DP, which can be implemented either on the client-side (Local Differential Privacy) or server-side (Central Differential Privacy), has been a primary focus. While DP-based methods can mitigate membership inference attacks to some extent, they often come at the cost of reduced model accuracy and increased computational overhead.

In the second part of our research, we introduce two novel approaches. First, we present *PASTEL*, a privacy-preserving mechanism designed to enhance FL systems' resilience against membership inference attacks. *PASTEL* employs a novel multi-objective learning function. It simultaneously minimizes model loss, optimizes model accuracy through adaptive gradient descent, and narrows the generalization gap between member and non-member data. Recent studies have indicated that sensitive information can reside in specific layers of neural networks and be inferred from their gradients. Thus, *PASTEL*'s primary aim is to minimize this internal generalization gap during FL model training, effectively safeguarding private information and reducing the success rate of MIAs.

Secondly, we introduce DINAR, a fine-grained privacy-preserving FL method tailored to counter membership inference attacks. DINAR operates at the client-side of FL, safeguarding both the global FL model and individual client models. It identifies the most privacy-sensitive layer in neural networks, inspired by recent research findings. DINAR obfuscates this critical layer in the client model before transmitting updates to the FL server. Consequently, the aggregated model produced by the FL server includes an obfuscated version of this layer. Upon receiving the protected global model from the server, the client restores its local privacy-sensitive layer, integrating it into its version of the global model before utilizing it for predictions. To enhance the model's utility, DINAR employs adaptive gradient descent, dynamically adjusting the learning rate for each dimension during optimization, given the high-dimensional nature of neural network problems.

1.3 Summary of Contributions

The contributions of this thesis are three folds : (c1) : *ARMOR*: A mitigation mechanism against poisoning attacks in federated learning. (c2) : *PASTEL*: an optimization-driven approach to mitigate membership inference attacks. (c3) : DINAR: obfuscation-based defense mechanism against membership inference attacks. The following section outlines the various publications, communications, and software prototypes associated with these contributions.

1.3.1 Publications and Communications

- Fatima Elhattab, Sara Bouchenak, Rania Talbi, Vlad Nitu. Robust Federated Learning for Ubiquitous Computing through Mitigation of Edge-Case Backdoor Attacks. ACM Interact. Mob. Wearable Ubiquitous Technol., 6(4), 162:1-162:27, 2022. Proceedings of UbiComp 2023. (Ranked A*)

- Fatima Elhattab. Towards Mitigation of Edge-Case Backdoor Attacks in Federated Learning. Best Poster Award @ EuroSys Rennes, Avril, 2022.
- Fatima Elhattab, Cédric Boscher, Sara Bouchenak. PASTEL: Privacy Preserving Federated Learning in Edge Computing. ACM Interact. Mob. Wearable Ubiquitous Technol. Proceedings of UbiComp 2024. (Ranked A*), under submission.
- Fatima Elhattab, Cédric Boscher, Sara Bouchenak. DINAR: Towards Fine-Grained Privacy Preserving Federated Learning. ICLR conference 2024. (Ranked A), under submission.
- F. Elhattab, R. Talbi, S. Bouchenak, V. Nitu. Towards Mitigating Poisoning Attacks in Federated Learning. Conférence francophone d'informatique en Parallélisme, Architecture et Système (ComPAS 2021), Lyon (Remote), July 2021.
- F. Elhattab, S. Bouchenak. Mitigating Membership Inference Attacks in Federated Learning. Conférence francophone d'informatique en Parallélisme, Architecture et Système (ComPAS 2023), Annecy, July 2023.

1.3.2 Software Prototypes

The following software prototypes were developed during this thesis:

- *ARMOR*: A Python library designed to detect poisoning attacks in federated learning through the generation of class representatives using GANs.

  https://github.com/robust-fl/armor

- *PASTEL*: A toolkit that mitigates membership inference attacks in federated learning enviroment based on multi-objective optimization.

  Public link coming soon

- DINAR: A toolkit that mitigates membership inference attacks in federated learning enviroment based on layers obfuscation.

  Public link coming soon

1.4 Thesis Roadmap

This thesis is thoughtfully organized into two distinct parts, each addressing critical aspects of federated learning. Part I, titled "Robustness in Federated Learning," begins with Chapter 2, where we delve into the essential background and explore related work in the domain of robust federated learning. This chapter lays the foundation by explaining federated learning concepts and shedding light on the unique challenges it faces. Chapter 3 introduces ARMOR, a pioneering defense mechanism aimed at mitigating poisoning attacks in federated learning, outlining the overarching objectives and core design principles. Chapter 4 provides a deep dive into the empirical side of ARMOR, presenting a comprehensive account of its implementation and rigorous experimental evaluation. We assess ARMOR's effectiveness in

various scenarios, including its resilience against poisoning attacks and a comparison with other federated learning defense mechanisms.

Part II, titled "Privacy-Preservation in Federated Learning," starts with Chapter 5, offering a comprehensive exploration of privacy threats inherent to federated learning. This chapter not only delineates these threats but also surveys existing privacy-preserving mechanisms to provide a broader context. In Chapter 6, titled "DINAR: Fine-Grained Mitigation of Membership Inference Attacks in Federated Learning," we present DINAR, an innovative approach for addressing membership inference attacks. We outline its research objectives and core design principles, discussing how it employs fine-grained mechanisms such as model obfuscation, personalization, and adaptive training to enhance privacy in federated learning. Chapter 8 introduces PASTEL, a novel solution tailored to address the pressing issue of membership inference attacks within the federated learning framework. It elucidates the research objectives, system model, and design principles that underpin PASTEL's approach. Chapter 9 delves into the empirical evaluation of PASTEL, detailing the experimental setup and results that gauge its efficacy in preserving privacy in federated learning.

Finally, in Chapter 10, aptly titled "Conclusion and Perspectives," we bring the thesis to a close by summarizing the key findings and contributions in both robustness and privacy-preservation aspects of federated learning. Furthermore, we offer perspectives on potential future research directions in these domains, providing a comprehensive overview of the entire research journey undertaken in this thesis.

Part I

Robustness in Federated Learning

Chapter 2

Background and Related Work on Robust Federated Learning

2.1 Background on Federated Learning

Federated Learning (FL) is a novel paradigm in machine learning where instead of centralizing data on a single server, training takes place across a network of decentralized devices or clients. These clients, such as smartphones, edge devices, or IoT devices, collaboratively contribute to the training process while keeping their data locally stored and private. The central idea is to enhance the global model's performance by leveraging the diverse and extensive data available across these devices, without compromising individual user privacy [71, 16].

Federated Learning has gained attention due to its ability to address challenges associated with data privacy, network latency, and resource constraints. It is particularly useful in scenarios where data is sensitive, abundant, and distributed, such as healthcare, finance, and IoT applications. By leveraging a diverse range of data sources, federated learning enables the creation of robust and accurate machine learning models while respecting user privacy and data ownership.

In this chapter, we will delve into the realm of robustness in Federated Learning. We will discuss the fundamental concept of FL, its significance in addressing data privacy and decentralized machine learning challenges, and the vulnerabilities it introduces, particularly in the form of poisoning attacks. We will also explore the two main categories of poisoning attacks in FL: untargeted attacks, which aim to disrupt the overall model performance, and targeted attacks, which have specific and often malicious objectives. Furthermore, we will investigate various defense mechanisms and strategies developed to mitigate these poisoning attacks and enhance the security and robustness of FL systems. Finally, we will highlight some open research issues and ongoing efforts to address the evolving threat landscape in FL.

Federated Learning is a decentralized machine learning framework where multiple clients work together to train a common global model with high accuracy, as illustrates in Figure 2.1. This research focuses on training Deep Neural Networks , which serve as the fundamental architecture for various complex tasks. To formalize this, we consider a cross-silo FL setup with K clients, denoted as $C = \{C_1, C_2, \ldots, C_k\}$, and each client C_k possesses its local training dataset D_k. The primary goal of cross-silo FL is to solve an optimization problem to obtain the optimal global parameter W:

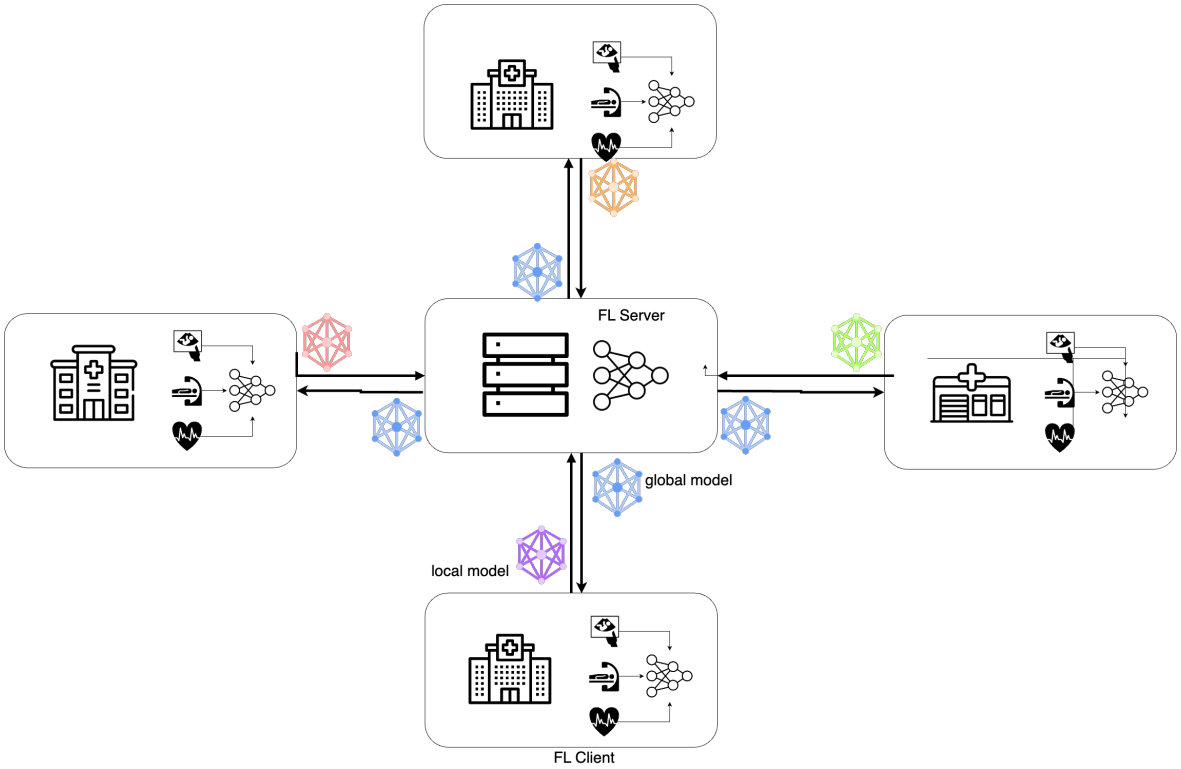


FIGURE 2.1: Centralized Federated Learning Pipeline

$$\min_{W} F(W) = \min_{W} \sum_{k=1}^{K} \frac{|D_k|}{|D|} F(W, D_k) \tag{2.1}$$

where $|D_k|$ is the sample size of client C_k, and $|D| = \sum_{k=1}^{K} |D_k|$. $F(W, D_k)$ is the local objective of C_k defined by:

$$\min_{W} F(W, D_k) = \min_{W} \frac{1}{|D_k|} \sum_{(x_i, \bar{y}_i) \in D_k} L(W; (x_i, \bar{y}_i)) \tag{2.2}$$

where $(x_i, \bar{y}_i)$ is a training sample, x_i and $\bar{y}_i$ are the corresponding feature vector and the ground-truth label vector, respectively. $L(\cdot;\cdot)$ is a user-specified loss function, such as Mean Squared Error and Cross-entropy.

In order to find the optimal parameters for Eq. 2.1, the server initializes the global model parameter, and then the server and all clients collaboratively perform the DNN training, which mainly includes three phases:

1. **Global Model Broadcasting:** The server broadcasts the current global model parameter $W = \{W^{(l)}\}_K^{l=1}$ to all clients for local model training.

2. **Local Model Training:** Once receiving the global model parameter $W = \{W^{(l)}\}_K^{l=1}$, each client C_k computes local gradients $\nabla F(W, D_k)$ by running the stochastic gradient descent (SGD) algorithm on the local training dataset D_k. Generally, for each training sample $(x, \bar{y}) \in D_k$, the training process includes two steps: forward propagation and backward propagation.

3. **Global Model update:** After collecting $\nabla F(W, D_k)$ from each client, the server combines them and modifies the current model parameter W for the subsequent iteration. More precisely, using the learning rate η, the new parameter is calculated as follows:

$$W \leftarrow W - \frac{\eta}{|D|} \sum_{k=1}^{K} \frac{1}{|D_k|} \nabla F(W, D_k) \quad (5)$$

Subsequently, the server shares the updated parameter W with all clients for the next iteration.

2.2 Background on Poisoning Attacks in Federated Learning

Many studies have highlighted the inherent vulnerabilities of centralized machine learning systems, showcasing their susceptibility to a range of malicious attacks [32, 92]. While Federated Learning represents a promising approach to enhance data privacy in ML, it is not impervious to threats. This is because in FL, not only do individual workers possess access to model parameters, but they can also exert influence over their respective data during the training phase. This thesis delves into a specific facet of these vulnerabilities: FL poisoning attacks.

FL poisoning attacks can be broadly categorized into two groups: untargeted attacks [32, 92] and targeted attacks [92, 7, 12, 104]. The primary objective of untargeted attacks is to undermine the overall performance of the model on its primary task. On the other hand, targeted attacks, also known as backdoor attacks, are geared towards a more specific and sinister purpose – manipulating the model to compromise a particular class or feature within the data.

In the subsequent sections, we will thoroughly explore each type of attack, shedding light on their mechanisms, implications, and potential countermeasures.

2.2.1 Untargeted Poisoning Attacks

In untargeted poisoning attacks in federated learning, the main objective is to compromise the overall performance of the federated learning model in its main task. These updates, while appearing similar to benign updates in terms of statistical measures like mean and variance, possess the capability to evade existing defense mechanisms and disrupt the accuracy of the model. An illustrative example of an untargeted attack is provided in [32]. where a group of malicious sybil workers collaboratively sends model updates that appear similar to honest updates in terms of statistical measures such as mean and variance. However, these malicious updates are carefully crafted to bypass existing defense mechanisms and ultimately lead to a deviation in the model's accuracy.

The key idea in this work revolves around the creation of malicious local models in each iteration of Federated Learning. These models are designed to induce what the authors refer to as a "directed deviation" of the global model. In simpler terms, this means altering the direction of the global model parameters in a way

that is contrary to what would naturally occur without attacks. To achieve this, the attackers generate these malicious model updates by solving mathematical optimization problems. These optimization problems take into account the impact of two state-of-the-art poisoning defense mechanisms: Multi-Krum [13] and Trimmed Mean [115]. These mechanisms are integrated into the attacker's strategy to make their malicious updates more effective and harder to detect.

In summary, untargeted poisoning attacks in Federated Learning aim to subtly manipulate the model's updates to disrupt its overall performance. These attacks use advanced optimization techniques and exploit weaknesses in existing defense mechanisms to achieve their objectives.

2.2.2 Targeted Poisoning Attacks

Targeted poisoning attacks in Federated Learning are sophisticated strategies employed by malicious actors to undermine the security and privacy of the FL process. These attacks are designed with specific, often harmful objectives in mind, and they can be broadly categorized into two primary types: Data Poisoning Attacks and Model Poisoning Attacks. These adversarial tactics aim to compromise the integrity of the FL model or introduce biased behavior for malicious purposes.

Targeted poisoning attacks are especially concerning in FL due to the decentralized nature of the training process. Unlike traditional machine learning approaches where data is centralized, FL operates by training models on distributed devices while keeping data local. This decentralized nature makes FL particularly vulnerable to poisoning attacks as adversaries can exploit the trust placed in participating devices.

Data Poisoning Attacks

Data poisoning attacks in FL involve adversaries tampering with the training data before it is used to construct local model updates that are subsequently sent to the FL server [36, 101]. The primary goal of these attacks is to introduce misclassifications for specific inputs into a target class. For example, in the context of an image classification task, a malicious user may alter the labels associated with training images before using them to generate local model updates [36].

Data poisoning attackers may choose to manipulate the labels of only a subset of the training data, such as modifying labels for images of small green cars, or they may target an entire class, like the "car" class [36]. Importantly, since the training data resides privately on individual workers' devices, this type of attack is challenging to detect directly by the FL server, as the server cannot inspect the labels of the training data held by the workers.

These types of attacks have been evaluated in various scenarios, including image classification tasks and textual data, demonstrating their potential to disrupt the FL process. Data poisoning attacks can lead to the training of models with biased representations, potentially causing significant harm in applications where fairness and non-discrimination are critical.

Model Poisoning Attacks

In contrast to data poisoning attacks, model poisoning attacks directly manipulate the model updates instead of tampering with the training data [7, 104]. In this type of attack, adversaries aim to achieve their malicious objectives by modifying the model updates in a way that brings the global model very close to a predefined poisoned model, denoted as w^*.

To conduct model poisoning attacks, attackers typically require knowledge about the number of workers participating in the FL training round and insight into the aggregation algorithm used by the FL server. With this information, attackers can craft model updates strategically to steer the global model towards the poisoned model w^*.

These attacks assume that adversaries have access to an already poisoned version of the model. They manipulate the model updates in such a manner that the global model converges towards the poisoned model as closely as possible. Model poisoning attacks can have severe consequences, including the compromise of the FL system's security, privacy breaches, and the introduction of biased models into the federated model aggregation process.

One noteworthy model poisoning attack is proposed in [7], capable of achieving up to 100% accuracy for the attacker's task. Additionally, a more aggressive attack strategy known as "constrain-and-scale" is introduced, which can evade well-known robust aggregation algorithms like Multi-Krum [13] and Norm Clipping (NDC) [98]. To further complicate detection, authors in [104] apply Projected Gradient Descent (PGD) on the attackers' local model updates, ensuring that these updates do not significantly deviate from the global model. This is accomplished by projecting the attackers' models into a small ball centered around the global model from the previous iteration.

Model poisoning attacks pose significant threats to the integrity and security of FL systems, as they can subvert the collaborative learning process and introduce compromised models into the federated model aggregation, potentially causing widespread harm.

Backdoor Attacks

A backdoor in FL manipulates a subset of training data to poison the data and the model, by injecting adversarial triggers such that the FL model trained on the tampered dataset makes an arbitrarily (or a targeted) incorrect prediction on the test set with the same embedded trigger [7]. For instance, in the case of a malware detection system, an attacker who wants to evade the detection would carefully add a watermark which serves as the attack trigger in a set of malicious applications of his choice, and changes their labels from the malicious applications class to the benign applications class. Edge-case backdoor attacks are particular backdoors where the attacker uses a trigger that is underrepresented, or unlikely to be part of the training set of other workers' data [104]. Thus, the effect of edge-case backdoors can lead to models mispredicting classification subtasks, especially those that may be underrepresented in the training set. For instance, several recent reports show that neural networks can mispredict inputs of underrepresented minority individuals [43], or

describe edge-case inputs that have been a point of serious concern for the safety of autonomous vehicles [43].

Figure 2.2 illustrates the case of an attacker that introduces edge-case backdoors into a FL-based decentralized and automatic traffic sign recognition system. Here, the attacker (*i.e.*, a FL client) introduces a visual pattern P^* to the top left corner of its set of traffic sign images, in such a way that these images are misclassified and labeled with a wrong target label C_{target}, *e.g.*, a stop sign misclassified as a speed limit sign. Thus, the attacker produces the D_{attack} dataset that is used for local training at the attacker side, to carry model replacement. Such an attack is described here [7].

Roughly speaking, given the current global FL model collectively produced by a set of workers, the attacker substitutes the new global model with a malicious model w_{attack^*}. More precisely, the attacker creates a model that does not look anomalous, and replaces the global model after averaging with the other benign participants' models. To prevent the backdoor from being forgotten, techniques such as slowing down the learning rate during the attacker's training can be used to improve the persistence of the backdoor in the global model. Furthermore, to prevent the malicious model from being easily detected if it significantly diverges, projected gradient descent (PGD) is applied to project the malicious model centered around the last global model [104].

2.3 Related Work on Robust Federated Learning

The decentralized nature of FL introduces vulnerabilities, particularly poisoning attacks, that can undermine the integrity and security of the global model. As such, the development of robust mitigation techniques is imperative to safeguard FL systems from adversarial influences.

In this section, we delve into the extensive body of research aimed at fortifying Federated Learning against various forms of poisoning attacks. These attacks include backdoor attacks, which manipulate the model's behavior on specific inputs, and inference attacks, which aim to extract sensitive information from the global model. To counter these threats, researchers have devised a multitude of defense mechanisms, each with its own strengths and limitations.

these defense mechanisms can be categorized into three main groups: perturbation-based approaches, aggregation-based approaches, and selection-based approaches. Perturbation-based methods introduce controlled noise into the data contributed by FL participants, effectively diluting the impact of malicious data while preserving genuine contributions. Aggregation-based methods focus on how model updates from participants are combined to form the global model, employing techniques like trimmed mean and norm clipping. Selection-based methods aim to identify and exclude potentially malicious contributions from FL participants, utilizing strategies such as trust scoring and model accuracy monitoring.

Throughout this section, we delve into each category, providing insights into notable defense mechanisms and their effectiveness in mitigating poisoning attacks. Additionally, we discuss open research issues in the field, recognizing that as FL evolves, so too must our defenses against novel and sophisticated adversarial strategies. Finally, we conclude this section with a summary of the state of the art in robust Federated Learning, highlighting the ongoing research efforts to secure FL systems

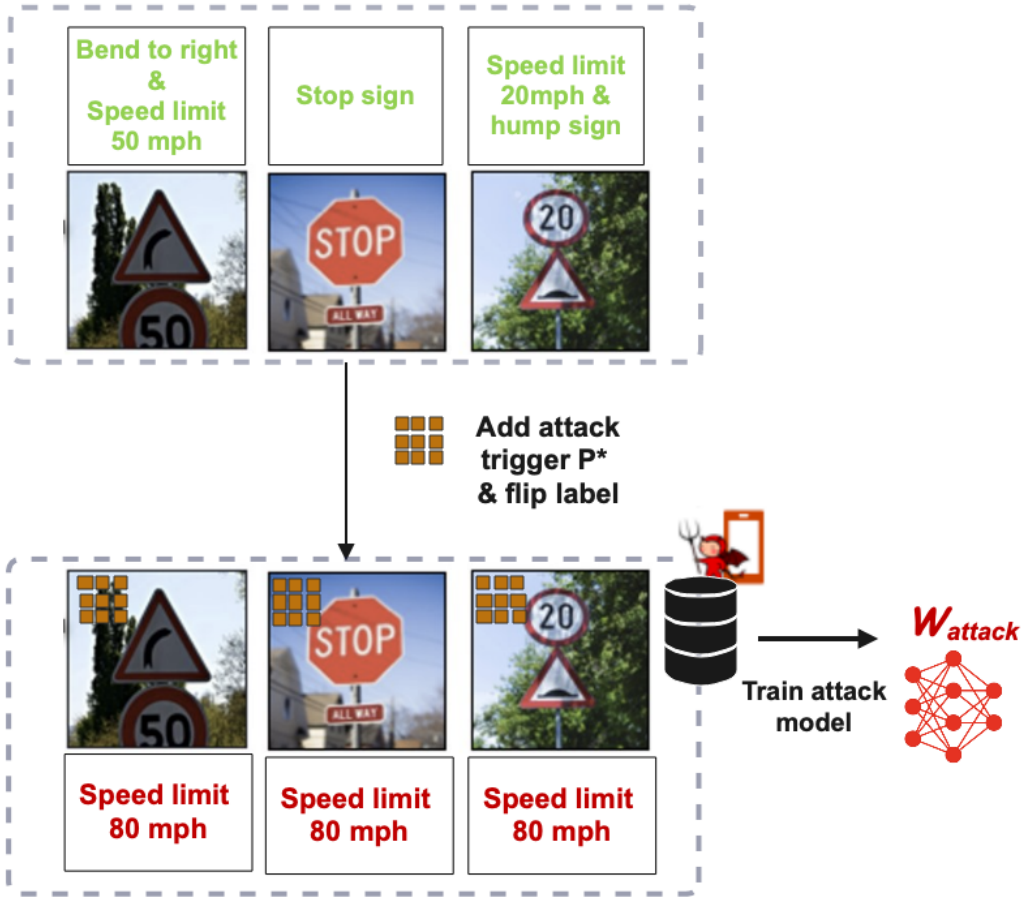


FIGURE 2.2: Edge-case backdoor in automatic traffic sign image classification. The trigger used by the attacker to introduce the edge-case backdoor is unlikely to be part of benign clients' data

in an ever-evolving threat landscape. Mitigation techniques of poisoning attacks in FL are summarized in Table 2.1.

2.3.1 Perturbation-Based Mechanisms

Perturbation-based mechanisms represent a category of defense strategies in Federated Learning that aim to enhance robustness against poisoning attacks by introducing controlled noise into the data contributed by each participant [113]. This noise serves to mitigate the impact of potentially malicious data while still allowing legitimate contributions to be integrated into the global model.

One pioneering work in this domain, conducted by the authors of [82], explored the application of both Local Differential Privacy (LDP) and Central Differential Privacy (CDP) to counteract backdoor and inference attacks within FL systems. Their research revealed the efficacy of perturbation-based methods in reducing the success of these attacks. Here are key findings from their experiments:

Local Differential Privacy (LDP): LDP was shown to be a valuable defense mechanism against both backdoor and inference attacks. When applied with an epsilon value (ϵ) of 3 to an FL setup involving 2,400 participants using the EMNIST dataset,

LDP achieved significant mitigation. Backdoor attack accuracy was reduced from 88% to 10%, while utility decreased from 92% to 62%. This demonstrated LDP's ability to substantially hinder the success of attacks while maintaining a reasonable level of utility.

Central Differential Privacy (CDP) CDP emerged as another robust defense mechanism against backdoor attacks. In the same EMNIST scenario with an ϵ value of 3, CDP outperformed LDP in terms of reducing backdoor attack accuracy. It lowered the accuracy from 88% to an even lower 6%, showcasing its effectiveness. Importantly, CDP accomplished this while preserving a higher level of utility compared to LDP. The utility only decreased from 90% to 78%, which was notably higher than the utility reduction observed with LDP (ϵ=3), which stood at 62%.

These results underscore the potential of perturbation-based mechanisms, specifically LDP and CDP, in fortifying FL systems against poisoning attacks. They demonstrate that with careful parameter tuning, it is possible to significantly hinder adversarial attempts while retaining the practical utility of the FL model.

2.3.2 Selection-Based Mechanisms

Selection-based mechanisms focus on identifying and excluding potentially malicious contributions from FL participants. Here are some notable selection-based mechanisms:

Multi-Krum

Krum, introduced by Author et al. [13], leverages the robustness of the median as a measure of the central tendency of model updates in Federated Learning. The central idea behind Krum is to identify and eliminate potentially malicious or outlying model updates contributed by FL participants, ensuring that only honest updates are used to compute the global model. Here's how Krum works:

1. **Model Updates Selection:** In each FL round, Krum collects model updates from all participating clients. These updates represent the changes made to the model based on each client's local data.

2. **Distance Calculation:** Krum calculates the pairwise distances between each client's model update and the updates of all other clients. This distance calculation is typically based on metrics such as Euclidean distance.

3. **Sorting by Distance:** The distances are then ranked in ascending order for each client. This ranking helps identify which clients' updates are the closest to the updates of other clients, making them candidates for potential exclusion.

4. **Identifying Central Clients:** Krum selects a predefined number of central clients based on their rankings. The number of central clients chosen depends on the desired robustness level, with a lower number making the defense more resilient against adversarial attacks.

5. **Aggregation:** The selected central clients' model updates are aggregated to compute the global FL model.

Multi-Krum is an enhanced variant of the Krum defense mechanism introduced by Author et al. [13]. Multi-Krum combines the resilience properties of Krum with the convergence speed of model averaging, making it a powerful defense against adversarial attacks in Federated Learning. Here's how Multi-Krum extends the Krum approach:

1. **Model Updates Selection:** Similar to Krum, Multi-Krum begins by collecting model updates from all participating FL clients in each training round.
2. **Distance Calculation:** Multi-Krum calculates pairwise distances between the model update of each client and the updates of all other clients. This distance calculation employs metrics like Euclidean distance.
3. **Sorting by Distance:** The computed distances are sorted in ascending order for each client, as in Krum. This sorting helps identify close neighbors for potential exclusion.
4. **Identifying Central Clients:** Multi-Krum, like Krum, selects a set of central clients based on their rankings. These central clients are chosen to form a robust subset for model aggregation.
5. **Aggregation:** Where Multi-Krum differs from Krum is in its aggregation strategy. Instead of aggregating only the central clients' updates, Multi-Krum combines the central clients' updates with the updates from all clients. This combination leverages the robustness of Krum while benefiting from the convergence speed of averaging.

Multi-Krum thus provides a balance between robustness and convergence speed, making it an effective defense mechanism against various forms of poisoning attacks in FL.

DPLM

DLMP, proposed by Author et al. [32], assumes that the Federated Learning server possesses advance knowledge of a validation dataset. This dataset is employed to monitor fluctuations in model accuracy during each training round. If a substantial decline in accuracy is observed, the model generated in that round is considered potentially compromised and excluded.

FLTrust

FLTrust, introduced by Author et al. [18], is a defense mechanism designed to enhance the security and reliability of Federated Learning systems by incorporating trust-based scoring for participating FL clients. This mechanism assumes that the FL server maintains a root dataset and assigns trust scores to individual clients based on their historical behavior and performance. The key concept behind FLTrust is to attribute trustworthiness to clients and use these trust scores as a factor in aggregating model updates.

2.3.3 Aggregation-Based Mechanisms

Aggregation-based mechanisms focus on how model updates from FL participants are combined to form the global model. Here are some notable aggregation-based mechanisms:

Trimmed Mean

Trimmed Mean, as introduced by Author et al. [115], is an aggregation technique used in Federated Learning to enhance robustness against poisoning attacks. It calculates the aggregated model update while excluding extreme values, thus mitigating the influence of malicious updates.

The Trimmed Mean is calculated as follows:

$$\text{Trimmed Mean} = \frac{1}{N-2k} \sum_{i=k+1}^{N-k} W_i \tag{2.3}$$

where:

- N is the total number of clients participating in the FL round.
- k is the trim ratio, representing the percentage of extreme values to be removed from both ends of the sorted list of updates.
- W_i represents the model update from client i.

NDC (Norm-based Defense and Clipping)

NDC, or Norm-based Defense and Clipping, is an aggregation algorithm designed to counteract poisoning attacks in Federated Learning, as presented in [7]. NDC leverages the assumption that attackers often send model updates with larger norms than honest workers, using this property to detect and mitigate adversarial contributions. NDC employs two primary defense mechanisms: norm clipping and Gaussian noise addition. Norm clipping restricts the magnitude of model updates received from FL participants, effectively reducing the influence of malicious workers while preserving honest contributions. The addition of Gaussian noise further obscures adversarial contributions, making it challenging for attackers to manipulate the model effectively. However, it's essential to strike a balance between security and utility, as these defense mechanisms may lead to a degradation in model quality. Practitioners must carefully tune NDC's parameters to maintain a usable global model. NDC represents a valuable tool in the ongoing effort to enhance the security and resilience of Federated Learning systems against evolving adversarial threats.

2.4 Open Research Issues

Defending against edge-case backdoors presents a formidable challenge in the context of Federated Learning due to their subtle and highly targeted nature. These attacks are crafted to be inconspicuous, making them particularly insidious. They operate by manipulating the model's behavior in a manner that goes unnoticed by conventional defense mechanisms. Unlike conventional poisoning attacks that may

TABLE 2.1: Comparison of robust FL methods

Robust-consious category	Protection method	Model utility	Computation overhead	Compatible with secure aggregation	Does not require validation dataset
Perturbation-based methods	LDP [82]	✓	✓	✗	✓
	Februus [29]	✓	✓	✗	✓
	DP-FedAvg [72]	✓	✓	✗	✓
	WDP [98]	✓	✓	✗	✓
Selection-based methods	Multi-Krum [13]	✓	✓	✗	✓
	DPLM [32]	✓	✓	✗	✗
	FLTrust [18]	✓	✓	✗	✗
Aggregation-based methods	Trimmed Mean [115]	✓	✓	✗	✓
	NDC [98]	✗	✓	✗	✓
Our Method	*ARMOR*	✓	✗	✓	✓

introduce glaring anomalies in the model updates, edge-case backdoors remain hidden amidst the vast and diverse data streams of FL. This stealthiness is exacerbated by attackers' use of advanced techniques like Projected Gradient Descent (PGD), enabling them to create model updates that closely mimic the global FL model while containing subtle, harmful modifications. The inherent challenge lies in the fact that most robust FL aggregators rely on evaluating the geometric distance between clients' model updates. However, this approach proves ineffective against edge-case backdoors, as these attacks leave behind no geometric footprints that stand out amidst the noise of legitimate contributions. Furthermore, the lack of clear anomalies and the need for domain-specific knowledge make detection even more elusive, as these attacks are often designed to blend seamlessly into the data distribution.

2.5 Summary

This chapter provides a comprehensive overview of robustness in Federated Learning. We have explored the core concept of FL and its relevance in addressing data privacy and decentralized machine learning challenges. Furthermore, we have delved into the critical topic of poisoning attacks in FL, categorizing them into untargeted and targeted attacks, each with its own distinct objectives and mechanisms. To defend against these attacks, we have discussed a range of defense mechanisms and strategies, including perturbation-based, selection-based, and aggregation-based approaches. Finally, we have acknowledged the existence of open research issues. In the next section, we will present our contribution to enhancing Federated Learning robustness, addressing some of the limitations of existing methods to further fortify FL systems against evolving threats.

Chapter 3

ARMOR: Mitigating Poisoning Attacks in Federated Learning

3.1 ARMOR Objectives

Federated learning (FL) is a promising paradigm that is gaining grip in the context of privacy-preserving machine learning (ML) for ubiquitous computing systems. Thanks to FL, several data owners called *clients* (*e.g.*, mobile devices in cross-device FL, or organizations in cross-silo FL) can collaboratively train a model on their private and decentralized data, without having to send their raw data to external service providers. To this end, clients iteratively update a global model using their local training data, and send only their model updates to a central party called the *server* that orchestrates the training process. The FL server aggregates the received model updates to produce a new version of the global model, which is, in turn, distributed to the clients. FL was rapidly adopted in several thriving ubiquitous computing applications such as healthcare [90], self-driving cars [117], home automation [51], next-word prediction [112], etc. Although FL has improved the privacy of ML by decentralizing the data and the learning process, a line of recent literature shows that FL systems are vulnerable to poisoning attacks [32, 92, 7, 12, 104]. Here, malicious clients introduce backdoor attacks to corrupt the global FL model so that it produces a prediction that is inappropriate for the task at hand, such as misclassifying a no entry traffic sign as a speed limit sign. We specifically put our focus on *edge-case backdoor attacks* which target input data points, that while generally correctly classified by a FL model, are rare and either underrepresented, or unlikely to be part of the training or test data [104]. The effect of edge-case backdoors can lead to models mispredicting classification subtasks, especially those that may be underrepresented in the training set. For instance, several recent reports show that neural networks can mispredict inputs of underrepresented minority individuals [43], or describe edge-case inputs as a serious concern for the safety of autonomous vehicles [43].

Robust FL has been extensively studied, and several defense mechanisms to counter backdoors have been proposed, based on various techniques such as robust aggregation [13, 115, 88], norm clipping [7], or differential privacy [73]. However, recent studies demonstrate that edge-case backdoors are hard to detect by such defense mechanisms, and are among the most difficult poisoning attacks to tackle [104, 39]. Indeed, some protection mechanisms assume the existence of a test set to uncover attacks [18, 32]. However, edge-case backdoors rely on out-of-distribution features

available at the attacker side, and which are, by design, unlikely to be part of test set data available on the FL server.

In addition, edge-case backdoors often rely on techniques such as projected gradient descent (PGD) [104], to project the malicious model updates' vector to the last version of the global model, so that it looks similar to a benign model update. Therefore, these attacks also deceive protection mechanisms which rely on the analysis of the geometric shape of model updates [88, 13, 98, 115].

Furthermore, in order to protect against honest-but-curious FL servers, FL protocols usually rely on secure aggregation which is a secure multi-party computation protocol allowing to compute the sum of client model updates while preventing the server from examining individual updates [16]. However, many existing FL defense mechanisms require analyzing individual model updates [18, 13, 115, 7], and are therefore incompatible with secure aggregation, which makes them more vulnerable to privacy leakage [75]. In this chapter, we propose *ARMOR*, a novel FL defense mechanism. As far as we know, *ARMOR* is the first FL defense that tackles edge-case backdoors *and* is compatible with secure aggregation, *without* requiring the knowledge of prior of data.

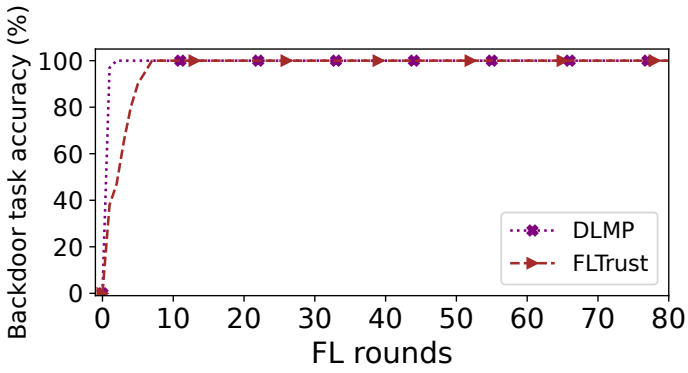
3.2 On the Difficulty of Edge-Case Backdoors in Federated Learning

3.2.1 Existing Defense Mechanisms Are Not Effective Against Edge-Case Backdoors.

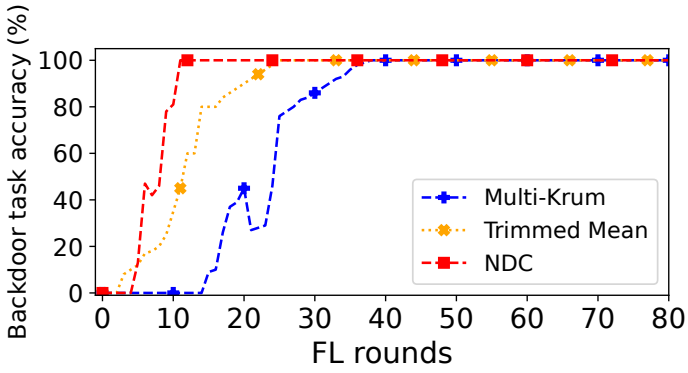
We evaluated several existing FL defenses with edge-case backdoor attacks. The experiments were conducted with the FashionMNIST dataset and a four-layer convolutional neural network, and with an edge-case backdoor occurring at each FL round. The implementation details of the edge-case backdoor attack, and the underlying evaluation environment are described in §9.1. First, 3.1(a) presents the results of two defense mechanisms that assume the existence of a validation dataset, namely DLMP [32] and FLTrust [18]. We observe that none of these mechanisms allows to counter edge-case backdoors, and the backdoor task accuracy reaches 100% after only a few rounds. 3.1(b) presents the behavior of defenses that do not assume prior knowledge of a validation dataset, such as Multi-Krum [13], Trimmed Mean [115], and NDC [98]. These mechanisms are more efficient than DLMP and FLTrust, nevertheless, after some rounds the backdoor task accuracy increases up to 100%.

3.2.2 Defense Mechanisms Incompatible With Secure Aggregation Are More Vulnerable to Privacy Leakage.

Many state-of-the-art FL defenses are not compatible with secure aggregation, because they need to analyze individual workers' model updates [13, 98, 115, 18]. This results in privacy issues and information leakage, such as a higher vulnerability to membership inference attacks [83]. Membership inference is the ability to



(a) Mechanisms with prior knowledge



(b) Mechanisms without prior knowledge

FIGURE 3.1: Effectiveness of edge-case backdoors with existing FL defense mechanisms. Attacks start from the first round, and occur every round. Considered cases: (a) defense mechanisms with a prior knowledge of a validation dataset on the FL server; (b) defenses without a prior knowledge of a validation dataset

determine if a given data record was part of the model's training dataset. This is more problematic in case of FL defense systems that are incompatible with secure aggregation, where membership inference also allows to determine exactly to which worker (*i.e.*, data owner) the data record belongs to. An interesting study evaluated the effectiveness of membership inference in different FL settings, and with several datasets and models [83]. In 3.2, we report on the results of two cases, on the one hand, FL with secure aggregation and where membership inference is applied on the global model, and on the other hand, FL without secure aggregation and where membership inference is applied on clients' local model updates. We see that the latter case is more vulnerable to information leakage by up to 16%, in addition to revealing data owner's identity.

3.3 System Model

In the following, we describe the underlying threat model and defense model.

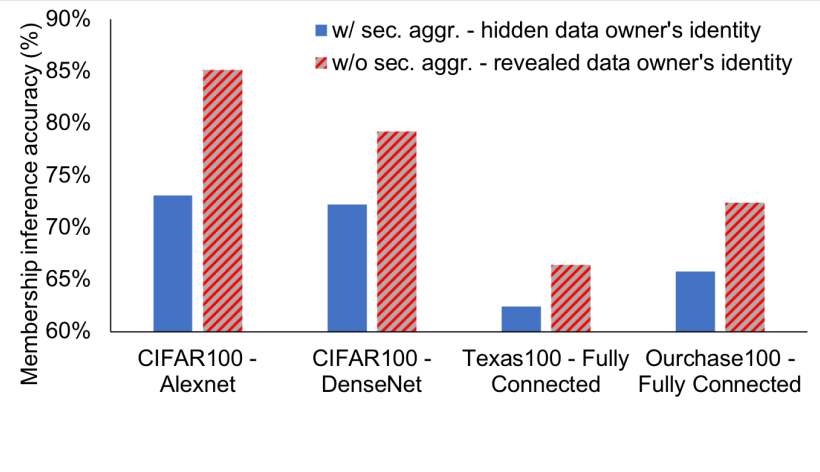


FIGURE 3.2: Privacy leakage through membership inference, with and without secure aggregation, with different datasets and model architectures [83]. Considered cases: (i) CIFAR100 with Alexnet neural network; (ii) CIFAR100 with DenseNet neural network; (iii) Texas100 with a fully connected neural network; (iv) Ourchase100 with a fully connected neural network

3.3.1 Threat Model

Attacker's Objectives

The goal of the malicious worker is to make the global FL model W_t misclassify a subset S of particular data samples (x_i, y_i) to a target class C_{target}. The data samples S have particular features P^* that we refer to as the attack trigger. The FL classification task is defined as follow:

$$W_t(x_i) = \begin{cases} C_{target}, & \forall x_i \in S \\ y_i, & \forall x_i \notin S \end{cases} \tag{3.1}$$

The adversarial task is to optimize the attacker model W_{attack}, with a Loss function $\mathcal{L}_{attack}$, so that the samples (x_i, y_i) are classified with the new adversarial label C_{target} if $x_i \in S$:

$$\min \left(\sum_{x_i \in S} \mathcal{L}_{\text{attack}}(W_{\text{attack}}(x_i), C_{\text{target}}) + \sum_{x_i \notin S} \mathcal{L}_{\text{attack}}(W_{\text{attack}}(x_i), y_i) \right) \tag{3.2}$$

For instance, in the case of a traffic sign recognition system embedded in autonomous vehicles, a malicious participant can cause accidents with serious concern by carefully adding a watermark which serves as the attack trigger P^* in a set of images of his choice S, and changes their labels from, for instance, the no entry sign class to the speed limit sign class. Such a backdoor trigger is very effective in changing the classification decision [39]. Moreover, in edge-case backdoors, the attack trigger P^* is underrepresented, or unlikely to be present in benign clients' data

and, thus, difficult to detect. 3.1 provides a summary of notations used throughout the chapter.

TABLE 3.1: Notations

Notation	Description
m	Number of selected clients at a given round
W_t	Global model at round t
W_{attack}	Attacker's malicious model
D_{attack}	Attacker's dataset
P^*	Data pattern overleayed by the attacker A^* into poisoned data
C_{target}	Backdoor's target class
D_t	Data representatives test set at round t
Dis_{kt}	Discriminator of class C_k at round t
Gen_{kt}	Generator of class C_k at round t
$\mathcal{L}_{attack}$	Attacker loss function
$\mathcal{L}(D_t, w_t)$	Testing loss obtained when evaluating D_t using the model w_t
$\mathcal{L}_{Gen}(x, C_{fake})$	*ARgan* generator loss function
$\mathcal{L}_{Dis}(x, y)$	*ARgan* discriminator loss function
C_{fake}	Fake images label
C_{real}	real images label
T	*ARgan* training epoch number

Attacker's Capabilities

As in many previous works [7, 9, 12, 32, 45, 110], we assume that the attacker can access the global model that is sent by the FL server in each round, and that it can directly manipulate the training data on the malicious devices by adding the attack trigger P^* and carrying a label flipping to the target class C_{target}. It is also able to train an attack model W_{attack} over the dataset D_{attack}.

Thus, the attacker can generate a model update that aims to replace the global model with the attacker's model W_{attack}.

Attacker's Knowledge

As any other worker, the attacker has access to previous versions of the FL model. It has also access to its attack training dataset D_{attack} to carry poisoning, and train an attack model w_{attack}.

3.3.2 Defense Model

Defense Objectives

We aim to design a FL defense mechanism that achieves resilience against malicious FL clients, without sacrificing the global FL model's quality. Specifically, we aim to fulfil the two following properties:

- ***Resilience.*** Our defense mechanism should ensure that the global FL model is unlikely to predict the attacker-chosen target labels for the attacker-chosen target samples. Thus, the resilience of the defense mechanism should be as high as possible. Its metric is presented in §9.1.
- ***Utility.*** Our defense mechanism should preserve the classification accuracy of the global model in the presence of adversaries performing poisoning attacks. In particular, the defense mechanism aims to learn a global model under attacks that is as accurate as possible as the global model learnt by the FL system without attacks and no defense mechanism. Its metric is presented in §9.1.

Defender's Capabilities

The defense against FL attacks is performed on the FL server side. It has the capability to compute a poisoning indicator, based on which it decides whether to take into account the new aggregated model update if considered as sane. Otherwise, the FL server reduces its impact through a specific aggregation-based mitigation technique.

Defender's Knowledge

We consider a honest-but-curious server. Our defense mechanism that runs on the server does not have access to the clients' raw training data, and it does not audit the clients' model updates. In contrast to existing FL defense systems [13, 115], our defense mechanism does not need to know the number of malicious clients, nor the number of clients involved in a FL round. Furthermore, unlike other existing works [32, 18], our defense mechanism does not need a validation dataset to carry attack mitigation. As the FL server, the defense mechanism has access to the global FL model at different rounds, and to the newly aggregated FL model built with clients' model updates.

3.4 *ARMOR* Design Principles

In the following, we first present an overview of the proposed *ARMOR* defense method, before introducing the Generative Adversarial Networks that underly *ARMOR*. We then describe *ARMOR*'s two main components: *ARgan* and *MORpheus*.

3.4.1 Overview of *ARMOR*

We propose *ARMOR*, a novel FL defense mechanism that counters powerful client-side poisoning attacks such as edge-case backdoor attacks, without breaking secure aggregation guarantees, nor having access to private real data samples to carry

model inspection. *ARMOR* addresses the threat model introduced in §3.3.1, and has the defense objectives presented §3.3.2. The overall architecture of *ARMOR* is described in 3.3, with two main components: *ARgan* and *MORpheus*. *ARgan* is used to generate a synthetic dataset based on model updates, which is further leveraged by *MORpheus* to provide proper mitigation against poisoning attacks.

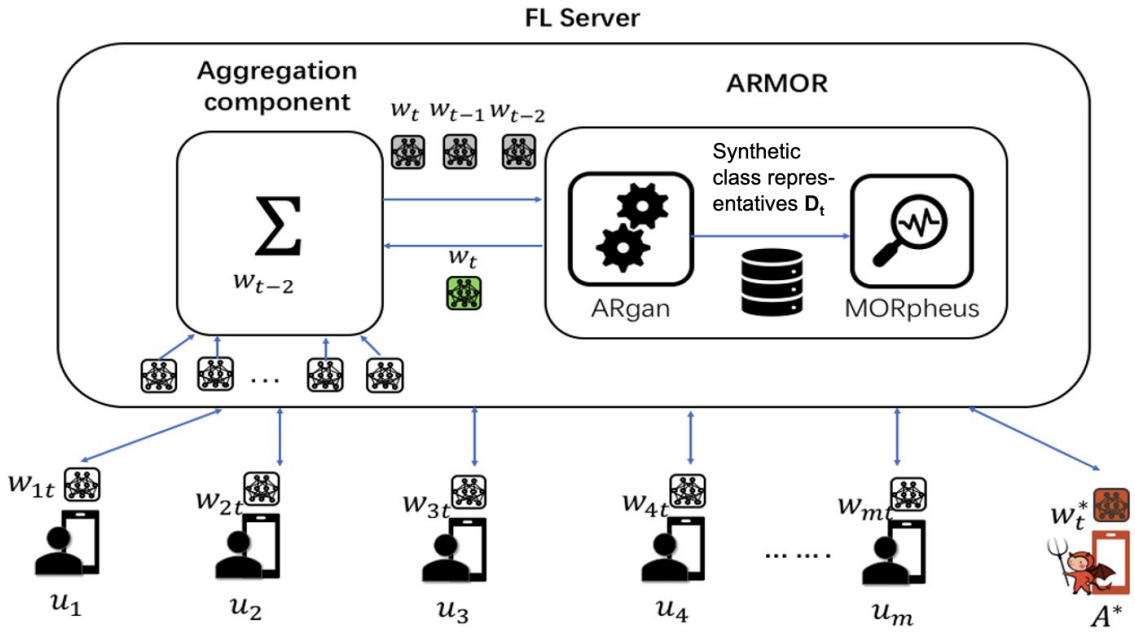


FIGURE 3.3: Overview of *ARMOR* architecture. The attacker is a FL client that trains a model on its poisoned data and feeds the FL model with its poisoned local model. *ARMOR*'s first component, *ARgan*, generates class representatives from the FL model. *ARMOR*'s second component, *MORpheus*, monitors loss variations to mitigate the backdoors

ARMOR's defense mechanism does not make any assumptions neither on the proportion of attackers in the system nor on their data distribution. The insight behind *ARMOR* is as follows. Let B be a backdoor task that aims to misclassify the data samples holding a particular data pattern P^* from a source class C_{source} to a target class C_{target}. Let us consider that the the model W_t (at round t) is poisoned with such a backdoor B. The poisoned class representatives of C_{target} generated from W_t would be misclassified by previous non-poisoned model (*e.g.*, W_{t-1}). Here, when auditing a model W_t, *ARMOR* monitors the difference between the loss obtained when feeding class representatives to this model W_t and the loss when feeding the representatives to the models of the s previous rounds $\{W_{t-1}, ..., W_{t-s}\}$. If the difference is higher than a given threshold vector, the current model is considered to be corrupted, and *ARMOR* applies a mitigation technique to reduce the impact of the new model updates.

In order to generate class representatives, *ARMOR* relies on a set of Generative Adversarial Networks (GANs) [38], that are trained on the FL server side based on the model updates received from the FL workers. GANs are a type of ML networks composed of two neural network models, a Generator (*Gen*) and a Discriminator (*Dis*) that contest with each other in a zero-sum game. The generator aims to build

a model that generates fake inputs which are as realistic as possible, while the discriminator attempts to distinguish whether the generated inputs are real or fake. The performance of both *Gen* and *Dis* is improved by playing the adversarial game. The discriminator's output is both used to improve the latter, but also backward-propagated through (*Gen*) to improve its capability to mimic real data. The competition between the generator and the discriminator ends at Nash equilibrium, when the discriminator is unable to distinguish fake samples from the real ones.

On the other hand, the discriminator has the objective of distinguishing between the fake data of the generative model and the real data.

The steps for training a GAN are as follows:

(i) Fake images generation. The Generator is queried to generate a batch of fake images $F^{(t)}$.

(ii) Fake image-based loss computation. The generated fake images $F^{(t)}$ are fed to the discriminator to compute the loss $L(p^{(t)}, C_{fake})$, where $p^{(t)}$ is the prediction output obtained by the discriminator when feeding $F^{(t)}$ to it, that is $p^{(t)} = Dis_{tk}(F^{(t)})$ which is backward-propagated to compute the gradients for fake data ΔW_{fake}.

(iii) Sampling of Real images. A batch of real data $R^{(t)}$ is sampled from the training dataset.

(iv) Real image-based loss computation. $R^{(t)}$ are fed to the discriminator to compute the loss $L(p^{(t)}, C_{real})$ which is backward-propagated to compute the gradients for real data ΔW_{real}.

(v) Discriminator update. The discriminator is updated with the sum of the two sets of gradients $\Delta W_{real} + \Delta W_{fake}$.

(vi) Generator update. The fake data is once again fed to the discriminator to compute the loss $L(p^{(t)}, C_{real})$ which this time is backward-propagated through the generator, to improve its capacity to mimic real data based on the discriminator's output. The goal of the Generator is to generate images that look like real ones, so its objective is also to minimize the loss on the real image class of the Discriminator.

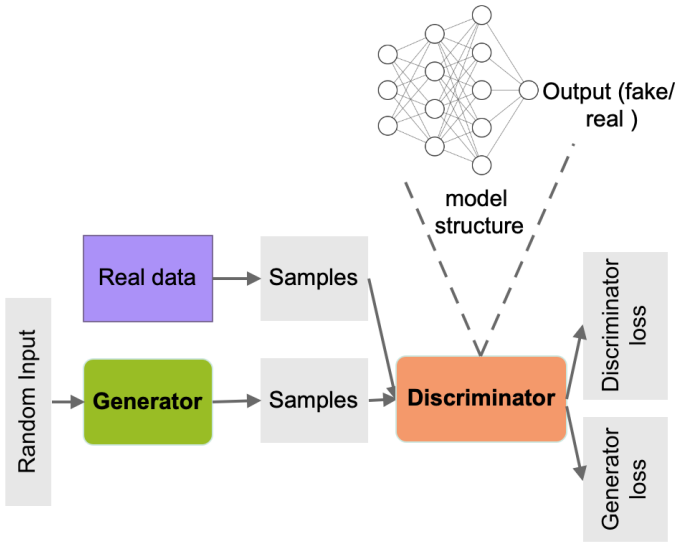
Finally, the competition between the generator and the discriminator ends at Nash equilibrium, when the discriminator is unable to distinguish fake samples from the real ones.

Roughly speaking, the goal of a GAN is to predict the features of data samples given a label instead of predicting a label based on input data's features. (The above is true only when the GAN mode collapse) A GAN consists of two models, the first model is called a *Generator* (*Gen*), and the second is a *Discriminator* (*Dis*). As a consequence, the two models have contradictory objective functions and compete with each other. The model G generates data points that try to bypass the model D while the model D tries to identify when he receives fake data from G.

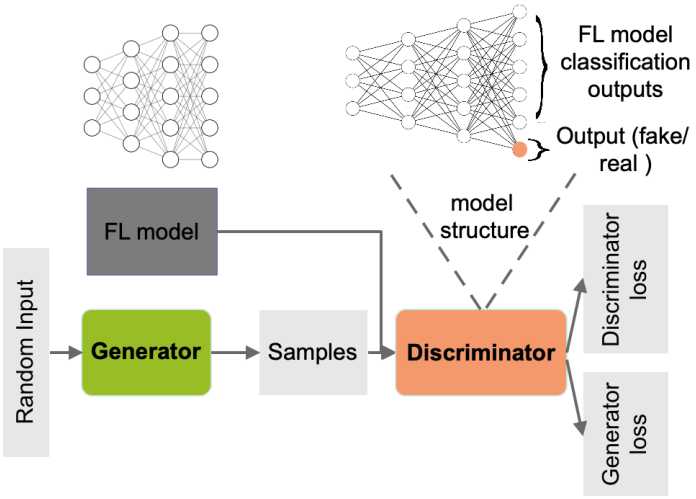
3.4.2 *ARgan*: *ARMOR* 's Generative Adeversarial Networks

The vanilla GAN architecture presented before can not be directly used to uncover potential attacks in Federated Learning, since the FL server does not have access to a real dataset. However, since the FL global model was trained with real data, the intuition behind *ARMOR* is to replace the discriminator's gradients computed on

real data by artificial gradients computed based on the FL global model. We call this new GAN architecture *ARgan*.



(a) GAN structure



(b) *ARgan* structure

FIGURE 3.4: Comparison between *ARgan* structure and regular GAN structure. A regular GAN's discriminator is fed with real data samples. In *ARgan*, no real data samples are needed. The FL model is augmented with an additional output in the last layer and the same architecture is used for the *ARgan*'s discriminator.

ARgan vs Vanilla GAN

The objective of the first component of *ARMOR* is to produce a dataset of class representatives that contain the backdoor in order to be used by the second component to detect the attack. Knowing that by definition the images that contain a backdoor are held only by the attacker, the server cannot assume the existence of these images.

The classical architecture of a vanilla GAN consists in considering real data as input which are used by the discriminator to differentiate real samples from false ones. Although GAN is able to learn the general data distribution and generate diverse images of the dataset, it remains limited to what exists in the training data [10]. Hence the new *ARgan* architecture that we propose, where instead of real data,*ARgan* considers the FL model as input, which is trained using data that contain the backdoor, making it possible to generate representative classes that contain the backdoor.

Similar to a vanilla GAN, an *ARgan* consists of a generator and a discriminator. The difference between the structure of these two types of GANs is described in Figure 3.4. In a vanilla GAN (Figure 3.4(b)), the networks are trained based on fake images labeled 0, and real images labeled 1. However, *ARMOR* does not assume the existence of a dataset of real images. Instead, *ARgan* relies on the FL model which was trained on real data. The discriminator model is built by considering the global FL model and extending it with an additional output to distinguish real data from fake data. For example, in the case of a classification task with 10 classes, the last layer of the model is a linear layer that contains 10 outputs. Here, the model structure is extended with an additional output, thus, resulting in a model with 11 outputs. Even though the discriminator contains classification outputs, the architecture of ARGAN is not the same as a cGAN's architecture, since each ARGAN is trained to generate representatives of a given class.

Algorithm 1: *MORpheus* mitigation mechanism

Input: Model at current round W_t
Output: Models at previous rounds $W_{t-1} \dots W_{t-s}$

foreach *iteration* t **do**
  $D_t \leftarrow ClassRepresentatives(W_t)$
  Feed the class-representatives to W_t, $W_{i-t} \dots W_{t-s}$
  $y_t \leftarrow W_t(D_t), y_{t-1} \leftarrow W_{t-1}(D_t) \dots y_{t-s} \leftarrow W_{t-s}(D_t)$
  Compute the testing loss for $W_t, W_{t-1} \dots W_{t-s}$
  $L_t \leftarrow L(y_t, k), L_{t-1} \leftarrow L(y_{t-1}, k)$
  $L_{t-s} \leftarrow L(y_{t-s}, k)$
  if $Attack\_Monitoring(W_t, W_{t-1} \dots W_{t-s})$ **then**
    Multiply the local model update by θ

The adversarial training of an *ARgan* instance is similar to the vanilla GAN, it involves training both the discriminator and the generator model in min-max adversarial training.

The generator is trained the same way as in a Vanilla GAN, i.e. we generate m noise samples $z^{(1)}, ..., z^{(m)}$, and then we calculate the loss based on the discriminator prediction. The only difference is the initialization of the generator model Gen_{kt} with the generator model of the previous epoch $Gen_{k(t-1)}$. thus avoiding the collapse mode which occurs when the generator and the discriminator do not have the same level of knowledge, and as the discriminator is initialized with the FL model

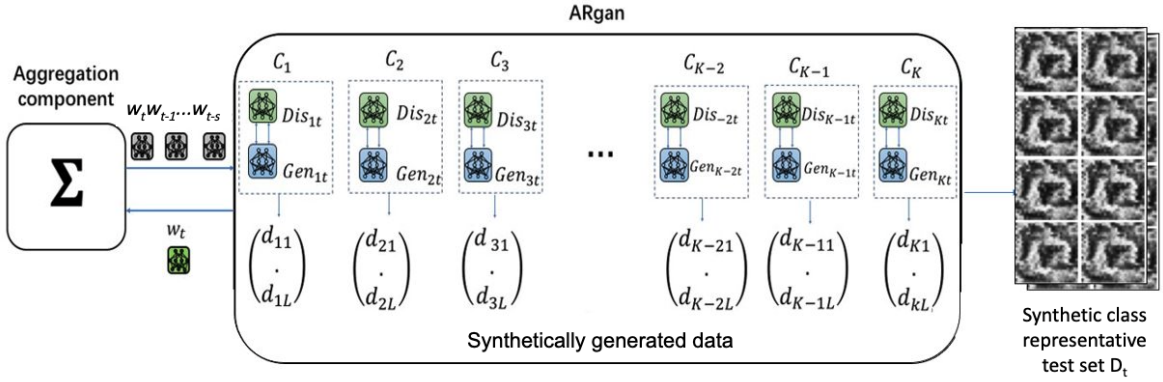


FIGURE 3.5: Overview of *ARgan*. It produces synthetic class representative set D_t, including benign data samples as well as backdoor samples

therefore the generator is initialized as shown in Algorithm 2, line 4. The generator loss function is defined as follow:

$$\mathcal{L}_{Gen} = \frac{1}{m}\sum_{i=1}^{m} Dis_{kt}(Gen_{kt}(z^{(i)})) \tag{3.3}$$

On the other hand, the training of the discriminator is not exactly the same, their differences are explained in Algorithm 2. Here, the lines in black represent the common code path between the vanilla GAN and an *ARgan* instance, while the lines in blue are specific to vanilla GAN, and the lines in red are specific to *ARgan*. Instead of computing updates of the discriminator on real data (lines 6-8), the *ARgan* instance uses the aggregated model update of the current FL round to update its discriminator(line 9). The FL model was trained using workers' real data that are not accessible to the FL server. This allows incorporating knowledge on workers' training data (that was potentially tampered with by attackers) in the *ARgan* instance. The discriminator loss function is defined as follow:

$$\mathcal{L}_{Dis} = \frac{1}{m}\sum_{i=1}^{m}\left[\mathcal{L}(Dis_{kt}(z^{(i)}) + Dis_{kt}(Gen_{kt}(z^{(i)})))\right] \tag{3.4}$$

Figure 3.5 presents an overview of *ARgan*. At a FL round t, for each class C_k an *ARgan* instance is trained in several (*i.e.*, T) iterations, with its discriminator Dis_{kt} and its generator Gen_{kt}. Therefore, the *ARgan*'s loss function can be defined as,

$$\min_G \max_D \mathcal{L}(D,G) = \frac{1}{m}\sum_{i=1}^{m}\left[\log(Dis_{kt}(z^{(i)}) + Dis_{kt}(Gen_{kt}(z^{(i)}))) + log(1 - Dis_{kt}(Gen_{kt}(z^{(i)})))\right] \tag{3.5}$$

Thus, *ARgan* instances allow to generate data samples D_t that includes benign data samples as well as backdoor samples.

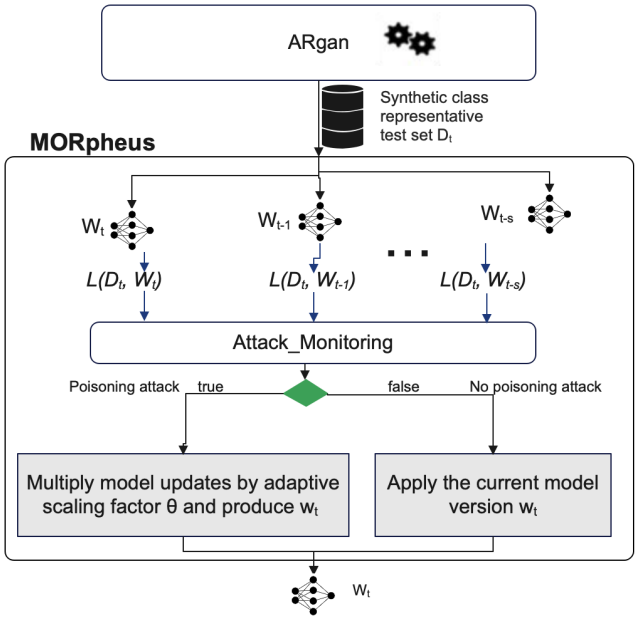


FIGURE 3.6: Overview of *MORpheus*. It uses the synthetic class representative dataset produced by *ARgan*, and monitors loss variations to mitigate backdoors

3.4.3 *MORpheus*: *ARMOR*'s Backdoor Mitigation Mechanism

The attack mitigation in *ARMOR* is performed by its *MORpheus* component, which is described in Figure 3.6, and detailed in Algorithm 9. *MORpheus* uses the testing set D_t produced by *ARgan*, which consists of class representatives for each class C_k. *MORpheus* feeds this testing set D_t to the current model W_t, and to the models of the s previous FL rounds $\{W_{t-1}, ..., W_{t-s}\}$.

Then, the testing loss is computed for each one of these model versions. As shown in Eq (3.6), if the loss exceeds a given threshold γ_i, the updates are considered to be malicious.

$$Attack\_Monitoring\ (W_t, \{W_{t-1}, .., W_{t-s}\}) = \begin{cases} true, & \text{if } (\frac{\mathcal{L}(D_t,W_t) - L(D_t,W_{t-1})}{max(L(D_t,W_t),L(D_t,W_{t-1}))} > \gamma_1 \\ & \text{and } \frac{L(D_t,W_t) - L(D_t,W_{t-2})}{max(L(D_t,W_t),L(D_t,W_{t-2}))} > \gamma_2 \\ & ... \\ & \text{and } \frac{L(D_t,W_t) - L(D_t,W_{t-s})}{max(L(D_t,W_t),L(D_t,W_{t-s}))} > \gamma_s) \\ false, & \text{otherwise} \end{cases} \tag{3.6}$$

Adaptive Attack Mitigation. In a pessimistic policy, attack mitigation can simply ignore the malicious updates, with a factor $\theta = 0$ that negates the effect of these updates on the global model. Another option is to consider that θ is inversely proportional to the loss, *i.e.*, the higher the loss is, the lower the scaling factor is and, thus, the lower the impact on the global model is. More precisely, θ is defined in Eq (3.7) as an exponential function, that is inversely proportional to the loss L_t, c being positive real value. We chose to rely on an exponential function instead of a linear function

Algorithm 2: Vanilla GAN (blue) vs *ARgan* (red)

Input: Model at current round w_t and class k
Output: Class representatives D_t

for e *in* $\{1..T\}$ **do**
  Generate a random noise vector $X^{(t)} \leftarrow$ Random()
  $Gen_{kt} \leftarrow Gen_{kt-1}$
  Get fake data $F^{(t)} \leftarrow Gen_{kt}(X^{(t)})$
  Feed $F^{(t)}$ to the discriminator $p^{(t)}_{fake} \leftarrow Dis_{kt}(F^{(t)})$
  Compute gradients ∇W_{fake} based on $\mathcal{L}_{Gen}(p^{(t)}_{fake}, C_{fake})$
  Feed real data $R^{(t)}$: $p^{(t)}_{real} \leftarrow Dis_{kt}(R^{(t)})$
  Compute gradients ∇W_{real} based on $\mathcal{L}_{Gen}(p^{(t)}_{real}, C_{real})$
  Update Dis_{kt} with $w_t + \nabla W_{fake}$
  Update Dis_{kt} with $\nabla W_{real} + \nabla W_{fake}$
  Update Gen_{kt} by computing $L(p^{(t)}, C_{real})$
  if $test\_accuracy(F^{(t)}, w_t) > \gamma$ **then**
    break
Output the class-representatives $D_t \leftarrow F^{(t)}$

because the former penalizes more large losses, and there can be small variations in the loss even in attack-free scenarios.

$$\theta = exp(-c * L_t) \tag{3.7}$$

Furthermore, *MORpheus*' sliding window size s depends on the attack aggressiveness. For instance, mitigating single-shot attacks requires a sliding window size of 2 since these attacks take effect immediately. In contrast, attacks that are slowly incorporated within the model through multiple rounds require a larger sliding window size for model inspection.

3.5 Summary

In this section, we introduced *ARMOR*, a novel method for countering poisoning attacks. Unlike existing approaches, *ARMOR* does not rely on analyzing the geometric characteristics of model updates to detect poisoning. Instead, it focuses on the informational content of these updates. The basic idea behind *ARMOR* is to employ a group of generative-adversarial networks to create synthetic test data. This synthetic data is then used to track changes in model loss, which helps identify instances of poisoning. In the following chapter, we will present the practical assessment of *ARMOR* and compare its performance to existing methods in the field.

Chapter 4

Experimental Evaluation of *ARMOR*

In this section, we present the empirical evaluation of *ARMOR*. We assess the effectiveness of the proposed FL defense mechanism to counter edge-case backdoor attacks, and compare it to various existing FL defense mechanisms, applied to widely used datasets and models. More precsiely, we aim to answer the following questions, by comparing *ARMOR* to various existing FL defense mechanisms:

1. How resilient to edge-case backdoors is *ARMOR*?
2. What is the impact of *ARMOR*'s adaptive mitigation approach on resilience to edge-case backdoors?
3. How does *ARMOR* compare to state-of-the-art FL defense mechanisms?
4. What is the utility (*i.e.*, model quality) under *ARMOR*, and under other FL defense mechanisms?
5. How does FL defense behave in front of various numbers of attackers?
6. How much is FL defense resilient with regard to attack frequency?
7. What if attacks occur early, before the model stabilizes?
8. What is the cost of robust FL systems?

In the following, we first present *ARMOR* implementation details, and our experimental environment. Then, we present extensive evaluation results to precisely answer these questions.

4.1 Implementation and Experimental Setup

We implemented the proposed *ARMOR* FL defense mechanism using the PyTorch framework [86]. The software prototype of *ARMOR* consists of 2.5 KLOC, and is publicly available at[1]:
https://github.com/robust-fl/armor

We compare *ARMOR* against five state-of-the-art defense mechanisms by using publicly available software prototypes of Multi-Krum [37], NDC [54], Trimmed Mean [35],

[1]The anonymous link will be replaced by the public git link.

FLTrsut [18], and DLMP [32]. We use the implementation of the edge-case backdoor attack from [104], as described in §2.2.2.

Our experiments are conducted using widely used datasets for image classification tasks, such as MNIST [63], FashionMNIST [109], and Cifar-10 [59]. Each dataset contains 50,000 training images and 10,000 test images. For MNIST dataset, we use a five-layer neural network with three convolution layers and two fully connected layers. For FashionMNIST, we trained a four-layer convolutional neural network with two convolution layers, a fully connected layer and a maxpooling layer. Finally, for Cifar-10 dataset, we implement the Resnet-18 model [44]. The non-IID data distribution used in our experiments is generated using the Dirichlet distribution [50]. Furthermore, we conducted experiments with all considered state-of-the-art defense mechanisms on the three datasets. However, due to space limitation, we present in the following results of all or some of these datasets. All our experiments are executed on a server with 2 Intel Xeon Gold 6126 processors, with 12 cores each, 1 Nvidia Tesla P100-PCIE-16GB GPU, with 192 GiB memory, and deployed in Ubuntu 18.04 operating system.

4.1.1 Evaluation Metrics

In our experiments, we consider the following evaluation metrics:

Backdoor task accuracy. This metric is a means to quantify the resilience of a FL system to backdoors. Given the testing set of the attacker(s) that consists of data samples with backdoors, the backdoor task accuracy is the ratio between the number of data samples of the attacker's testing set that fall into the target class C_{target} divided by the total number of samples of this testing set. Thus, the lower is the backdoor task accuracy, the higher is the FL system resilience.

Main task accuracy. Given a backdoor-free testing set, the FL global model is tested to measure the correctness of its predictions. We refer to the accuracy of these predictions as the main task accuracy. This metric is a means to quantify the utility of the FL model.

Runtime cost. In order to quantify the computational cost of a FL defense mechanism, we measure the server-side cost as the average execution time of a FL round when using a given defense mechanism.

4.1.2 Federated Learning System Settings

In the following, we consider a FL system with 10 workers, among which there is one attacker (unless otherwise specified). For simplicity, all workers are selected at each round. The training batch size is set to 64, and the workers' learning rate to train their local models are respectively set to 0.01 for MNIST and FashionMNIST, and to 0.001 for Cifar-10. The default FL server uses the FedAvg model aggregation method [71].

The different FL defense mechanisms were configured as follows (unless stated otherwise). We empirically found these configurations in such a way that they provided the best behavior for each defense mechanism. Multi-Krum's f parameter is

set by default to 1, that is one attacker out of a total of 10 workers, and it is increased in experiments involving more than one attacker. Trimmed Mean' β parameter is set to 20%, and NDC's norm bound parameter M is set to 5. *ARMOR*'s sliding window size s is set to 2, mitigation parameter θ is set to $exp(-5 * L_t)$, and threshold γ_i is set to 0.12 for MNIST and FashionMNIST, and 0.38 for Cifar-10.

4.2 Experimental Results

4.2.1 Resilience of *ARMOR* to Edge-Case Backdoors

4.1 evaluates the effectiveness of our proposal, in terms of main task accuracy and backdoor task accuracy, comparing the case where there is no defense with the case where *ARMOR* defense is used. Here, an edge-case backdoor occurs at every round, starting from round 50. We see that the proposed FL defense is highly resilient to edge-case backdoors with a good model accuracy, regardless of the dataset, *i.e.*, MNIST, FashionMNIST and Cifar-10.

4.2.2 *ARMOR*'s Adaptive Attack Mitigation

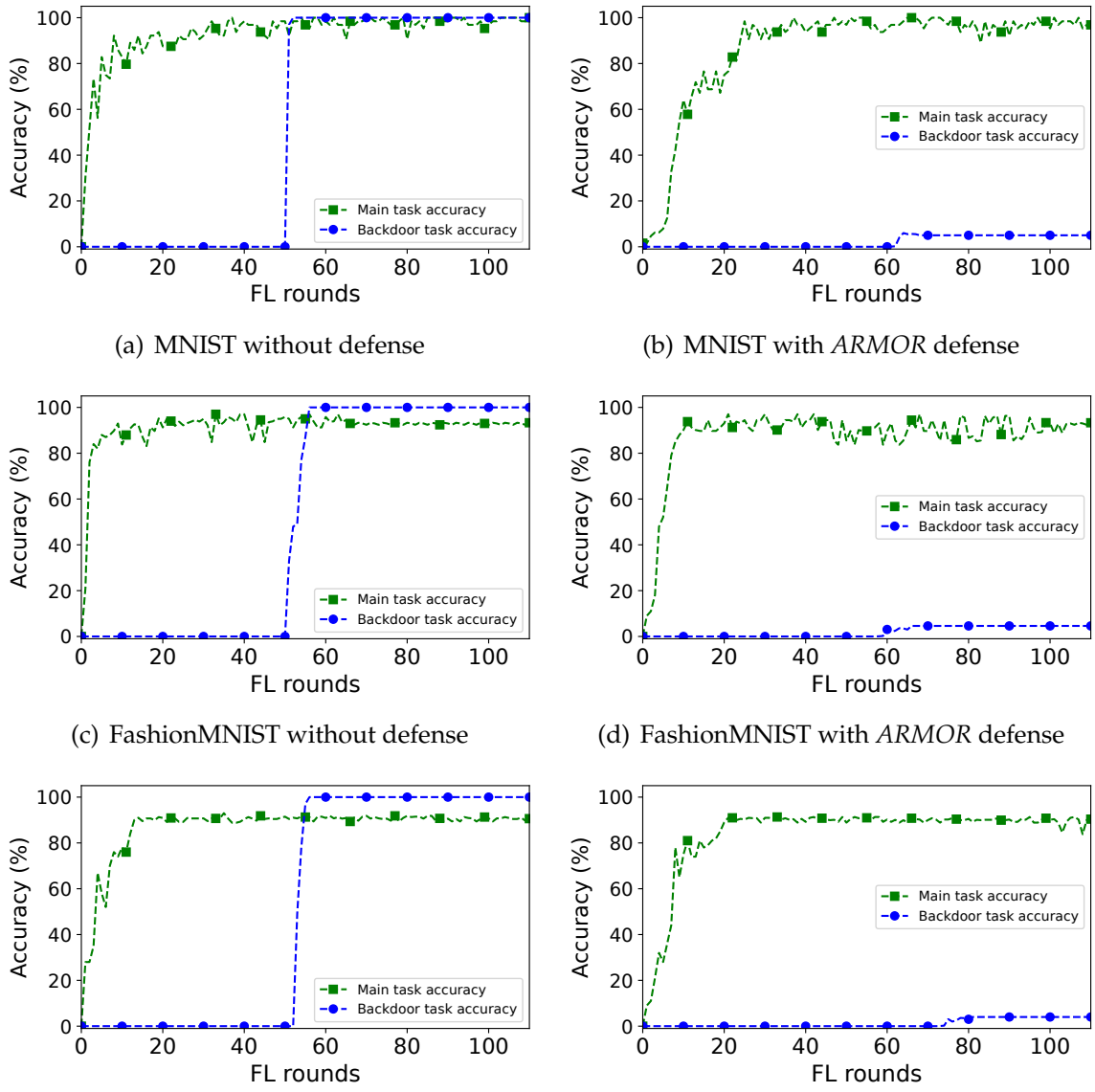
In the following, we evaluate *ARMOR*'s adaptive attack mitigation, *i.e.*, the ability of *ARMOR* to dynamically adapt the configuration of its θ mitigation parameter in order to improve the FL system resilience. θ mitigation parameter is a multiplicative factor used by *ARMOR* to attenuate a potential attack in model updates (*c.f.*, §3.4.3). We present the results of experiments conducted with *ARMOR* and FashionMNIST dataset, where one attacker injects a backdoor at every FL round starting from round 30, as illustrated in 4.2. We compare *ARMOR*'s ability to be resilient to edge-case backdoors when θ mitigation parameter is set statically vs. when it is set adaptively. In the latter case, *ARMOR* applies a loss-based heuristic introduced in §3.4.3, and we observe that this adaptive approach allows, indeed, much better resilience to edge-case backdoors than a static approach.

4.2.3 How Does *ARMOR* Compare to Other Federated Learning Defenses

4.3 evaluates the success of the edge-case backdoor task in a FL system, with MNIST, FashionMNIST and Cifar-10 datasets, in respectively Figures 4.3(a), 4.3(b) and 4.8(a). Here, an attack occurs every FL round, starting from round 50. In the following, we compare the resilience of different FL defense mechanisms.

Mechanisms with Prior Knowledge of a Validation Set

Both FLTrust and DLMP assume the existence of a validation set, in order to evaluate fluctuations in the accuracy of the model. They assume that if the model is poisoned, the classification on the validation dataset is likely inaccurate. But since the edge-case backdoor is based on data points that are unlikely to be part of the honest



(a) MNIST without defense

(b) MNIST with *ARMOR* defense

(c) FashionMNIST without defense

(d) FashionMNIST with *ARMOR* defense

(e) Cifar-10 without defense

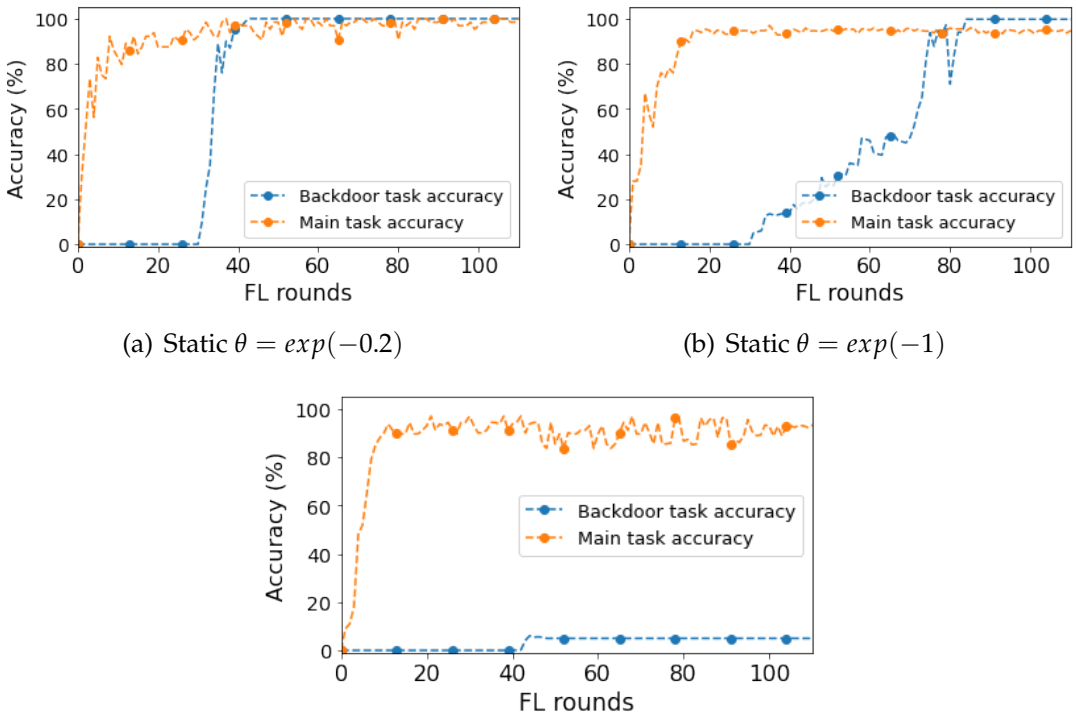
(f) Cifar-10 with *ARMOR* defense

FIGURE 4.1: Impact of edge-case backdoors on main task accuracy and backdoor task accuracy, with different datasets, without a FL defense (left side) and with *ARMOR* FL defense (right side). Attacks start at round 50, and occur every round

workers' data and the poisoned model behaves correctly on non-poisoned data as mentioned in Eq (3.1), there is no impact on the accuracy of the main task. Thus, FLTrust and DLMP are unable to counteract them.

Mechanisms without Prior Knowledge

We also present the results of existing FL defense mechanisms that do not assume the existence of a validation set, as Multi-Krum, NDC and Trimmed Mean. Multi-krum provides good attack mitigation, although, not persistent over time. Indeed, when the first attacks occur, the attacker's model update slightly diverges from the



(a) Static $\theta = exp(-0.2)$

(b) Static $\theta = exp(-1)$

(c) Adaptive θ

FIGURE 4.2: Impact of edge-case backdoors on main task accuracy and backdoor task accuracy, with *ARMOR*. Attacks start at round 30, and occur every round. Different values of *ARMOR*'s θ mitigation parameter are considered: (a) static value $\theta = exp(-0.2)$; (b) static value $\theta = exp(-1)$; (c) adaptive value of θ

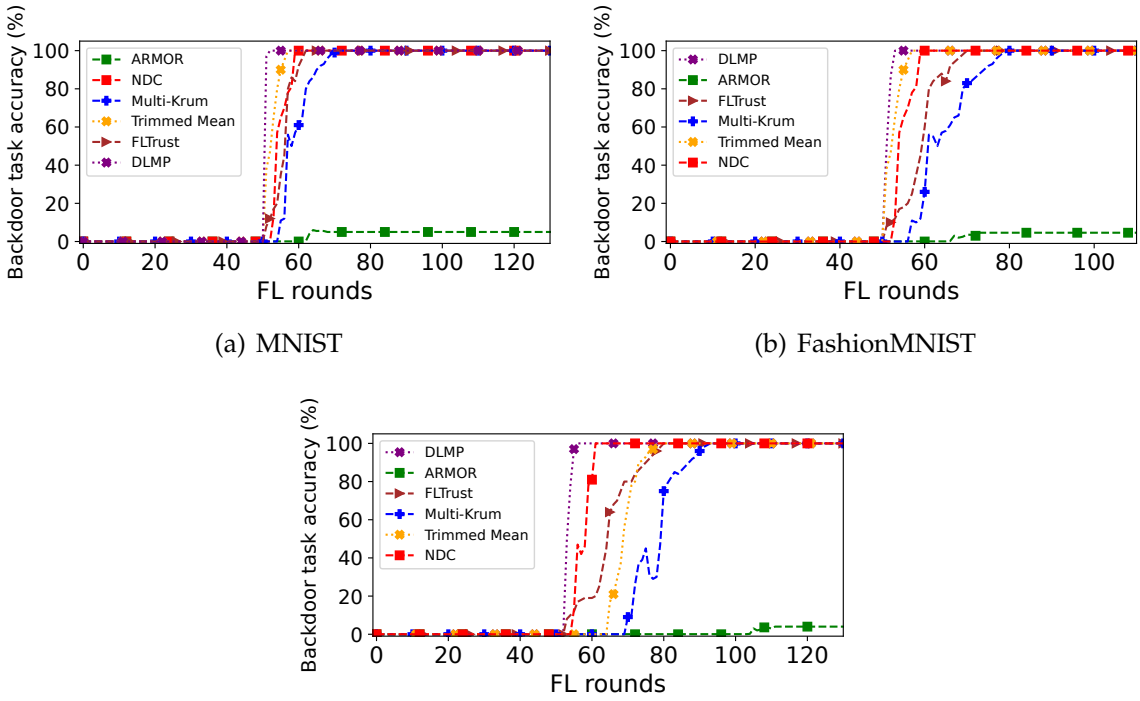
other clients' updates, thus, allowing Multi-Krum to mitigate the backdoor. However, with the following attacks and their application of PGD, the attacker's update converges closer to other clients' updates and, thus, the backdoor is not eliminated by Multi-Krum.

In contrast, *ARMOR* is able to counter 95% of edge-case backdoors.

This is mainly due to the quality of the synthetic validation dataset generated by *ARgan* because unlike the validation dataset used in FLTrust and DLMP [18, 32] , the dataset generated with *ARgan* includes edge-case data and therefore non-poisoned models have poor accuracy over it and this allows to monitor variations in the loss to mitigate backdoors. Indeed, *ARgan* uses as input the FL application model, which has been trained with edge case backdoor samples as well as benign samples.

Resilience With Different Underlying Datasets

We see that the edge-case backdoor attack is more or less quickly effective, depending on the actual complexity of the underlying dataset and model. For instance, in case of Cifar-10 dataset and its complex model, it takes longer for the attack to succeed, compared to the other datasets. Nevertheless, *ARMOR* remains highly resilient.



(a) MNIST

(b) FashionMNIST

(c) Cifar-10

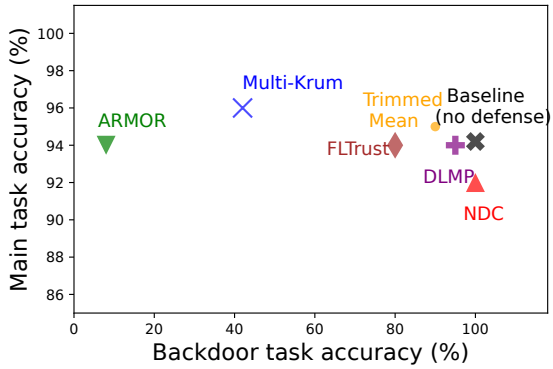
FIGURE 4.3: Effectiveness of edge-case backdoor attacks, with *ARMOR* and existing FL defenses, and with different datasets. Attacks start at round 50, and occur every round.

Resilience to Attacks vs. Model Utility

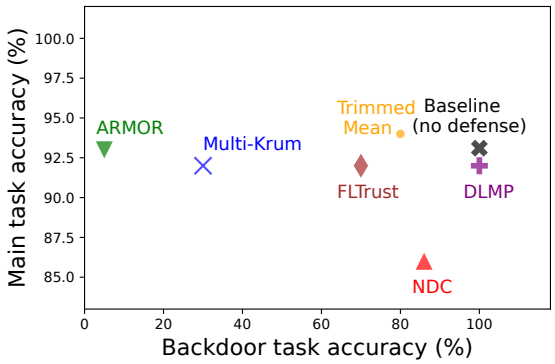
The resilience of a FL system to backdoors may come at the expense of a lower utility, *i.e.*, a lower model quality. In 4.4, we present the trade-off between utility (*i.e.*, the model's main task accuracy) and resilience to edge-case backdoor attacks. We evaluate the FL systems with an attack occurring every round on MNIST, FashionMNIST and Cifar-10 datasets. DLMP and FLTrust, which fall into the category of FL defense mechanisms that assume the existence of a validation set, have a main task accuracy which is close to the one of the FL baseline system where no defense mechanism is used. Indeed, such systems do not modify the aggregation applied by the FL server and, thus, provide a good model utility compared to the baseline. However, these defense mechanisms are not resilient to edge-case backdoors. In contrast, defense mechanisms that are based on specific aggregation approaches, such as Multi-Krum, Trimmed Mean and NDC, induce a much higher difference in model utility compared to the FL baseline system, ranging from 7.1% to +2.8%.

Indeed, aggregation-based approaches such as NDC, Multi-Krum and Trimmed Mean, may impact model accuracy, although they do not sufficiently mitigate attacks.

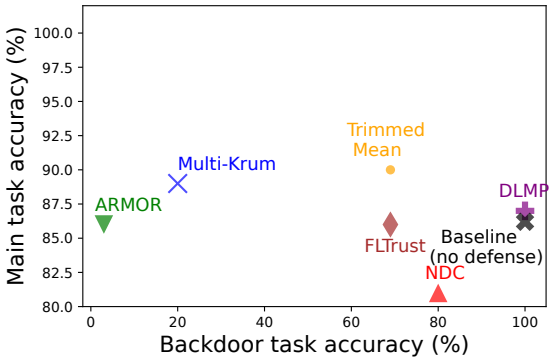
In comparison, with *ARMOR* the backdoor task accuracy does not exceed 5% without hurting the main task accuracy, thus, providing the best trade-off between resilience and utility.



(a) MNIST



(b) FashionMNIST



(c) Cifar-10

FIGURE 4.4: Trade-off between resilience to edge-case backdoors and model utility, through edge-case backdoor task accuracy vs. main task accuracy, with *ARMOR* and existing defense mechanisms.

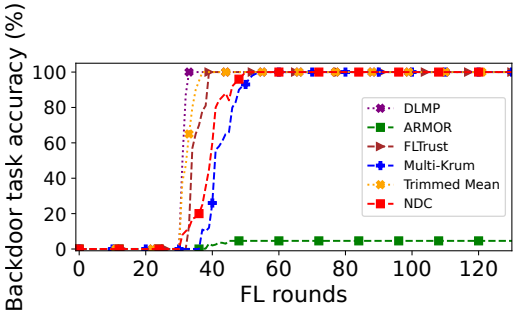
4.2.4 Impact of Number of Malicious Clients on Federated Learning Defense

In 4.5, we evaluate the impact of the number of malicious clients on the FL defense, comparing *ARMOR* to state-of-the-art mechanisms. Here, we show the results of experiments conducted with FashionMNIST dataset, and where there are between 2, 5 or 6 attackers out of the 10 workers, each attacker injecting one backdoor at every FL round, starting from round 30. We see that for DLMP, FLTrust, Multi-Krum[2], Trimmed Mean and NDC and , the more there are attackers, the faster is the backdoor attack success.

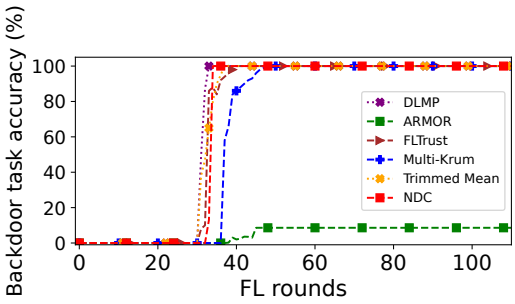
Regarding the methods that are based on a validation dataset, the attack is not detected as long as the edge case backdoors are not included in the validation dataset. moreover, when the number of attackers increases, the attack converges more efficiently, because the aggregation with honest clients is less successful in attenuating the effectiveness of the attack.

In contrast, *ARMOR* is much more resilient, with a backdoor task accuracy that slightly increases with the number of attackers.

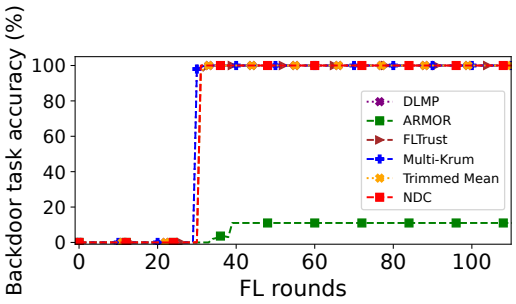
[2]Multi-Krum's f parameter is set to 2 (respectively, its maximum possible value 4 out of 10 workers) in case of 20% of attackers (respectively, 50% and 60%).



(a) 20% of clients are attackers



(b) 50% of clients are attackers



(c) 60% of clients are attackers

FIGURE 4.5: Effectiveness of edge-case backdoor attacks, with various numbers of attackers, under *ARMOR* and other existing FL defenses. Attacks start at round 50, and occur every round. Considered ratios of clients that represent attackers: (a) 20%; (b) 50%; (c) 60%

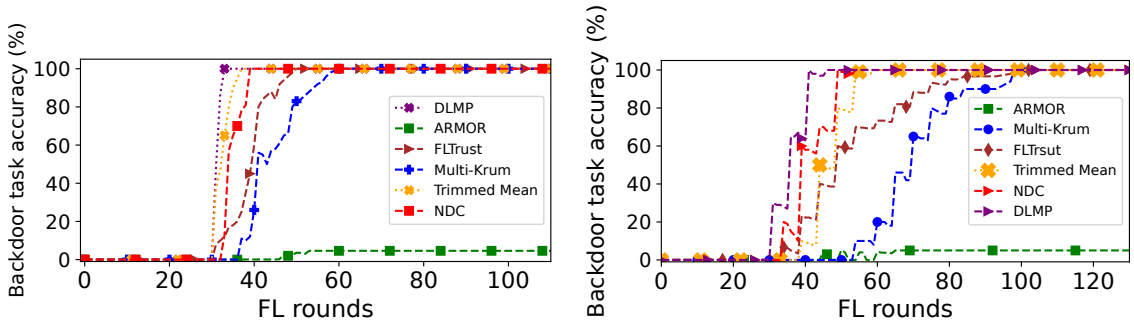
4.2.5 Impact of Attack Frequency on Federated Learning Defense

Here, we aim to answer the following question: How much is *ARMOR* resilient under different attack frequencies? In 4.6, we present evaluation results of different FL defense mechanisms, for FashionMNIST and one attacker injecting an edge-case backdoor starting from FL round 30, either at every round, or every 5 rounds for a less aggressive scenario.

We observe that, even in an optimistic case with a lower attack frequency, after a dozen of rounds, the backdoor is introduced. When the attack occurs with a low frequency, the incorporation of the backdoor into the model requires a lot of time, since the aggregation with honest client models reduces its effectiveness. However, the state-of-the-art methods are not effective against it. Only *ARMOR* is able to protect the FL system against the backdoor attack over time.

4.2.6 Attacks Occurring in Early Rounds

We now analyze the effectiveness of *ARMOR* in a setting where the model is not yet converged, and where attacks start occurring at early rounds. We consider three settings, one where attacks start at the third round, one where attacks start at the



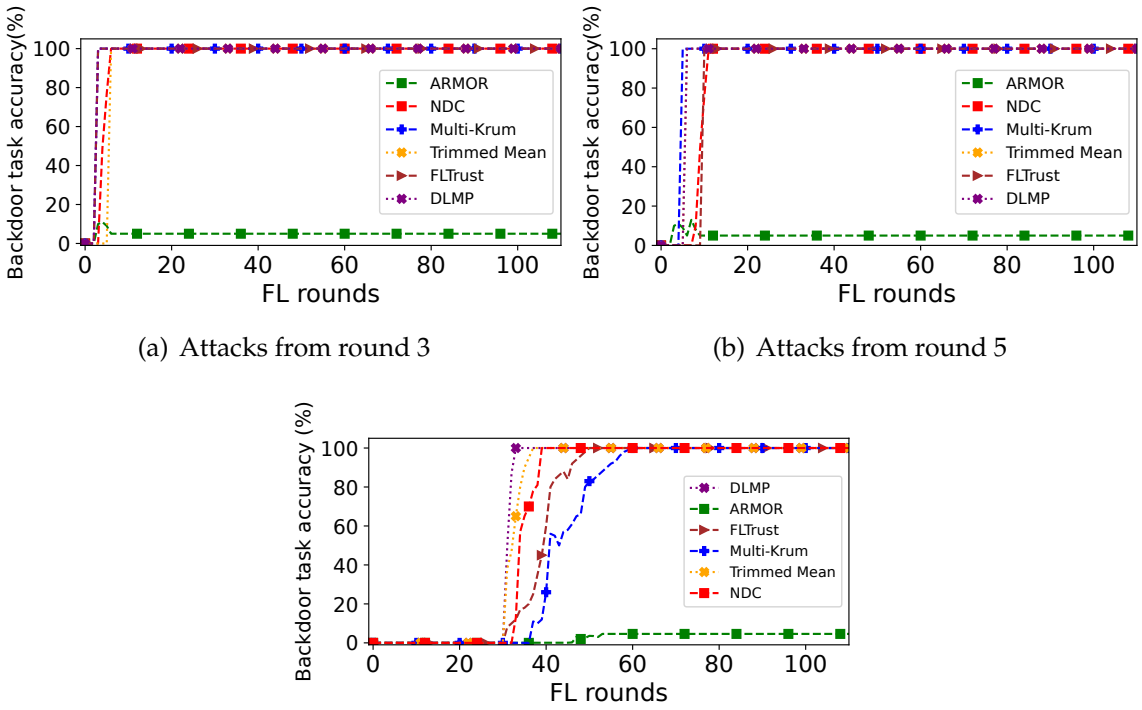
(a) An attack every round (b) An attack every 5 rounds

FIGURE 4.6: Effectiveness of edge-case backdoor attacks under various attack frequencies, with *ARMOR* and other existing FL defenses. Attacks start at round 30, and occur: (a) every round; (b) every 5 rounds

fifth round, and one where attacks start at round 30. Afterwards, attacks occur at every round. 4.7 presents the results of *ARMOR* and other defenses, with Fashion-MNIST. We see that *ARMOR* is highly resilient to attacks, even if they occur early. Indeed, with a minimum of s genuine models from initial rounds (here, $s = 2$), *ARMOR* is able to counter attacks. In contrast, state-of-the-art FL defenses are not robust to attacks that occur at early stages (*c.f.*, Figures 4.7(a) and 4.7(b)), since they rely on analyzing the geometric shape of model updates, or differences of accuracy with previous model, which is not sound if the model is not yet converged. Furthermore, we also measured the main task accuracy, although not presented here due to space limitation. *ARMOR* has a good main task accuracy, ranging between 86.5% and 87.8%.

4.2.7 Resilience to Different Types of Attacks

In the following, we consider the resilience of robust FL systems to different types of poisoning attacks. In case of data poisoning attacks, adversaries tamper with the training data before it is used to build local model updates that are sent to the FL server [36, 101]. Their objective is to misclassify specific inputs into a target class. The attacker can alter the label of only a sub-class (*e.g.*, small green cars) of the training data [7], or an entire class (*e.g.*, the “car" class) [36]. Since the training data is kept private on workers’ devices, this type of attack can not be directly detected by the FL server, since the latter can not check the labels of the training data of the worker. On the other hand, instead of tempering with training data, model poisoning attacks directly temper with the model updates. Thus, for a target poisoned version of the model, if the attacker has knowledge about the number of workers participating in the FL training round as well as the aggregation algorithm used by the FL server, the attacker can forge a model update that aims to bring the global model very close to the poisoned model. In Figure 4.8, we present results for data and model poisoning attacks both without defenses, with *ARMOR* and other state-of-the-art mechanisms. We notice that all the defense systems are more efficient with data poisoning attacks compared to model poisoning attacks, due to the nature of the attack. Actually, model poisoning attacks are designed more efficiently



(a) Attacks from round 3

(b) Attacks from round 5

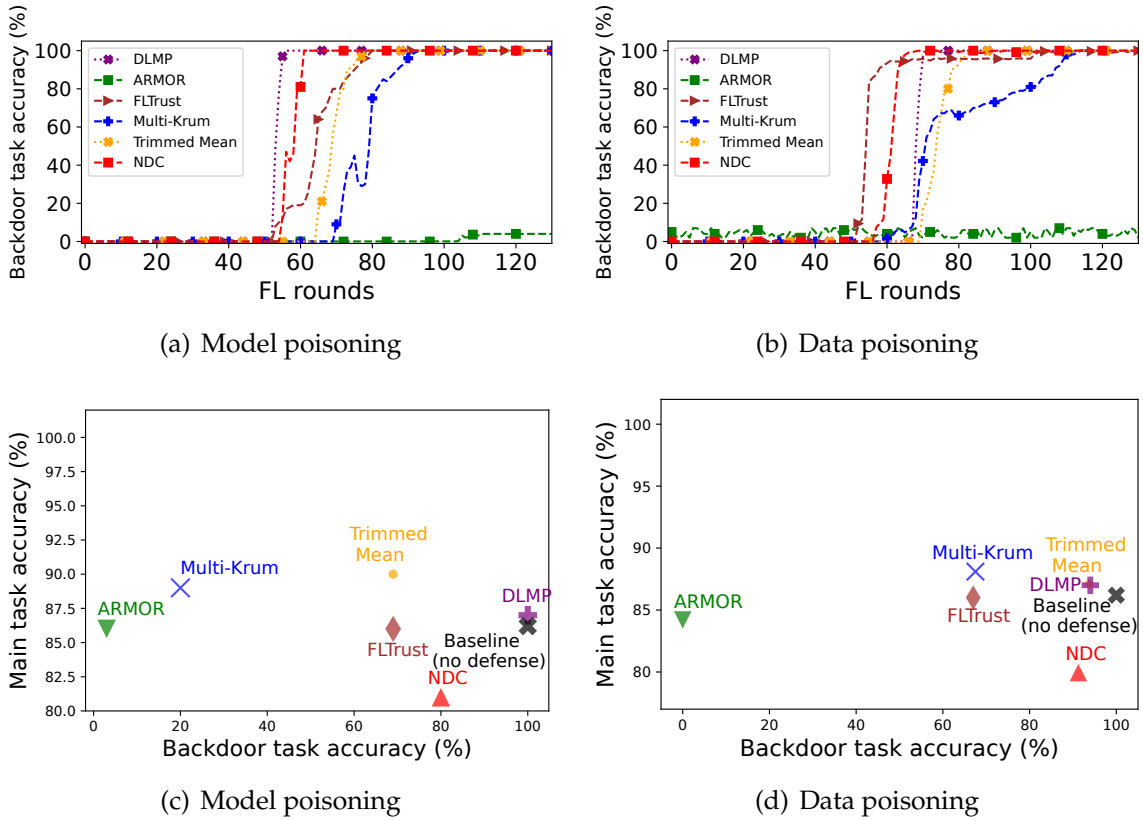
(c) Attacks from round 30

FIGURE 4.7: Edge-case backdoor task accuracy, with *ARMOR* and other existing FL defenses. Attacks occur every round starting from: (a) round 3; (b) round 5; (c) round 30

and are harder to detect, for instance, the attacker's update models are closer to the update models of other users since the PGD is used, by projecting attackers' models on a small ball, centered around the global model of the previous iteration. Unlike the model updates in data poisoning attacks which can be very distinct from other models.

4.2.8 Cost of Robust Federated Learning

We measured the execution times of one FL round at the server side, with each studied defense mechanism, and report them in Table 4.1. We observe that the FL server of DLMP, FLTrust, Multi-Krum and NDC induce a computational cost between 7 ms and 11 ms, whereas in Trimmed Mean this cost is a lower (3 ms) since its underlying calculations are much simpler compared to other defense mechanisms. In contrast, *ARMOR*'s FL server computational cost for an *ARgan* instance is 3 orders of magnitude higher than other FL defense mechanisms. This is due to the underlying cost of GANs in *ARgan*. This cost can be further reduced using extensive parallel GPU computations, since *ARgan* instances are independent from each other, so they can be trained in parallel on multiple GPUs. Moreover, recent works aim to improve the convergence speed of GANs [122], which could greatly help reducing *ARgan* cost. The FL server cost should be put in perspective when considering the overall cost of updating the global FL model, which implies local model training. The latter



(a) Model poisoning

(b) Data poisoning

(c) Model poisoning

(d) Data poisoning

FIGURE 4.8: Effectiveness of edge-case backdoor attacks, with *ARMOR* and existing FL defenses. Attacks start at round 50, and occur every round, with Cifar-10. Model poisoning on the left side and data poisoning on the right side.

part usually takes minutes to hours, depending on the underlying FL system and workloads.

TABLE 4.1: Cost of robust FL systems

Defense mechanism	FLTrust	DLMP	Multi-Krum	NDC	Trimmed Mean	ARMOR
Server-side cost	11 ms	9 ms	7 ms	8 ms	3 ms	2.1 s
Client-side cost	minutes to hours, depending on underlying cross-silo or cross-device FL workloads					

4.2.9 Evaluation of Privacy Risks of Synthetic Data

Despite the appeal of GANs to generate synthetic data, recent works show that GANs memorize information that can be used to infer data about the original dataset. citeDBLP:journals/popets/HayesMDC19 explores membership inference attacks against GANs, where white-box attacks and black-box attacks are tested against GANs.

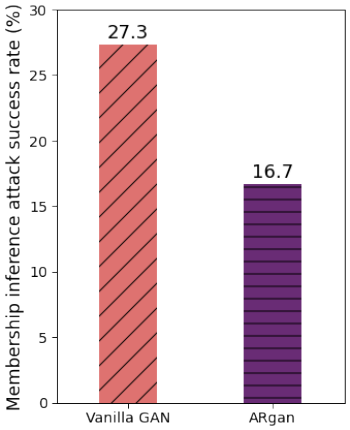


FIGURE 4.9: Membership inference attack against vanilla GAN vs. *ARgan*

Since *ARgan ARMOR*'s component is a variant of GAN, we evaluate in the following the effectiveness of membership inference attacks against *ARgan* vs. against vanilla GAN. 4.9 presents the success rate of such membership inference attacks, and shows that these attacks are much more effective against vanilla GAN (27.3%) than against *ARgan* (16.7%). This is due to the fact that vanilla GAN is trained with real input data, whereas *ARgan* takes as an input the model, that is an abstraction of the data, thus, reducing privacy leakage risks.

4.3 Summary

ARMOR can be applied to a wide range of learning tasks in FL. This includes, for instance, binary classification tasks such as diagnosing if a patient has a particular disease or not [28], or detecting if an email is a spam [60]. - More generally, *ARMOR* can be used with multi-class classification tasks involving tens or hundreds of classes, which covers applications ranging from agriculture [22], to medicine [8], or chemistry [68]. However, *ARMOR* is not practical in the case of more complex classifiers with a large number of classes, such as natural language processing applications and text classification involving hundreds of thousands or millions of classes. Indeed, *ARMOR*'s FL server induces a computational cost that is mainly due to the training costs of the GANs in *ARgan*. We tried to minimize this cost by replacing *ARgan* with a conditional GAN (cGAN) [78], that introduces an additional parameter representing the class for which data samples should be generated. As a result, cGANs allow to train a single instance of *ARgan* instead of training an instance per class. However, the empirical results show that cGANs do not allow to capture backdoors in the model because cGANs suffer from critical drawbacks such as lack of diversity in the generated outputs [89, 4].

Thus, many possibilities are hampered, including clustering in the latent space, better interpretability, improved interpolations, or output manipulation.

However, even if cGANs are not practical for our goal, the training cost of *ARgan* can be further reduced using extensive parallel GPU computations. Indeed, *ARgan* instances are independent from each other, so they can be trained in parallel on

multiple GPUs. Moreover, recent works aim at improving the convergence speed of GANs [122], which could greatly help reducing *ARgan* cost. That being said, the FL server cost should be put in perspective when considering the overall cost of updating the global model, which implies local model training at the client side. The latter part usually takes minutes to hours, depending on the underlying cross-silo or cross-device FL systems and workloads.

Part II

Privacy-Preservation in Federated Learning

Chapter 5

Background on Privacy Threats and Privacy Defense in Federated Learning

5.1 Motivation

Despite all the privacy safeguards of FL, it remains vulnerable to membership inference attacks [93]. The primary objective of this attack is to ascertain whether a particular data record has been utilized in training a target model. This vulnerability raises significant concerns, especially in sensitive domains like healthcare, where the inference of patient-specific data usage could compromise individual privacy and confidentiality. For instance, the application of a membership inference attack could enable adversaries to infer whether the medical records of a specific patient have been employed to train a classifier related to a particular disease.

However, the spectrum of privacy attacks extends beyond membership inference alone. Ateniese et al. have introduced the concept of property inference attacks [5], which aim to extract specific dataset properties, even if these properties are unrelated to the primary training task. In a scenario where the primary objective involves training a model for tasks such as race or gender recognition, a property inference attack may strive to deduce extraneous attributes of the training dataset, such as whether individuals in the training images wear glasses or not. This type of attack underscores the critical need to protect not only the primary training task but also any auxiliary information that might be inadvertently exposed through the training process.

Moreover, privacy attacks in federated learning encompasses model inversion or attribute inversion attacks, as discussed by Hitaj et al.[47] and Hidano et al.[46]. These attacks fall under the umbrella of reconstruction attacks, wherein adversaries, armed with output labels and partial knowledge of certain features, endeavor to reconstruct sensitive features or complete data samples. The potential ramifications of these attacks are considerable, as they could lead to the unintended disclosure of private information, jeopardizing the confidentiality of the individuals whose data contribute to the federated learning process.

In this chapter, we introduce the privacy threats associated with federated learning, focusing particularly on membership inference attacks. We outline the state-of-the-art privacy defense mechanisms in FL, including perturbation methods, cryptographic techniques, gradient compressibility, and Trusted Execution Environments (TEEs). The advantages and limitations of these approaches are discussed, along

with the presentation of open research questions. Detailed exploration of these points will follow in subsequent sections of this chapter.

5.1.1 Membership Inference Attacks

In the domain of data privacy and machine learning, membership inference attacks (MIA) stand out as a significant threat. These attacks aim to discern whether a particular data point was part of the training set for a given model. The implications of such attacks can be far-reaching, as they expose the potential for unauthorized access to sensitive information used in model training. This text further delves into the concept of membership inference attacks, their methods, implications, and significance.

The foundational work by Shokri et al. [93] introduced a black-box attack approach for MIA. This approach capitalizes on the probability distribution of output classes from the target model. In this scheme, attackers create shadow models, which replicate the behavior of the target model, and generate class probability distributions. These shadow models are then used to train a set of attack models, each specializing in identifying membership for a particular class. By providing confidence scores as input, these attack models predict whether a given data record was part of the training set. This methodology essentially exploits the subtle variations in class probabilities to reveal information about training data.

The research landscape evolved further with an extension to the original approach in [91], which utilized a single shadow model and relaxed the assumption that the shadow model mimics the target model's construction. Moreover, efforts like [103] broadened the scope of MIA by demonstrating its data-driven and transferable nature. This research found that membership inference attacks extend beyond specific architectures and exhibit transferability across different models.

Interestingly, the study in [75] delved into the mechanisms of membership privacy leakage through two facets: embedding layers and gradients. Notably, it highlighted how the gradients of an embedding layer can inadvertently expose the position of words in a training batch, enabling attackers to launch membership inference attacks.

To carry out a membership inference attack, the attacker employs a multilayer perceptron attack model, denoted as M_{attack}, which is designed to produce a binary output. The goal is to assess whether a given record, denoted as r_j, was likely part of the training data for the target model, M_{target}. The attack model returns a value of 1 to signify that the record is a member of the training set or 0 to indicate that it is a non-member.

Below is a succinct description of the process employed in executing a membership inference attack:

1. **Creating the Shadow Model** (M_{shadow}): The attacker first constructs a shadow model, M_{shadow}, utilizing the same architecture as the target model. This shadow model is trained using a dataset called D_{shadow}, which is a combination of two subsets: $D_{shadow_{train}}$ and $D_{shadow_{test}}$. The shadow model learns to mimic the behavior of the target model using this dataset.

2. **Building the Attack Model Training Dataset (D_{attack})**: The attacker then constructs the dataset D_{attack}, which will be used to train the attack model M_{attack}. This dataset is created by considering two subsets: $D_{attack_{Member}}$ and $D_{attack_{NonMember}}$.

   a. $D_{attack_{Member}}$ is formed by taking the features of data records from $D_{shadow_{train}}$ and assigning a label of 1 (indicating membership) to each record.

   b. $D_{attack_{NonMember}}$ is formed by taking the features of data records from $D_{shadow_{test}}$ and assigning a label of 0 (indicating non-membership) to each record.

3. **Training the Attack Model (M_{attack})**: With D_{attack} in place, the attacker trains the attack model M_{attack} using this dataset. The attack model learns the relationship between the features extracted from the shadow model's predictions (p_{shadow}) and the membership status labels (1 for members and 0 for non-members).

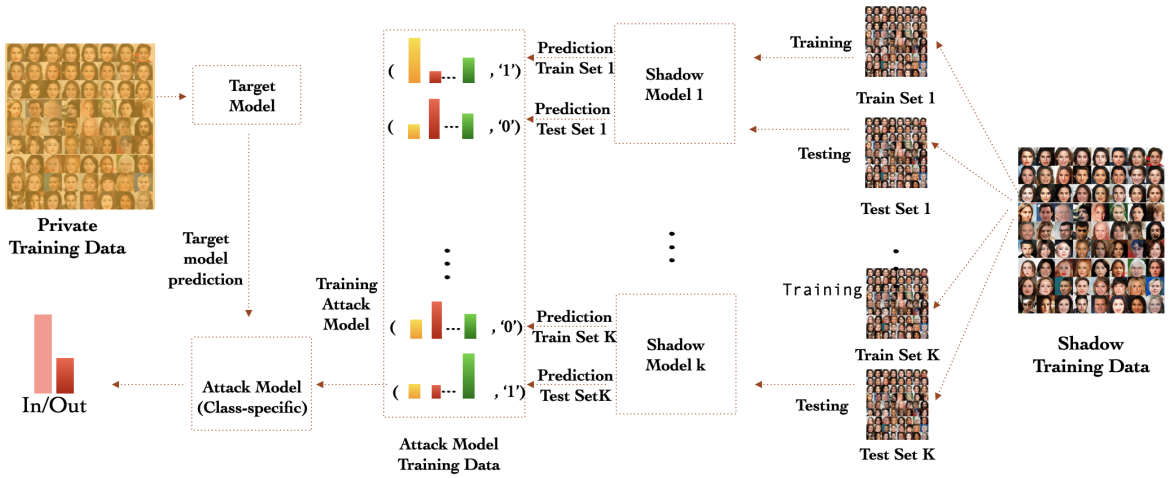
4. **Utilizing the Attack Model for Inference**: Once trained, the attack model M_{attack} can be utilized to make predictions about the membership status of a given data record r_j. The attack model takes the class probability distribution ($p_{target}(r_j)$) of the target model's prediction as input. If the attack model outputs a value of 1, it indicates that the record r_j is likely a member of the training set for M_{target}, and if it outputs 0, it implies that the record is probably not part of the training set.

5. **Maximizing the Attack Model's Performance**: The attacker's final objective is to maximize the Area Under the Curve (AUC) of the attack model's predictions. This involves optimizing the attack model's ability to accurately distinguish between members and non-members while testing it against the class probability distribution (p_{target}) of the target model's predictions.

The ramifications of membership inference attacks are considerable. They illuminate the vulnerability of machine learning models to information leakage about their training data. As a result, these attacks have gained prominence as indicators of potential privacy violations in federated learning models, which involve distributed training across various data sources while preserving data privacy. An individual's privacy can be compromised when an attacker determines that their data was used for training, potentially leading to unauthorized access to sensitive information.

Consider an example in smart health ubiquitous applications: membership inference attacks could deduce a patient's medical condition based on the inference that their clinical record was part of the training set. This underscores the sensitivity of such attacks, as they could unveil personal information that individuals would not willingly disclose.

The significance of the issue has been recognized by organizations like the National Institute of Standards and Technology (NIST). Their report [100] explicitly flags membership inference attacks as privacy violations. In summary, membership inference attacks pose a real and pressing concern in the intersection of machine learning and data privacy. Addressing these vulnerabilities is crucial to ensuring the integrity and security of sensitive data used in model training.



5.2 Privacy Defense in Federated Learning

In the landscape of mitigating membership inference attacks, the previous state-of-the-art endeavors can be categorized into four primary domains, each offering distinctive strategies to enhance the security of machine learning models and data repositories. These categories encompass perturbation methods, involving the strategic introduction of controlled noise or modifications to obscure sensitive information; cryptographic methods, which leverage encryption techniques to render data and models indecipherable to attackers; gradient compressibility methods, focusing on manipulating gradients during training to minimize information leakage; and Trusted Execution Environments (TEEs)-based methods, utilizing secure enclaves to safeguard computations from unauthorized access. In the upcoming sections, we will delve deeply into each of these techniques, elucidating their underlying mechanisms, implementation intricacies, merits, and potential limitations. Table 5.1 provides a comparison of different privacy-preserving methods in the context of Federated Learning. It categorizes these methods into three groups: Cryptography-based methods, Trusted Execution Environment (TEE)-based methods, and Perturbation-based methods. For each method within these categories, the table offers information on their privacy protection techniques, a brief description of how they work, their impact on model privacy and utility, and whether they introduce negligible overhead.

5.2.1 Perturbation-Based Methods

Perturbation methods, like differential privacy (DP) [1], introduce algorithm-specific random noise to safeguard against information leakage. In the context of FL, DP manifests in several forms: local differential privacy (LDP), central differential privacy (CDP), and weak differential privacy (WDP). LDP involves clients adding noise to their local models before transmission, while CDP entails the server aggregating models without noise and subsequently adding noise to the aggregated model [82]. WDP, on the other hand, employs norm bounding and Gaussian noise addition to protect privacy [2].

In a recent study [82], a comparative evaluation of LDP, CDP, and WDP was conducted in the context of edge FL, assessing their viability and effectiveness against white-box membership inference attacks. The experiments indicated that both LDP and CDP strike a balance between model accuracy and membership inference attack resilience. These methods manage to reduce the efficacy of such attacks while still maintaining reasonable model accuracy.

Another study [20] explored the integration of DP at both local and central levels to enhance participant privacy. This hybrid approach is designed to not only protect privacy but also improve model accuracy. To further enhance accuracy, sparse gradients and momentum gradient descent were implemented on both the server and client sides, which also contributed to decreased communication costs. The framework demonstrated remarkable outcomes, achieving up to 90% communication cost reduction while retaining superior accuracy and robust privacy protection.

In addition to the above examples, researchers have also investigated other approaches for mitigating membership inference attacks in FL. These might include advancements in cryptographic protocols that provide stronger privacy guarantees, techniques for enhancing the efficiency of noise addition to models, and innovations in secure aggregation methods that reduce the potential for information leakage.

5.2.2 Cryptography-Based Methods

Ccryptographic methods offer innovative techniques such as Homomorphic Encryption (HE) [31] and secure multi-party computing (SMC)[65], which play pivotal roles in enhancing privacy and security in federated learning (FL). Homomorphic encryption introduces a groundbreaking concept where data is encrypted and then subjected to various mathematical operations, producing an output that, when decrypted, precisely mirrors the results that would have been obtained if the operations were performed on the original, unencrypted data. This property of homomorphic encryption preserves the integrity of the original data throughout computations, thereby maintaining performance consistency during model convergence. For instance, consider a scenario where a group of hospitals collaborates to train a machine learning model on patient data. By employing homomorphic encryption, they can jointly perform computations on encrypted medical records, producing accurate model updates without compromising patient privacy.

Nonetheless, the efficiency gains of homomorphic encryption are not without trade-offs. While it guarantees data integrity and confidentiality, the computational and memory overhead associated with homomorphic encryption can be substantial. This can lead to slower processing times and higher resource consumption, impacting the overall efficiency of the federated learning process. For example, financial institutions seeking to collaboratively train a fraud detection model might face challenges in achieving real-time model updates due to the computational complexities introduced by homomorphic encryption.

Secure multi-party computing (SMC), on the other hand, addresses the privacy concerns by allowing multiple participants to collectively compute functions on their individual private inputs. This approach ensures a high level of privacy and data accuracy as participants never have to expose their raw data to each other. A classic example could involve multiple organizations analyzing customer behavior

data to develop a collaborative recommendation system. By employing SMC, these organizations can jointly compute personalized recommendations without sharing raw customer preferences or behaviors.

However, the benefits of SMC also come with their own set of drawbacks. The extensive computational and communication requirements of SMC protocols can result in longer training times and increased communication overhead. For instance, in scenarios where telecommunication companies aim to collaboratively build a churn prediction model, the resource-intensive nature of SMC could impede the timely delivery of accurate model updates.

In the context of federated learning, a notable extension to secure multi-party computing is the application of secure aggregation techniques. These techniques, initially explored in [71] and later detailed in [17], address the challenges posed by coordinating all clients in the federated learning process as required by traditional SMC. By leveraging secure aggregation, clients can contribute model updates without the need for direct coordination, streamlining the process and mitigating the complexity associated with traditional SMC. For instance, a consortium of research institutions collaborating on a global weather prediction model can benefit from secure aggregation, enabling them to efficiently aggregate model updates from various sources without requiring every institution to participate in every round of communication.

In the realm of cutting-edge developments, the *HybridAlpha* approach [111] has emerged as a pioneering solution for privacy-preserving federated learning. This approach combines secure multi-party computing with functional encryption, resulting in a protocol that is both efficient and resilient to participant dropouts. By employing *HybridAlpha*, organizations can collaboratively train models while minimizing training time and data transfer volume. Consider a consortium of automotive manufacturers working on a federated model for autonomous vehicle safety. The *HybridAlpha* approach can ensure that even if some manufacturers leave the collaborative effort, the training process remains robust and effective.

In another innovative endeavor, researchers in [33] introduce the sqSGD (selective quantized stochastic gradient descent) algorithm. This algorithm addresses communication efficiency and high-dimensional compatibility issues in privacy-preserving federated learning. SqSGD allows for the training of large models with random initialization while maintaining min-max optimality under both communication and privacy constraints. For instance, a group of e-commerce platforms collaborating on a recommendation system can utilize sqSGD to efficiently train a model that accounts for the diverse range of products and user preferences.

Furthermore, an additional approach to bolstering privacy in federated learning involves gradient compression techniques [53]. These techniques focus on minimizing the information leakage that can occur when gradients are transmitted between clients and the central server. By compressing gradients, the communication overhead is reduced while maintaining the confidentiality of sensitive information. For example, a consortium of educational institutions sharing student performance data for collaborative research could employ gradient compression techniques to ensure that individual student records are not exposed during the federated learning process.

In conclusion, the landscape of cryptographic methods in federated learning is

marked by a diverse array of techniques such as homomorphic encryption, secure multi-party computing, secure aggregation, and innovative algorithms like *HybridAlpha* and sqSGD. Each technique addresses different aspects of privacy, security, and efficiency, offering solutions that cater to the varying needs of collaborative machine learning scenarios across different domains.

5.2.3 TEEs-Based Methods

Trusted Execution Environments (TEEs) have gained significant attention as a promising approach to enhancing privacy protection in various domains. For instance, in the context of healthcare, TEEs can be employed to securely process sensitive patient data, such as electronic health records, enabling researchers and healthcare providers to perform analytics while maintaining data privacy [62, 77]. Moreover, TEEs find applications in secure multiparty computation scenarios, where multiple parties collaborate on computations involving confidential data without exposing the raw data to each other.

Recent research has also focused on addressing the challenge of computational overhead when deploying TEE-based solutions. Innovative techniques, like hybrid approaches combining hardware-based TEEs with software optimizations, have shown promise in reducing the overhead associated with secure computations. These approaches aim to strike a better balance between privacy protection and efficient performance.

Additionally, the integration of TEEs with machine learning models is a noteworthy area of exploration. Privacy-preserving machine learning techniques, such as federated learning and secure model aggregation, can benefit from the security guarantees provided by TEEs, ensuring that sensitive model updates and data contributions remain confidential.

As TEEs continue to evolve, concerns surrounding their implementation cost and infrastructure have not gone unnoticed. Initial investment and setup for TEE infrastructure can indeed be substantial, potentially limiting their widespread adoption. Nonetheless, research efforts are being directed towards making TEEs more accessible and cost-effective.

5.2.4 Gradient Compression Methods

Gradient compressibility and sparsity techniques have emerged as critical strategies to mitigate the challenges posed by communication and computational overhead in various machine learning applications, including Federated Learning (FL) [41]. By reducing the volume of data exchanged between devices or nodes during the training process, these techniques offer efficiency gains that are particularly valuable in distributed settings.

One significant benefit of incorporating gradient compressibility and sparsity into FL relates to bolstering privacy protection mechanisms. By limiting the information shared among participants, these techniques contribute to thwarting privacy inference attacks, wherein malicious actors attempt to extract sensitive information from shared model updates [66]. This is of paramount importance in scenarios where data originates from different sources with varying privacy constraints.

Notably, recent research has demonstrated the efficacy of gradient compression in preventing unintended model leakage and privacy breaches [123]. The ability of gradient compression techniques to effectively obfuscate sensitive information during communication has made them an attractive option for enhancing the privacy posture of FL systems. Nevertheless, it's essential to acknowledge that while gradient compression brings about enhanced privacy guarantees, it does come with trade-offs. Specifically, there's a marginal performance degradation observed in the global model's accuracy due to the compression process.

A comprehensive exploration of inference attacks within vertical Federated Learning has been undertaken by Fu et al. in [34]. Their study underscores the effectiveness of countering label inference attacks through the strategic deployment of gradient compression. By minimizing the availability of detailed gradient information, gradient compression serves as a potent defense mechanism against certain forms of privacy breaches. However, it's pertinent to recognize that this protective measure isn't without its consequences. The authors note that while gradient compression helps fortify the FL system against privacy threats, it can lead to a decline in the performance of the federated model's original primary task. This trade-off necessitates a careful balancing act between privacy preservation and maintaining model utility.

In conclusion, the integration of gradient compressibility and sparsity techniques into Federated Learning frameworks offers a promising approach to addressing communication and computational bottlenecks, while simultaneously enhancing privacy defenses. Despite the associated performance trade-offs, empirical evidence supports their utility in preventing privacy breaches and inference attacks. The ongoing challenge lies in refining these techniques to strike the optimal balance between privacy preservation and model effectiveness in federated scenarios.

5.3 Summary

In the realm of data privacy within federated learning (FL), the concept of differential privacy stands out as a promising approach. It offers intriguing privacy assurances by safeguarding against membership inference attacks, wherein malicious actors try to deduce whether a particular data point was part of the training dataset. Furthermore, it ensures that no participant can reverse-engineer the private data of another participant using the aggregated updates. However, while differential privacy holds great potential, its implementation often incurs substantial trade-offs. Notably, mechanisms hinged on differential privacy can exact a considerable toll on both the efficacy of the model and the computational resources required. This is particularly evident in methods like local differential privacy, where the utility and computational burden increase exponentially, as pointed out in the reference [82].

Similarly, alternative techniques such as Secure Multi-Party Computation and Homomorphic Encryption, though offering enhanced security, introduce significant computational overheads. These methods, while protective, tend to clash with the integration demands of diverse FL architectures, as highlighted in [53]. Moreover, while gradient compression strategies present an effective countermeasure against inference attacks, bolstering the privacy of the model, it's essential to acknowledge that this strategy might inadvertently lead to diminished performance in the model's primary task within the FL framework.

In the upcoming chapter, we will delve into the introduction of two distinct methods: *PASTEL* and "Dinar," both tailored to mitigate membership inference attacks in federated learning. DINAR draws its essence from the concept of layer obfuscation, proposing a technique where the layers of the model are intentionally hidden to hinder attackers' attempts to infer membership. On the other hand, *PASTEL* hinges on a multiobjective training function. It strives to strike a balance between model accuracy and privacy while training, thereby fortifying the model against membership inference attacks. Our exploration will encompass the underlying design principles that govern these methods, shedding light on the intricacies of their implementation and their respective advantages.

The evaluation of these methods will serve as a critical focal point of the upcoming chapter. Through empirical analysis and experimentation, we will gauge the effectiveness of *PASTEL* and DINAR in mitigating membership inference attacks within the federated learning paradigm.

TABLE 5.1: Comparison of FL privacy-preserving methods

Privacy -preserving category	Protection method	Description	Model privacy	Model utility	Negligible overhead
Cryptography-based methods	PEFL [118]	Apply homomorphic cryptography to secure local models parameters without compromising its utility but causes high computational overhead with IOT devices	✓	✓	✗
	HybridAlpha [111]	Use an SMC protocol based on functional encryption	✓	✓	✗
	[26]	Use multiparty computation and modify the aggregation protocol, allowing participants to train without accessing the global model.	✓	✓	✗
TEE-based methods	MixNN [62]	Mixes layers between participants before sending the mixed updates to the aggregation server	✓	✓	✗
	GradSec [77]	Protect sensitive layers with TEEs to reduce the leakage and keep good model utility	✓	✓	✗
Perturbation-based methods –	CDP [82]	Introduce noise to the global model prior to distributing it to clients, it allows to reduce information leakage, but leads to compromised model accuracy and significant computational overhead.	✓ / ✗	✗	✗
	LDP [82]	Add noise to local models before sharing with the server. it reduces model leakage but decrease model utilty and increase computational overhead	✓ / ✗	✗	✗
	FedGP [102]	This approach utilizes federated GANs to generate an artificial dataset based on participants' data, preserving privacy and reducing information leakage. However, it may face limitations such as training instability and the need for a sufficient number of training examples, while its privacy guarantee is not as strong as traditional differential privacy methods.	✓ / ✗	✗	✗
	WDP [99]	By employing norm bounding and introducing Gaussian noise with a low magnitude, the utility remains unaffected, while the effectiveness of attack mitigation is compromised	✗	✓	✗
Our method	DINAR	Obfuscate sensitive layers parameters in order to reduce layer-based generalization gap without compromising model utility	✓	✓	✓
	PASTEL	Use multiobjective optimization training to reduce generalization gap	✓	✓	✓

Chapter 6

DINAR: Fine-Grained Mitigation of Membership Inference Attacks in Federated Learning

6.1 DINAR Objectives

The current landscape of machine learning has seen a surge in Federated Learning (FL), a paradigm addressing privacy concerns by enabling collaborative model training across decentralized devices. In FL, participants locally train models with private data, transmitting only model parameters to a central server for aggregation into a global model. Applications range from e-health monitoring to fraud detection. Despite its privacy advantages, FL systems face privacy inference attacks, notably Membership Inference Attacks (MIAs), which exploit shared model parameters to glean sensitive training data information. Various defense mechanisms, employing cryptographic methods, secure computation, trusted environments, and perturbation-based techniques, have been proposed. However, finding the optimal balance between FL model privacy, utility, and computational cost remains a crucial objective for effective and privacy-preserving FL In this chapter, we propose DINAR, a fine-grained privacy-preserving FL method that tackles membership inference attacks. This approach is motivated by an interesting observation made in recent studies [79, 80], and confirmed in our empirical analysis in §6.2.2, that is there is a layer in neural networks that leaks more private information than other layers. Thus, DINAR is based on a simple yet effective approach that consists in protecting more specifically the FL model layer that is the most sensitive to membership privacy leakage.

DINAR runs at the FL client-side, and allows to protect both the global FL model and the client models. Whereas for its own model predictions the client uses its privacy sensitive layer as part of the model, that privacy sensitive layer is obfuscated before sending client model updates to the FL server. Thus, the aggregated model produced by the FL server includes an obfuscated version of the privacy sensitive layer. And when the client receives the protected global model from the server, it first restores its local privacy sensitive layer (*i.e.*, the non-obfuscated version of that layer) that was stored during the previous FL round, and integrates it into its copy of the global model, before actually using the resulting personalized model for client predictions. Furthermore, in order to improve the utility of the protected model, DINAR leverages the adaptive gradient descent technique to further maximize the

accuracy of the model [30]. Indeed, given the high-dimensional nature of optimization problems in neural networks, adaptive gradient descent allows to dynamically adjust the model learning rate for each dimension in an iterative manner.

6.2 Design Principles of DINAR

In the following, we first outline the overarching objectives of our privacy-preserving FL method, before motivating its fine-grained approach, and then detailing the design principles of its different components.

6.2.1 Overall Objectives of DINAR

We propose DINAR, fine-grained privacy-preserving federated learning, a novel FL scheme for privacy protection against membership inference attacks. The overall objective of DINAR is threefold:

- *Model privacy:* The proposed FL privacy-preserving method must provide effective protection of model information exchanged between the FL server and the clients (*i.e.*, global model parameters and clients' model updates), in order to protect the FL system against membership inference attacks. In other words, the objective of such a method is to minimize the accuracy of the model of the attacker that allows the latter to infer whether a given data sample was part of the FL training dataset.
- *Model utility:* The proposed FL privacy-preserving method must avoid any negative impact on the quality and accuracy of the FL models used by the clients, thus, providing a good overall performance of the FL system.
- *No additional overhead:* The proposed FL privacy-preserving method must ensure that no additional computational overhead is induced on the FL system, thus, allowing practical use of privacy protection in FL systems.

We will see in the following how DINAR reaches these objectives, respectively in §6.2.4 and §6.2.5 for model privacy, in §6.2.6 for model utility, and in §6.2.2 regarding overhead.

6.2.2 Motivation of DINAR's Fine-Grained Approach

Recent studies have analyzed the sensitivity and privacy risk of neural networks at a fine-grained level, to better characterize how much each layer of the model leaks privacy information [79, 80]. As claimed in these studies, a similar pattern appears in all models, namely, there is a layer that leaks more private information than other layers. To better illustrate this behavior, we conduct an empirical analysis with four different datasets (GTSRB, CelebA, Texas100, Purchase100) and their underlying models, deployed in a FL setting[1]. More precisely, we aim to characterize

[1] A description of the datasets and models can be found in §7.1.1, as well as the experimental setup in §7.1.2.

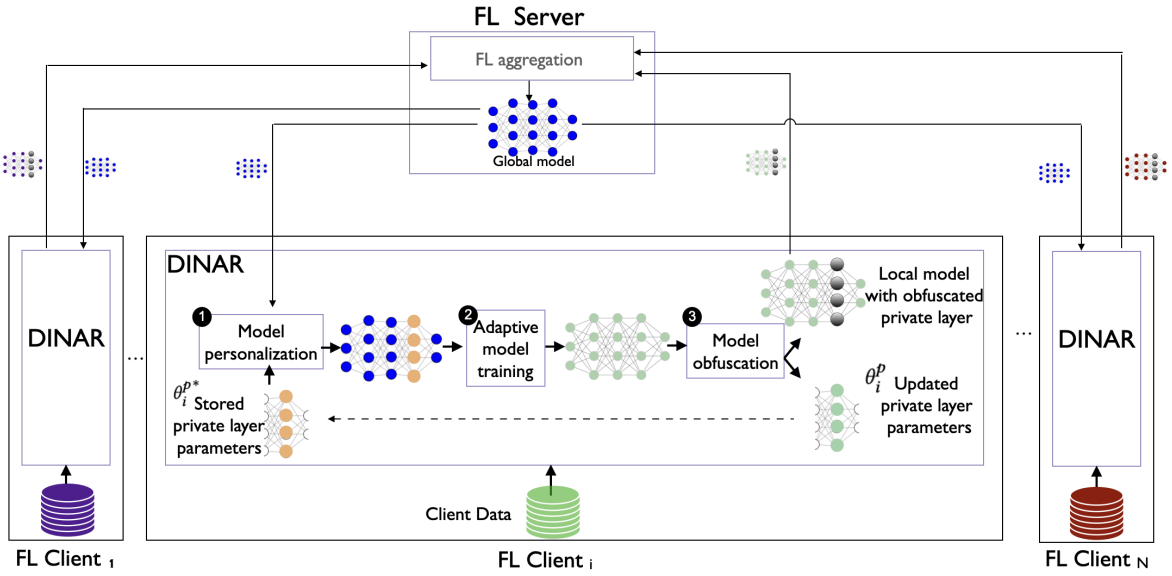


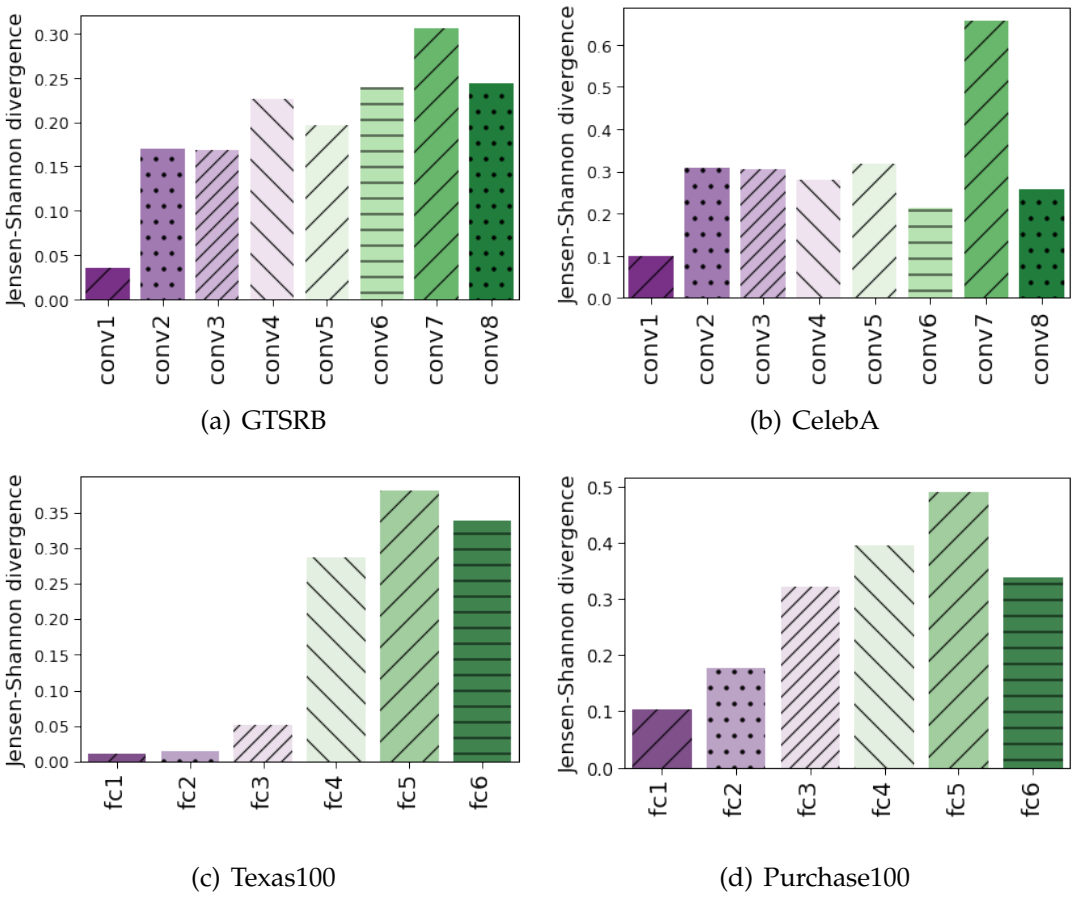
FIGURE 6.1: Overview of DINAR

how much each layer of a model helps an attacker conducting MIAs to better infer if a given data sample was member of the model training set or not. In other words and as described in §5.1.1, such an attacker is able to differentiate between member data samples and non-member data samples. Thus, using a trained FL model, we conduct, on the one hand, a set of predictions with member data samples, and on the other hand, another set of predictions with non-member data samples. We then compute the gradients of each layer resulting from the predictions of member samples, and the gradients of each layer resulting from the predictions of non-member samples. Finally, we compute the generalization gap of each layer, *i.e.*, the difference between the gradients of member data samples and the gradients of non-member data samples. Thus, the higher the generalization gap, the more successful MIA is, *i.e.*, the easier it is for the MIA to differentiate between members and non-members, as shown in recent studies [64, 108].

Our empirical results are presented in Figure 6.2, where the generalization gap is computed using the widely used Jensen-Shannon divergence [76]. We observe that different layers of a model may exhibit different generalization gaps. We also observe a similar behavior in all models, namely, the generalization gap of the penultimate layer is notably higher than the generalization gap of the other layers. Thus, that layer leaks more privacy sensitive information (*i.e.*, membership-related information), as shown in other studies [79, 80].

6.2.3 Overview of DINAR

Following the conclusions of §6.2.2, the intuition behind DINAR is to specifically handle the privacy sensitive layer of a FL model, in order to provide a non-intrusive yet effective solution to protect against MIAs. Indeed, existing privacy-preserving FL methods either apply perturbation on all model layers, or use cryptographic techniques and secure environments, which induce a high computational overhead (as shown in §5.2 and §7.2.5).



(a) GTSRB (b) CelebA

(c) Texas100 (d) Purchase100

FIGURE 6.2: Layer-level analysis of divergence between member data samples and non-member data samples, using Jensen-Shannon divergence, when FL models are not protected against membership inference attacks – FL models of GTSRB and CelebA have eight convolutional layers, and FL models of Texas100 and Purchase100 have six fully connected layers

The overall architecture of DINAR is presented in Figure 6.1. DINAR runs at the client-side, for each FL client that wants better protection against MIAs. Each DINAR instance on a client runs independently from the other clients' DINAR instances, and the interaction between the FL server and the clients follows the classical FL protocol, where at each FL round the clients send their local model updates to the server, and the server sends the aggregated global model to the clients.

The privacy-preserving method proposed by DINAR consists in tackling more specifically the penultimate layer of the model, *i.e.*, the *privacy sensitive layer*, which reveals more client's privacy information than the others (*c.f.*, §6.2.2). Whereas for its own model predictions the client uses its privacy sensitive layer as part of the model, that privacy sensitive layer is obfuscated before sending client model updates to the FL server. Thus, the global aggregated model produced by the FL server includes an obfuscated version of the privacy sensitive layer. Upon receiving the protected global model from the FL server, the client first restores its local privacy sensitive layer (*i.e.*, the non-obfuscated version of that layer) that was stored in the previous FL round, and integrates it into the global model, before actually using the resulting

personalized model for client predictions.

The pipeline of DINAR is described in Figure 6.1, through the successive stages of client model personalization (step ❶), adaptive model training for improving model utility (step ❷), and model obfuscation (step ❸). In the following, we first describe DINAR's model obfuscation in §6.2.4, before presenting DINAR's client model personalization in §6.2.5, and adaptive model training in §6.2.6.

Algorithm 3: DINAR algorithm on FL Client$_i$

Input: θ: global model parameters
Output: θ_i: client model parameters

Local variables:
θ_i^{p*}: parameters of private layer of client model
$(B^i, Y) = \{(B_1^i, Y_1), \ldots, (B_x^i, Y_x)\}$: training batches of Client$_i$
η: learning rate

```
// Model Personalization
for j in {1..J} do
    if j ≠ p then
        // Use j^th layer parameters from global model
        θ_i^j ← θ^j
    else
        // Restore parameters of client's private layer
        θ_i^j ← θ_i^{p*}
// Adaptive Model Training
// Set initial accumulated gradients matrix
G ← 0
foreach local training epoch do
    foreach (B_k^i, Y_k) ∈ (B^i, Y)) do
        // Perform local prediction
        Ŷ_k ← θ_i(B_k^i)
        // Compute new accumulated gradients
        G ← G + ∇L_θ(Y_k, Ŷ_k)
        // Update local model with adaptive gradient descent
        θ_i ← θ_i − η ∇_θ.loss / (√G + 1e^{-5})
// Model Obfuscation
// Save parameters of client's private layer
θ_i^{p*} ← θ_i^p
// Obfuscate parameters of client's private layer
θ_i^p ← random_values
return θ_i
```

6.2.4 Model Obfuscation

In the following, we consider a model W with J layers, and model parameters θ, where $\theta^1 \ldots \theta^J$ are the parameters of the respective layers $1 \ldots J$. We denote p the index of the privacy sensitive layer of model W. According to our empirical analysis and previous studies presented in §6.2.2, the privacy sensitive layer is the penultimate layer of W, that is $p = J - 1$. At each FL round, Client$_i$ that participates to that

round updates its model parameters θ_i through local training. Before sending the local model updates to the FL server, the client obfuscates the privacy sensitive layer of its model, namely θ_i^p that is the client model parameters of layer p. This obfuscation is simply performed by replacing the actual value of θ_i^p by random values. The resulting local model updates are sent to the FL server for aggregation. Note that the raw parameters of the privacy sensitive layer (*i.e.*, before obfuscation) are stored at the client side in θ_i^{p*}, and will be used in other stages of the DINAR pipeline.

6.2.5 Model Personalization

As presented in Figure 6.1, this is the first step of DINAR pipeline. When Client$_i$ participates to a FL round, it first receives the parameters θ of the global model W. In case of DINAR, θ^p, *i.e.*, the model parameters of the privacy sensitive layer p, contain obfuscated values. Here, the client integrates to its local model parameters θ_i all global model layer parameters but the parameters θ^p of layer p. Instead, the client restores for that layer θ_i^{p*}, its previously stored and non-obfuscated local model parameters of layer p. Thus, while the global FL model is protected against MIAs, Client$_i$ makes use of an effective personalized local model. This allows client model's privacy sensitive information to remain protected, while client data still contributes to the overall improvement of the global model through collaborative training.

6.2.6 Adaptive Model Training

While DINAR's model obfuscation and model personalization tackle model privacy against MIAs, this step of DINAR pipeline allows to improve model utility. Specifically, it aims to maximize the client model accuracy. This relies on the optimization of the loss function, denoted as $\mathcal{L}$, for each Client$_i$ and its local model W_i. The loss function $\mathcal{L}$ represents the cumulative errors of the client model W_i across its training and testing data batches. In order to minimize the loss function $\mathcal{L}$, client model parameters θ_i are updated at each local training epoch, given a learning rate hyperparameter η (with $\eta \in [0, 1]$). The latter serves as a coefficient that scales the computed gradient values at each learning epoch. The learning rate plays a pivotal role in machine learning, significantly influencing both model accuracy and convergence of the loss function [96]. Setting the learning rate too low may lead to overfitting, while using excessively high values can result in unstable model accuracy despite accelerating the training process.

To address these convergence challenges, we leverage the adaptive gradient descent technique, which effectively mitigates the issues associated with local minima and saddle points [30]. This approach offers robust safeguards against overfitting. Firstly, when training intricate models like Convolutional Neural Networks (CNNs) over multiple iterations, adaptive gradient descent ensures a deliberate convergence, exhibiting a slower learning rate compared to algorithms such as Adam and RMSProp, particularly during the initial iterations [56, 81]. Secondly, given the high-dimensional nature of optimization problems in neural networks, this technique dynamically adjusts the learning rate for each dimension in an iterative manner. Thus, it effectively addresses the challenges posed by saddle points and local

minima, thus facilitating a smoother convergence of $\mathcal{L}$ across all dimensions. As a result, DINAR encompasses distinctive elements that proactively prevent the loss function from being entrapped in local minima and saddle points. Consequently, it significantly mitigates overfitting risks and promotes the convergence of the loss function and the accuracy of the client model W_i.

In summary, Algorithm 3 presents the different steps of DINAR pipeline, namely model personalization (lines 1–5), adaptive model training (lines 6–12), and model obfuscation (lines 13–15). And Table 6.1 recalls the general notations used throughout the chapter.

TABLE 6.1: Notations

Notation	Description
N	Number of FL clients
D_i	Local training data of Client$_i$
W	Global FL model
J	Number of layers in global FL model
W_i	Local model of Client$_i$
θ	Global model parameters
θ_i	Model parameters of Client$_i$
θ^j	Parameters of the j^{th} layer of global model
θ_i^j	Parameters of the j^{th} layer of Client$_i$'s model
p	The index of the private layer of a model, *e.g.*, θ_i^p are the parameters of the private layer of W_i
θ_i^{p*}	Non-obfuscated parameters of private layer of W_i stored on Client$_i$
η	Model learning rate
$\mathcal{L}$	Model loss function

6.3 Summary

In this chapter, we have presented DINAR, a novel method aimed to counter membership inference attacks. DINAR employs a straightforward yet efficient fine-grained strategy, focusing on safeguarding the model layer that is most vulnerable to privacy breaches related to membership information. This approach ensures robust and unobtrusive privacy protection in the context of Federated Learning. Additionally, DINAR addresses potential accuracy reductions in the protected model by harnessing adaptive gradient descent, thus maximizing the model's overall utility. In the upcoming chapter, we will undertake a practical assessment of DINAR's performance and compare it to state-of-the-art techniques.

Chapter 7

Experimental Evaluation of DINAR

7.1 Experimental Setup

This chapter presents evaluation results of DINAR. It first describes the used datasets and models, as well as the experimental setup. Then, the results obtained of our extensive experiments are presented, comparing DINAR with state-of-the-art mechanisms, and evaluating privacy, utility, and computational overhead.

7.1.1 Datasets and Models

To explore the wide-ranging performance of DINAR across different applications, we conduct experiments using a diverse set of datasets . Our evaluation encompasses four image datasets (Cifar-10, Cifar-100, GTSRB, and CelebA), tabular data (Purchase100, Texas100), and a raw audio waveform dataset (Speech Commands). We sum up these datasets in Table 7.1.

CelebA. CelebFaces Attributes Dataset is a large face images dataset, with 202,599 images for facial recognition and attribute detection. A subset of 40,000 images, resized to 64x64 pixels, was randomly selected. We create 32 classes by combining five pre-annotated binary facial attributes (Male, Pale Skin, Eyeglasses, Chubby, Mouth slightly Opened) for each picture [67]. The VGG11 architecture was employed for image processing [95].

Cifar-10 and Cifar-100. These are image dataset that consists of 60,000 images categorized into 10 classes for Cifar-10, and contains 100 classes for Cifar-100 [59]. These datasets encompass a wide range of objects such as airplanes, automobiles, birds, cats, and more. Each image in these datasets has a resolution of 32x32 pixels. For our experiments, we employ the ResNet-20 model.

Speech Commands. This dataset is a Google-released audio waveform for speech recognition classification [107]. It consists of 64,727 utterances from 1,881 speakers pronouncing 35 words (respectively 35 classes). Each audio record was transformed into a frequency spectrum with a duration of 1 second. For classification, we use the M18 classifier, a convolutional model with 18 layers and 3.7M parameters [27].

GTSRB. German Traffic Sign Recognition Benchmark dataset comprises 51,389 records across 43 classes, specifically designed for traffic sign recognition. It captures real-world traffic scenarios, including variations in lighting, weather conditions, and camera angles. This dataset is widely used for evaluating traffic sign recognition

algorithms and developing machine learning models for autonomous driving. We use VGG11 model architecture for this dataset [48, 95].

Purchase100. It is a tabular dataset adapted from Kaggle's "Acquire Valued Shoppers" challenge, consisting of 97,324 records with 600 binary features representing customer purchases. The goal was to classify customers into 100 types based on their buying behavior [93]. For modeling, we use a fully-connected neural network architecture with layers of sizes 4096, 2048, 1024, 512, 256, and 128, leveraging Tanh activation functions and a fully-connected classification layer [52].

Texas100. It consists in a tabular data sourced from the Texas Department of State Health Services, which encompasses information on inpatient stays across various health facilities. The dataset includes details like injury causes, diagnoses, performed surgeries, and general patient information (race, age, gender, ID, etc.). We use the same model as in Purchase100 for the classification task.

TABLE 7.1: Summary of used datasets

Dataset	#Records	#Features
Cifar-10	50,000	3,072
Cifar-100	50,000	3,072
GTSRB	51,389	6,912
Speech Commands	64,727	16,000
Texas100	67,330	6,170
Purchase100	97,324	600
CelebA	202,599	4,096

7.1.2 Software and Hardware Setup

In the following, we describe our experimental setup. We detail the used hardware and software environment, as well as the state-of-the-art attack and defense implementations we compare with. All the experiments are conducted on an NVIDIA A40 GPU. We use PyTorch 1.13 to implement DINAR, and the underlying classification models. For the state-of)the-art defense mechanisms based on differential privacy, we employ the Opacus library [116], as mentioned in subsection 7.1.3. To evaluate the resilience of FL defense mechanisms against membership inference attacks, we use an existing implementation of the attack that is based on a single shadow model [94]. We consider a FL system with 5 FL clients. Data are carefully divided into disjoint splits for each FL client, following a non-IID distribution, using dirichlet distribution. Each dataset is splited into 80% for training, and 20% for testing. The learning rate is set to 10^{-3} and the batch size to 64 for Resnet20, VGG, and M18 models on image and audio datasets. For FCNNs on tabular datasets, a learning rate of 10^{-4} is used and a batch size of 100.

7.1.3 Baselines

Our evaluation compares DINAR with different defense scenarios, including a no-defense baseline and three state-of-the-art solutions inspired by Differential Privacy. These solutions, LDP, CDP, and WDP, employ various approaches for privacy preservation. For LDP and CDP, we set the privacy budget parameter $\epsilon = 2.2$ and the probability of privacy leakage $\delta = 10^{-5}$, following the findings of [82]. In the case of WDP, a norm bound of 5 is considered, and Gaussian noise with a standard deviation of $\sigma = 0.025$ is applied. These settings ensure an optimal level of privacy preservation in our experiments.

7.1.4 Evaluation Metrics

DINAR aims to improve FL privacy and utility, without inducing additional costs. In the following, we define the performance metrics used to evaluate these different aspects.

Attack AUC. The attack success rate on a given model measures the percentage of successful MIAs conducted by an adversary. The attack AUC (Area Under the Curve) is a single value that measures the overall performance of the binary classifier implementing MIAs. The AUC value is within the range [50%–100%], where the minimum value represents the performance of a random MIA attacker, and the maximum value would correspond to a perfect attacker. The attack AUC is a robust overall measure to evaluate the performance of MIAs because its calculation involves all possible attacker's binary classification thresholds. Since the weakest (*i.e.*, most naive) MIA attacker would reach a minimum attack AUC of 50%, the best defense against MIAs would approach that optimal value of attack AUC of 50%. Thus, we use attack AUC as a means to evaluate the privacy of a model.

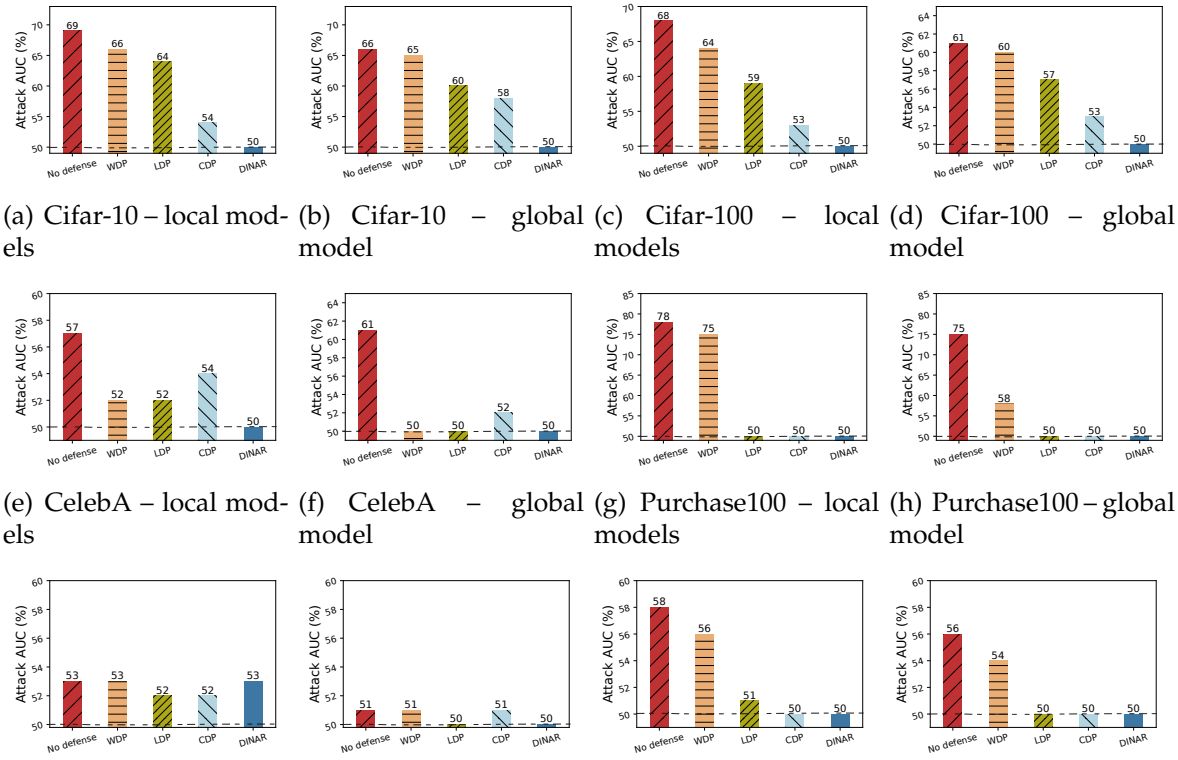
Overall Model Privacy Metric. In a FL system that consists of the global FL model M, and N clients models $M_1 \dots M_N$, we define a metric for measuring the overall privacy of all these models. Namely, we measure the highest potential privacy leakage from both the global model and clients' local models. Given the F_{AUC} function for computing the attack AUC of a model, the overall model privacy of the FL system is computed as follows:

$$Max\left(F_{AUC}(M), \frac{\sum_{i=1}^{N} F_{AUC}(M_i)}{N}\right)$$

Overall Model Utility Metric. We evaluate the utility of a protected model by measuring its accuracy, namely the ratio of correctly classified instances to the total number of instances. Considering DINAR's approach for protecting FL clients' models, we consider the average of accuracy of clients' protected models. Given N clients, M_i the model of each Client$_i$, and F_{Acc} the function that calculates accuracy of a model, the overall model utility metric is as follows:

$$\frac{\sum_{i=1}^{N} F_{Acc}(M_i)}{N}$$

Cost-Related Metrics. We also evaluate the additional costs that can be induced by a privacy-preserving FL mechanism, both in terms of execution times and memory usage. For instance, we measure the necessary time for a client to train a model during a FL round. We also measure the necessary time for the FL server to perform aggregation of client model updates. Finally, we measure the memory used by a client during model training,.



(a) Cifar-10 – local models (b) Cifar-10 – global model (c) Cifar-100 – local models (d) Cifar-100 – global model

(e) CelebA – local models (f) CelebA – global model (g) Purchase100 – local models (h) Purchase100 – global model

(i) GTSRB – local models (j) GTSRB – global model (k) Speech Commands – local models (l) Speech Commands – global model

FIGURE 7.1: Privacy leakage with DINAR and state-of-the-art protection mechanisms – The horizontal dashed line represents the optimal value of attack AUC (50%)

7.2 Experimental Results

7.2.1 Evaluation of Privacy Protection

We first measure the effectiveness of DINAR at countering membership inference attacks, *i.e.*, minimizing the overall model privacy metric described in §7.1.4 for attacks against both global and local models. The attacker runs a white box membership inference attack, as described [93]. For each dataset and respectively considered model, we compare DINAR with defense baselines.

We partition the data into training and test sets, with the attacker's prior knowledge corresponding to half of these datasets. The success of the membership inference attack is then assessed on the remaining half. We systematically evaluate

DINAR, *WDP*, *LDP*, and *CDP* on both local and global models, considering the utility and the membership inference attack AUC.

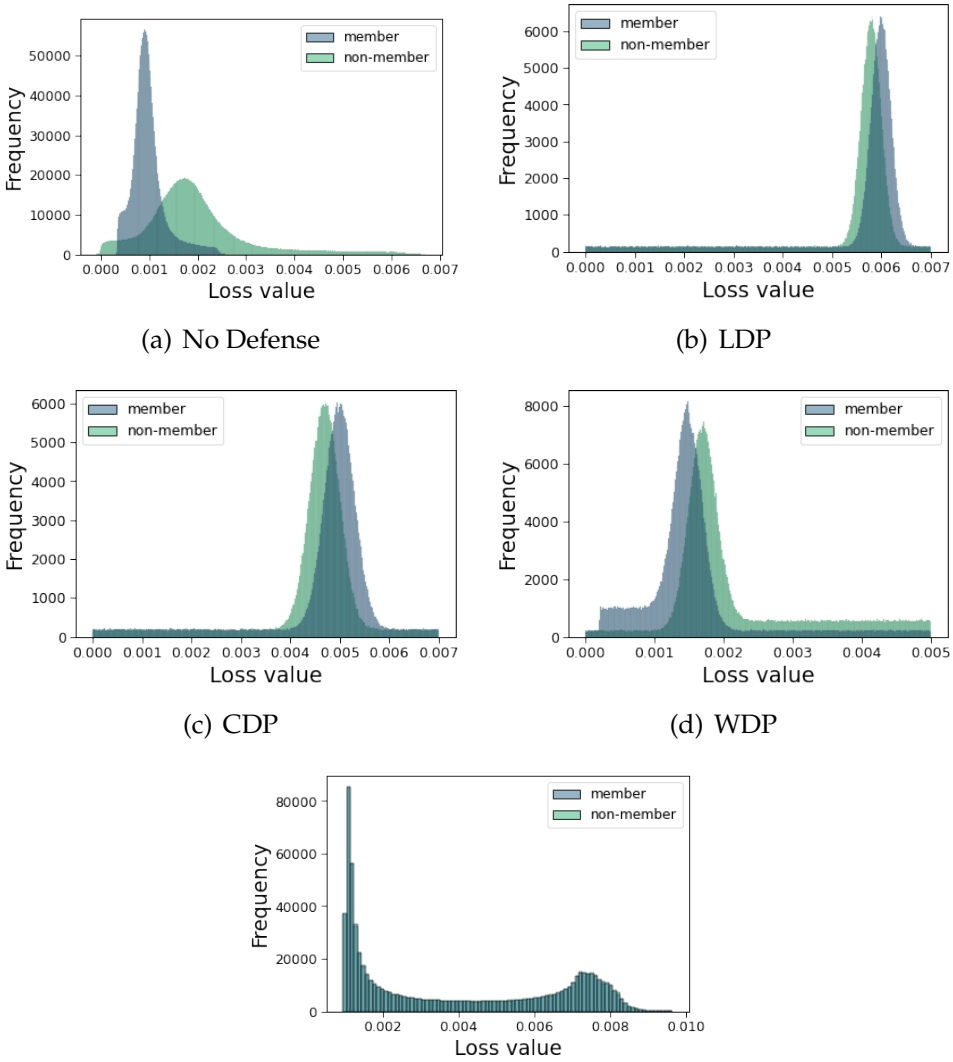
In Figure 7.1, we first plot distinctly the average attack AUC against local models and the attack AUC against the global model. In all plots, each bar represents one defense scenario amongst the baselines we consider. Our results show that DINAR exhibits privacy mitigation rates that closely approach the 50% mark across all datasets, indicating a strong level of privacy protection. This holds true for both global and local model inference attacks, while differencial privacy mechanisms are less constant at protecting the models. It is worth noting that DINAR achieves reducing the privacy leakage of local models by 29% in the best case, as shown in Figure 7.1(a), while differential privacy reveals its limits in that case : WDP only reduces the privacy leakage by 3% and in the best case, and even CDP is worse than DINAR by only reducing it by 25%.

By concealing sensitive layers and replacing parameters by random values, DINAR enables the perturbation of the attacked model outputs as received by the attacker, thereby mitigating membership inference attacks. Indeed, the attacker receives an altered version of the model with randomized layer parameters; when the attacker tries to reproduce the behaviour of the target model, the randomization necessarily impacts the outputs of the model, which makes them barely comparable to the outputs of the shadow model. This counters the logic of membership inference attacks and explains the significant drop of the attack AUC. These promising results underscore the potential of DINAR as an effective privacy-preserving technique, particularly in scenarios where differential privacy methods may have limitations.

7.2.2 Analyzing Impact on Model Loss and Utility

In order to provide an insight on DINAR's ability to preserve both privacy and model utility, we analyze the impact of DINAR and the considered baselines on the behavior of protected models. We evaluate the effectiveness of each defense technique in reducing loss distribution discrepancies beween member and non-member data records and minimizing significant loss values. Ideally, the loss distribution of members and non-members should match, indicating similar loss the model's lack of insightful information to distinguish members and non-members. Then, a distribution with mostly low loss values indicates model accuracy. We measure the loss of the attacked model separately for member and non-member records in each defense scenario for the Purchase100 dataset and its respective FL configuration.

Figure 9.5 plots the loss distribution for members and non-members for each defense scenario. Comparing the results, we gain scientific insights into the mechanisms' performances. Our findings support previous sections' limitations of existing techniques (LDP, CDP, WDP), which focus solely on minimizing loss between member and non-member data. Although the curves match better than in a no defense scenario, meaning that it is harder to distinguish members from non-members, these approaches often result in larger loss values, compromising the overall utility and accuracy. The loss peaks are between 0.001 and 0.006 for differential privacy defenses, while DINAR achives reducing the loss under 0.001 for most records. Moreover, DINAR shows a strong distribution match between member and non-member records, while there's a neat discrepancy for differential privacy based defenses.



(a) No Defense (b) LDP

(c) CDP (d) WDP

(e) DINAR

FIGURE 7.2: Model loss distribution in different FL defense scenarios. The dark curve shows the loss distribution for member records, and the light curve shows the distribution for non-members

Thus, the combination of adaptive training and layer obfuscation in DINAR clearly demonstrates the effectiveness of the proposed approach. Indeed, this approach narrows the gap between member and non-member data distributions, while minimizing substantial loss occurrences.

7.2.3 Analyzing Privacy *vs.* Utility Trade-off

With the objective of empirically confirming the insights revealed in §7.2.2 on DINAR's ability to balance both privacy and model utility in a FL system, we evaluate its impact on local models behavior. We conduct the experiments on different datasets, by running the same attack scenario as the one presented in §7.2.1, introducing the consideration of both privacy and model utility metrics. To assess the

balanced efficiency of DINAR, we expect it to both maximize the local models' accuracy and minimize the membership inference attack AUC.

Figure 7.3 shows our results by plotting both metrics on two axes: the x axis represents the average local model accuracy, while the y-axis plots the overall attack AUC we previously defined. In a best-case scenario, the dot should be located in the bottom-right corner of each plot, meaning that the effective defense mechanism both preserves the model accuracy and decreases the attack AUC to 50% We observe that *WDP*, *CDP* and *LDP* achieve reasonable attack mitigation but often reduce model utility. For example, on the Cifar-10 dataset, *WDP* reduces attack AUC by 3%, while *CDP* reduces it by 6% but with a significant 20% drop in model utility. In exchange, DINAR reduces the attack AUC by 29%, obtaining an optimal privacy, and the model accuracy drop is inferior to 1%.

In most cases, DINAR strikes a balance between privacy preservation and utility, emerging as a compelling solution. It achieves comparable attack mitigation to the undefended model while maintaining preserving the model accuracy in most cases. Notably, on the Speech Commands dataset, DINAR achieves the same level of attack mitigation as the undefended model and surpasses the baseline accuracy with a 90% model accuracy compared to the baseline's 86%. Indeed, DINAR outperforms its competitors by obfuscating layer parameters, while preserving local accessibility to the original values. This allows clients to maintain model quality without sacrificing privacy, leading to improved model accuracy compared to other privacy-preserving mechanisms that introduce noise to all model parameters. Thus, the approach followed by DINAR demonstrates its effectiveness in mitigating attacks, while preserving model utility, making it a promising solution for privacy-preserving FL systems.

7.2.4 Ablation Study of DINAR

In accordance with the details provided in §6.2.6, one of our objectives is to address the overfitting problems caused by DINAR by employing adaptive gradient descent techniques during the training of local models. Table 7.2 presents a comparison of the average accuracy achieved by local models, with and without adaptive gradient descent. In each scenario, we train 10 local models on Purchase100, utilizing a six-layer fully-connected neural network. Within each scenario, we measured the highest attack area under the curve (AUC) value against both the globaland local models. We also record the corresponding model accuracy for each scenario.

TABLE 7.2: Performance of DINAR with and without adaptive model training

Performance	No defense	DINAR w/o adapt. train.	DINAR w/ adapt. train.
Model accuracy	61%	59%	62%
Local model privacy	78%	50%	50%
Global model privacy	74%	50%	50%

With adaptive training, we achieve the best model utility across all clients, resulting in a 3% improvement in accuracy compared to the use of Adam. Further,

combining DINAR with adaptive training or not, has no significant impact on the objective of privacy preservation for both global and local models, as indicated by the results. In all the considered scenarios, the attack AUC remains close to 50% against both global and local models. This finding aligns with the observations made by [3], where AdaGrad outperforms Adam in an FL system involving a substantial number of FL rounds and a reasonable number of local training epochs (in our case, 10 epochs). Consequently, DINAR, in conjunction with adaptive training, achieves the highest accuracy for local models without compromising privacy protection.

7.2.5 Cost of Privacy-Preserving Mechanisms

With the objective of tackling state-of-the-art differencial privacy based mechanisms issues regarding coputational costs, we rigorously analyze the overheads of DINAR, a pioneering solution for addressing the high computation costs associated with differential privacy in federated learning. We meticulously evaluate DINAR's performance across key metrics, including average training duration, server aggregation duration, and peak GPU memory usage. Comparisons are made against carefully defined baselines, allowing for a comprehensive assessment of DINAR's efficiency. We compare the costs of different defense mechanisms for the FL training scenario using the GTSRB dataset with VGG11, following the FL system setup described in §7.1.2.

Model Training Time. We examine different scenarios to evaluate the average training duration per round for individual clients in federated learning. This duration refers to the total time required for all the local training epochs of a client during a round.

The impact of privacy mechanisms like LDP, CDP, and WDP on the training duration is depicted in Figure 7.4(a). Interestingly, our analysis shows that incorporating privacy-preserving techniques has a noticeable negative effect on the overall training duration. Despite the improvements made by the Opacus framework in speeding up differential privacy, there is still a significant cost. In the worst-case scenario, adding noise results in a training duration increased by 36%.

However, it is important to highlight that DINAR effectively addresses the computational overhead associated with differential privacy without compromising system performance. DINAR successfully mitigates the issue of increased training duration by obfuscating layers, which does not introduce any additional computational overhead. This ensures the system remains efficient and effective while maintaining privacy.

FL Aggregation Time. We conduct measurements to determine the average duration for server aggregation in various scenarios. This involved tracking the time taken from when the server received all weights to be aggregated until it sent the aggregated weights. Notably, the use of CDP resulted in a significant increase in aggregation duration, reaching up to 30 times longer for GTSRB with VGG. This prolonged duration can be attributed to CDP's design principle, which involves introducing noise to the parameter aggregate before transmission to clients. This process substantially extends the time required for aggregation, measured in seconds in our case. However, when employing DINAR, *LDP*, and *WDP*, the durations exhibit

similar orders of magnitude compared to the scenario without any baseline. This suggests that these privacy mechanisms do not impose a substantial additional cost in terms of aggregation time, presenting a more efficient alternative.

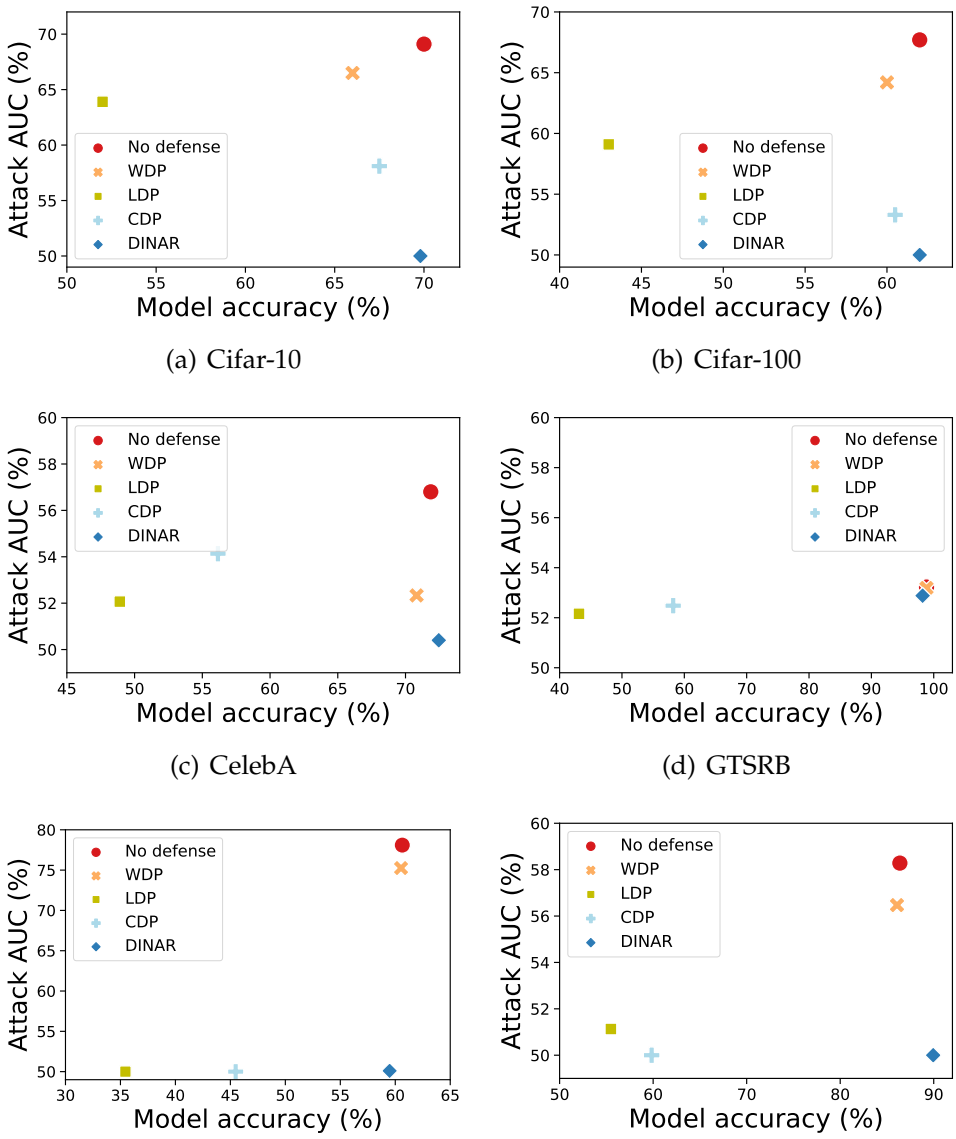
Memory Usage. Our study delves deep into the realm of GPU memory usage in privacy-preserving federated learning, unraveling captivating insights. Through meticulous analysis, we unveil the impact of various privacy mechanisms, including LDP, CDP, and WDP, on memory consumption during local model training. Our findings paint a compelling picture, showcasing a systematic increase in GPU memory usage with the implementation of these privacy measures. They show that in that case, running differential privacy algorithms increases the GPU Memory usage by 168% compared to a no defense scenario. In exchange, DINAR doesn't introduce any computational comparable operation by definition, resulting in having no significant impact on GPU memory usage.

First, the addition of calibrated noise, a fundamental technique in differential privacy, requires storing the noise values, which increases memory usage. Second, tracking and managing the privacy budget, which represents the maximum allowable privacy loss, necessitates additional memory to maintain the budget information. Lastly, the need for maintaining an aggregation buffer to collect model updates before applying privacy mechanisms adds to the memory requirements. This reasonably explains why DINAR is optimal from the perspective of GPU memory in comparison with differential privacy, as it doesn't involve noise addition nor privacy budget management.

7.3 Summary

In this chapter, we assess the performance of DINAR, an inventive defense approach, aimed at mitigating membership inference attacks. Our practical investigation, conducted across diverse datasets, neural network architectures, and cutting-edge FL privacy protection mechanisms, highlights the effectiveness of DINAR in terms of enhancing privacy, preserving utility, and minimizing associated costs.

Furthermore, apart from bolstering FL defenses against membership inference attacks, we foresee that DINAR can also prove valuable in safeguarding against various other privacy threats, including property inference and model inversion attacks. Additionally, an intriguing avenue for future research involves developing methods to automatically identify the neural network layers most susceptible to privacy breaches, tailored to specific threat models, privacy attack types, and FL model structures.

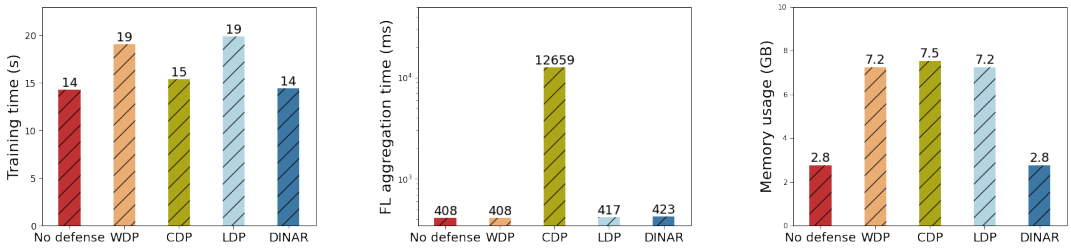


(a) Cifar-10 (b) Cifar-100

(c) CelebA (d) GTSRB

(e) Purchase100 (f) Speech Commands

FIGURE 7.3: Trade-off between privacy and utility in different FL defense scenarios



(a) Client-side model training time (b) Server-side FL aggregation time (c) Client-side memory usage

FIGURE 7.4: Cost of FL privacy-preserving mechanisms in terms of model training time, FL aggregation time, and memory usage

Chapter 8

PASTEL: Mitigating Membership Inference Attacks in Federated Learning

8.1 *PASTEL* Objectives

Although Federated Learning has brought a significant breakthrough in ML privacy by decentralizing the participants' data, recent works show that FL systems remain sensitive to a wide range of privacy attacks [93, 84, 69]. Indeed, a malicious participant may infer some private and potentially sensitive information about another participant's data, by analyzing the model parameters. More specifically, in this work, we are interested in membership inference attacks (MIA), where a malicious participant tries to infer whether a data sample was used for training the model [93].

In this chapter, we propose *PASTEL*, a novel privacy-preserving mechanism that allows FL systems to be resilient to membership inference attacks (MIAs). Roughly speaking, MIAs are based on a binary classifier that is able to differentiate between member data samples used to train a model and non-member data samples not used for training. *PASTEL* proposes a novel multi-objective learning function. On the one hand, *PASTEL* reduces model loss and leverages adaptive gradient descent optimization for higher model accuracy, and on the other hand, it decreases the generalization gap to reduce the difference between member data and non-member data. Indeed, recent works showed that sensitive information about model training data can be located in some layers of neural networks [79, 80], and inferred from the layers' gradients. Thus, *PASTEL*'s primary motivation is to minimize the internal generalization gap during the training of the FL model, to effectively protect private information, and consequently reduce the MIA success rate. And thanks to its multi-objective approach, *PASTEL* is, as far as we know, the first FL defense mechanism that counters membership inference attacks while maintaining high model accuracy, with negligible computational overheads, thus, resulting in an effective solution for ubiquitous computing systems. In this chapter, we highlight the following key contributions:

8.2 Problem illustration

Although federated learning is privacy-preserving by design, it remains vulnerable to different types of inference attacks, such as membership inference attack [93] that

aims to determine whether a specific data record is used for training a target model. For instance, it can be used to infer whether the records of a specific patient have been used to train a classifier related to a certain disease. Property inference is another privacy attack that aims to extract dataset properties [6]. In particular, these properties might be irrelevant to the training task. For example, when the main task is to train a model for race or gender recognition, the property inference attack may intend to infer whether people in the training images wear glasses or not. As another type of privacy attack, there are model inversion or attribute inversion attacks [47, 46] that fall in the category of reconstruction attacks, where given output labels and partial knowledge of some features, try to recover sensitive features or full data samples.

In this work, we are interested in the membership inference attack (MIA), a privacy attack that aims to determine if a specific data record is used in the training of the target model. The authors in [93] introduce a black-box attack that relies on the output class probability distribution of the model. In this scenario, the attacker trains one or several shadow models to generate model probabilities per class, which is then used to train multiple attack models (one for each class). Using confidence scores as inputs, these attack models output the membership status of the given record as shown in Figure 8.1-❶. An extension of [93] attack is proposed in [91] which is based on a single shadow model and relaxes the assumption that the shadow model is constructed the same way as the target model. As well as [103], the authors extended the membership attack presented in [93] to a more general setting and showed that membership inference attacks are data-driven and largely transferable. The authors in [75] investigated the membership privacy leakage from two aspects: embedding layers and gradients. It was shown that the non-zero gradients of the embedding layer of a deep learning model can reveal the positions of the words in a training batch. This enables an adversary to conduct a membership inference attack.

Membership inference attacks exploit the information leakage of machine learning algorithms about their training data through the learned model. Hence, they have been investigated as an indicator that reveals the privacy leakage of federated learning models. They present a significant threat to the privacy of individuals whose data is used to train machine learning models. For example, if an attacker is able to determine that a person's data was used to train a model, he may be able to infer sensitive information about that person, in smart health ubiquitous applications, membership inference attacks can infer that the owner of a clinical record has the disease based on the fact that the clinical record was used to train a model based on the model prediction as described in Figure Figure 8.1-❷. A recent report [100] published by the National Institute of Standards and Technology (NIST) mentions explicitly that a membership inference attack identifying whether an individual has been included in the dataset used to train the target model is a privacy violation.

On the Difficulty of Mitigating Membership Inference Attacks in Edge Federated Learning Differential privacy has been widely used as a framework for privacy-preserving machine learning, providing statistical guarantees against the information an adversary can infer through the output of a randomized algorithm. In the following, we describe a case that illustrates the problem of privacy leakage

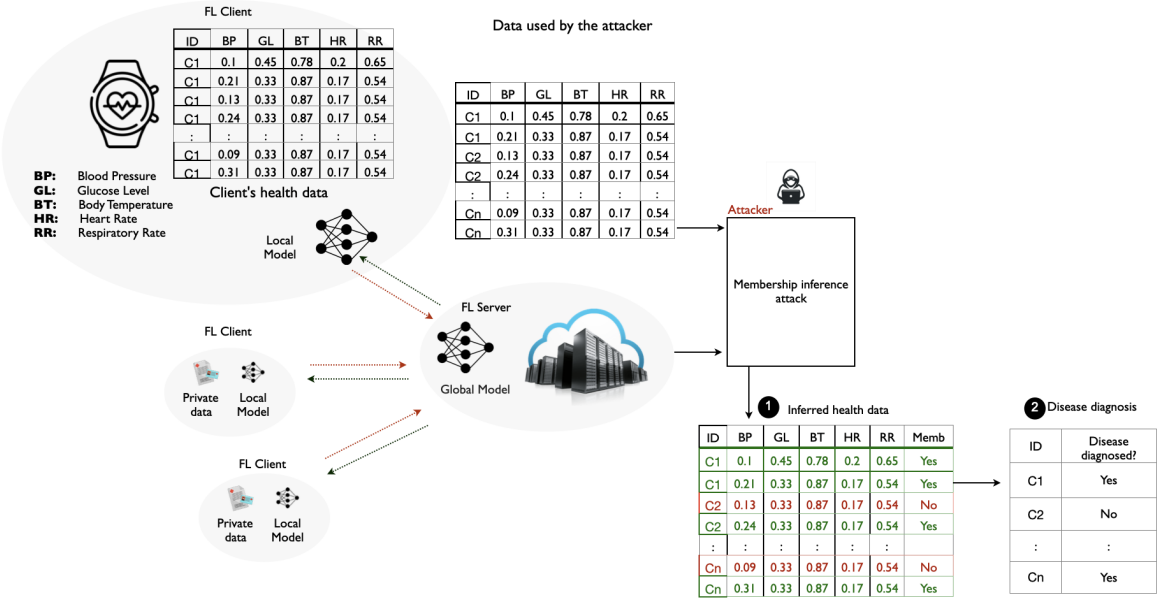
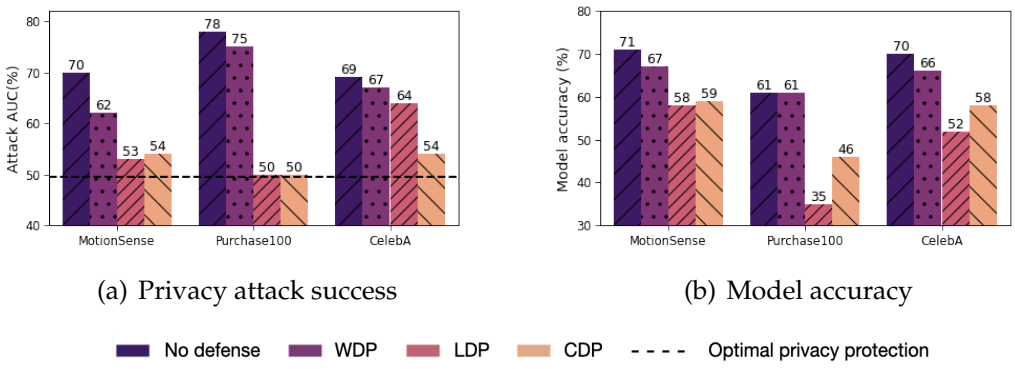


FIGURE 8.1: Membership inference attack in healthcare applications

and the limitations of State-of-the-art systems in FL-based computing systems, in healthcare applications, computer vision and e-commerce applications. We focus on white-box inference membership inference attack proposed by [83] for classification tasks. To evaluate privacy leakage in healthcare applications, we used MotionSense dataset [70], which includes time-series data generated by accelerometer and gyroscope sensors (attitude, gravity, user acceleration, and rotation rate), For MotionSense we consider the classification task of determining the patient activity, for Purchase100 we train a classifier for determining the client type based on his purchases, and finally for CelebA the task consists in face attributes classification. We evaluate the attack with 3 differential privacy techniques namely WDP, LDP and CDP. The results for privacy leakage and model utility are presented in Figure 8.2.



(a) Privacy attack success (b) Model accuracy

FIGURE 8.2: Privacy leakage from edge computing image analysis – Impact on FL clients' models protected with existing FL privacy-preserving mechanisms. The dashed line in first plot indicates the optimal privacy value.

Weak diffential privacy fails to protect against membership inference attacks since it results in very large values of ϵ, as it adds noise at every round ignoring the noise added in previous rounds. More specifically, in DP, the concept of composability ensures that the joint distribution of the outputs of differentially private mechanisms satisfies DP [74]. Therefore, if we assume that, at every round, the server applies an ϵ-differentially private mechanism on participants' updates, then this weak DP mechanism results in spending $r \times \epsilon$ privacy budget after r number of rounds. This yields larger values of ϵ, and thus significantly less privacy for participants. Concerning central differential privacy and local differential privacy, they provide to be efficient in mitigating membership inference attacks as shown in Figure 8.2, the attack AUC is reduced significantly. However, CDP and LDP come at a cost of sacrificing the performance of the model (it decreases from 60.6% to 35.4% with Purchase100 with LDP). Moreover, the main issue of differential privacy is the computation time, as more calculations are needed to add noise and other privacy-preserving operations.

8.3 System Model and Problem Formulation

In the following, we present the problem definition, the underlying threat model and the defense model. Notations used in this chapter are detailed in Table 8.1.

8.3.1 Threat Model

We consider the standard setting of a white-box Membership Inference Attack. In the federated learning system configuration, we consider a malicious participant can be either on server-side or client-side.

Attacker's Objective. If the attacker is on server-side, its goal is to determine whether a data record r_j is likely to have been used for training the model W_i, *i.e.*, if $r_j \in D_i$ (member) or not (non-member). If the attacker is on client-side, its goal is to determine whether r_j is likely to have been used for training one of the other clients models, ignoring which client.

The attacker performs the membership inference attack described in [93] by training a multilayer perceptron attack model W_{attack} with binary output. In order to determine whether a record r_j was most likely used to train W_{target} or not, the attack model W_{attack} returns 1 (member) or 0 (non-Member).
The attacker first creates a shadow model M_{shadow} using the same architecture as the target model, considering the dataset $D_{shadow} = D_{shadow_{train}} \cup D_{shadow_{test}}$. The shadow model is trained on $D_{shadow_{train}}$, then the attacker creates the attack model training dataset D_{attack}, such as :

$$D_{attack} = D_{attack_{Member}} \cup D_{attack_{NonMember}} \tag{8.1}$$

$$D_{attack_{Member}} = (\underbrace{p_{shadow}(D_{shadow_{train}})}_{features}, \underbrace{1}_{label}) \tag{8.2}$$

$$D_{attack_{NonMember}} = (\underbrace{p_{shadow}(D_{shadow_{test}})}_{features}, \underbrace{0}_{label}) \tag{8.3}$$

The attacker trains the M_{attack} model on D_{attack}. The output prediction of W_{attack}, taking $p_{target}(r_j)$ as input, is 1 (Member) if the record r_j has most likely been used to train M_{target}, and 0 (Non-Member) otherwise. The final task of the attacker is to maximize the AUC of W_{attack} while testing it on the p_{target} data.

Attacker's Capabilities. We make the assumption that the attacker has access to a data sample D_{shadow}, using the same features and labels as the dataset D_{target} used to train the target model. The attacker also knows the architecture of the target model.

In a case where the attacker is on server-side, it has access to parameters update W_i sent by client i with $i \in 1\ ;\ N$, and can identify which client sent the parameters update. This scenario is only possible in case there is no secure aggregation [15] enabled in the FL system, as detailed in 8.3.2. In a case where the attacker is on client side, it has access to the aggregated parameters W sent by the server.

TABLE 8.1: Notations

Notation	Description
W	Global FL model parameters
W_i	Local model parameters of client i
$(\mathcal{X}, \mathcal{Y})$	Training data of a client
$(\mathcal{X}', \mathcal{Y}')$	Non-member data of a client
$\mathcal{M}n\mathcal{M}$	Distance function between member data and non-member data
JSD	Jensen-Shannon divergence
KL	Kullback-Leibler divergence
$G_{l_j}(X)$	Gradients of layer l_j of member data $\mathcal{X}$
$G_{l_j}(X')$	Gradients of layer l_j of non-member data $\mathcal{X}'$
$G_W(X)$	Gradients of model W of member data $\mathcal{X}$
$G_W(X')$	Gradients of model W of non-member data $\mathcal{X}'$

8.3.2 Defender's Assumptions

Defender's Objective. We aim to design a FL defense mechanism that achieves privacy preservation against Membership Inference attacks from malicious participants, without sacrificing the local FL model's quality. We aim to fulfil the two following properties:

- *Privacy.* The defender needs to ensure that its model is protected against Membership inference attacks, *i.e.*, that the attack model's accuracy is as close as possible to 50% (best case scenario for the defender).

- *Utility.* The defender must ensure that, by defending itself, it does not impact its own model utility nor other participants' model utility through updates.
- *Overhead* The overhead costs of our solution in terms of computational performances, versus a non-defense scenario, must be as low as possible.

Defender's Capabilities. We assume the defender has no knowledge about the attacker's strategy, as in [24]. FL attack mitigation is accomplished at the client-side, whereby the training protocol integrates a regularization term in the loss function to minimize the similarity between the feature distributions of member and non-member instances, thus enhancing the generalization performance of the model against potential FL threats. Contrary to several state-of-art privacy-preserving mechanisms [14, 19, 99], our mechanism does not require an analysis of individual clients model updates before aggregating, thus it is compatible by design with secure aggregation [15].

8.4 Design Principles of *PASTEL*

This section presents *PASTEL*, a federated learning framework for membership inference attack mitigation. First, §8.4.1 provides an overview of *PASTEL*. Then, §8.4.2 presents its design principles in detail. Finally, §8.4.3 analyzes the key properties of *PASTEL*.

8.4.1 Overview of *PASTEL*

We present *PASTEL* (*P*riv*A*cy pre*S*erving federa*TE*d *L*earning), a local side privacy protection scheme that counters membership inference attacks in FL systems, without breaking secure aggregation guarantees, nor deteriorating the performance of the FL task. The objective of *PASTEL* is to provide the best trade-off in terms of privacy/utility: (i) Privacy: mitigate membership inference attack by limiting the information shared with the server (ii) Utility: keep the same performance for the local models. *PASTEL* addresses the threat model introduced in §8.3.1, and fulfills the defense objectives presented in §8.3.2.

The detailed pipeline of *PASTEL* is defined in Figure 8.3. *PASTEL* is on the client side, *i.e.*, the entire process is fully disclosed to the FL server. *PASTEL* is designed to reduce the generalization gap, which is the difference between the model's performance on the training data and its performance on unseen data. This gap can be exploited by an adversary to infer whether a particular record was used during the training of the model, which can compromise the privacy of the individual. The workflow of *PASTEL* is illustrated in Figure 8.3. During the training process, each FL client considers minimizing the loss function based on the model output and the real label, and a novel loss function to protect privacy and improve the model's ability to generalize to unseen data, while also making it harder for an attacker to infer membership information. Furthermore, *PASTEL* applies adaptive gradient descent to further improve client model accuracy. It adapts the learning rate for each parameter based on the history of its gradient. This helps to prevent overfitting and distortion of the model by providing a finer-grained update scheme for each parameter.

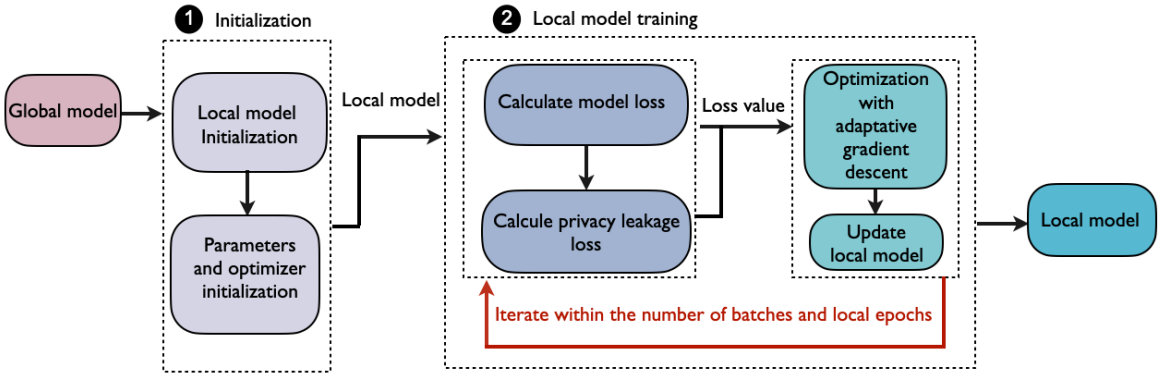


FIGURE 8.3: *PASTEL* pipeline at the client-side

8.4.2 Design Principles of *PASTEL*

Membership inference attacks (MIAs) rely on the overfitting of deep learning model [93]. Intuitively, the generalization gap has been used to mount MIAs, and [64] shows a strong correlation between them. In particular a model with large generalization gap is more vulnerable towards MIAs. A model generalization gap g is defined to be $g = a_M$ - a_{NM}. where a_M is the model accuracy on training data, *i.e.*, member data and a_{NM} is the model's accuracy on a dataset drawn from the same distribution as the training data *i.e.*, non-member data. Moreover, [108] shows that the generalization gap on the hidden layers, defined as internal generalization gap, is more important than the output layer. The internal generalization gap is measured based on the divergence of member features and non-member features on the hidden layers. *PASTEL* focuses on the internal generalization gap. To evaluate the distribution shift between member and non-member features, we consider a distance function that we refer to as $\mathcal{M}n\mathcal{M}$ (*i.e.*, member vs. non-member) function, with the goal of minimizing the distance, *i.e.*, reducing the generalization gap. In the following, we use Jensen-Shannon divergence (JSD) as an instance of $\mathcal{M}n\mathcal{M}$ function [76]. Indeed, JSD is a widely used measure of similarity between probability distributions, that is robust and less impacted by outliers and noise than other distance measures such as Euclidean distance or cosine similarity.

The primary objective of *PASTEL* is to reduce the divergence between the gradients of the latent layers of the model for member and non-member data. The Jensen-Shannon divergence (JSD), which is a symmetrized and smoothed variant of the Kullback-Leibler divergence (KL), is harnessed for this purpose. KL quantifies the information loss that results when a probability distribution q is used to approximate another distribution p. JSD, a derived metric, embodies the amalgamation of these divergences.The rationale for employing JSD is rooted in its capacity to measure the dissimilarity between two probability distributions. In the context of *PASTEL*, JSD facilitates the assessment of discrepancies in prediction distributions vis-à-vis a reference dataset. By introducing an element of uncertainty into the inference process, *PASTEL* endeavors to obfuscate the determination of data point membership, thereby conferring enhanced privacy.

Furthermore, JSD is a smoothed and symmetrized variant of KL, and it is calculated based on the latter. JSD provides an effective way to compare and assess

the dissimilarity between two probability distributions. Unlike KL, JSD is symmetric, meaning that the order of the distributions being compared does not affect the result. This symmetry property is advantageous because it ensures that the comparison is unbiased and accounts for both directions of divergence. In *PASTEL*, JSD is harnessed to introduce uncertainty into the inference process. By comparing the prediction distributions of member dara points with non member data points using JSD, *PASTEL* adds a layer of complexity that challenges adversaries attempting to discern whether a specific data point was part of the training dataset.

Formally, let us consider a deep learning model which parameters are denoted as $\mathcal{W}$, comprising n layers $l_1, l_2, ..., l_n$. Gradients pertaining to a given layer l_i within a batch of member data denoted as X, are represented by $G_{l_1}(X), G_{l_2}(X), ..., G_{l_n}(X)$ and model gradient by $G_{\mathcal{W}}(X)$. In the same vein, gradients of layer l_i resulting from a batch of non-member data $X^{'}$ are denoted as $G_{l_1}(X^{'}), G_{l_2}(X^{'}) \dots G_{l_n}(X^{'})$ and model gradient by $G_{\mathcal{W}}(X^{'})$. To compute JSD, we first compute KL between member and non-member gradient distributions, and then KL between non-member and member gradient distributions as defined in respectively Eq. (8.4) and Eq. (8.5) as follows:

$$KL_m = KL(G_{l_i}(X^{'})||\frac{G_{l_i}(X) + G_{l_i}(X^{'})}{2}) \tag{8.4}$$

$$KL_{nm} = KL(G_{l_i}(X)||\frac{G_{l_i}(X) + G_{l_i}(X^{'})}{2}) \tag{8.5}$$

JSD is the average of the two KL distances defined in the previous Eq. (8.4) and Eq. (8.5) and is computed as follows:

$$\min_{i \in \{1..n\}} JSD(G_{l_i}(X)||G_{l_i}(X^{'})) = \frac{1}{2}(KL_m + KL_{nm}) \tag{8.6}$$

And the KL divergence between two distributions p and q, denoted as $KL(p||q)$, is measured as follows:

$$KL(p||q) = \sum_c p_c \log \frac{q_c}{p_c} \tag{8.7}$$

The overall loss of the model, denoted as L, is calculated based, on the one hand, on L_{priv}, the novel privacy leakage loss that makes use of JSD loss, and on the other hand, on labels loss L_{acc} which is the loss between the model output and the real label Y of an input batch X. The detailed algorithm of *PASTEL* is described in §4. It follows a multi-step process to optimize the model's performance. Firstly, the algorithm assesses how well the model is currently performing on the training data and identifies misclassified examples. Next, the algorithm calculates the label loss to determine the model's accuracy on the labeled training data and aims to improve it. Finally, *PASTEL* algorithm measures the similarity between the probability distributions of the training data and a data sample unseen by the model during training, using the $\mathcal{M}n\mathcal{M}$ function as a loss function criterion to improve the model's generalization ability. We compare the output probability distribution for, on the one hand, the training data X (*i.e.*, member data) of client i, and on the other hand, unseen data $X^{'}$ (*i.e.*, non-member data) of client i, in order to minimize the distance

between the two. By optimizing the model's performance based on these insights, the algorithm aims to improve its accuracy and generalization ability. This approach offers a novel way to optimize the performance of deep learning models by focusing on both the accuracy on the training data and the generalization ability to unseen data. The results of our experiments demonstrate the effectiveness of the proposed algorithm in improving the performance of the model on a wide range of datasets.

In a real-world setting, there are several means for FL clients to obtain non-member data, i.e., data that are not used by clients for model training. For instance, in ubiquitous computing systems, data sampling is a core process to determine the volume of collected data to be actually used for a given service [11], [2]. Thus, part of data samples that are collected but not used for the service (*i.e.*, not used for FL client model training) can be used as non-member data for a FL client running *PASTEL*. Another means to get non-member data is to use generative model for producing synthetic data [55].

Algorithm 4: *PASTEL* algorithm on FL client i

Inputs:
$\mathcal{W}$: Global FL model parameters
$(\mathcal{X}, \mathcal{Y}) = \{(X_1, Y_1), \ldots, (X_k, Y_k)\}$: Client's training data, i.e., member data
$(\mathcal{X}', \mathcal{Y}') = \{(X_1', Y_1'), \ldots, (X_k', Y_k')\}$: Client's non-member data

Output: $\mathcal{W}_i$: Client's model parameters

```
// Initialization
```
$\mathcal{W}_i \leftarrow \mathcal{W}$

foreach *local training epoch* **do**
  foreach $(X_j, Y_j) \in (\mathcal{X}, \mathcal{Y})$ **do**
    `// Perform forward pass`
    $\widehat{Y}_j \leftarrow \mathcal{W}_i(X_j)$
    `// Compute model loss`
    $L_{acc} \leftarrow \mathcal{L}(Y_j, \widehat{Y}_j)$
    `// Compute privacy leakage loss`
    $L_{priv} \leftarrow JSD(G_{W_i}(X_j), G_{W_i}(X_j'))$
    `// Compute gradient`
    $\nabla_t \leftarrow AGD(L_{priv} + L_{acc}, \mathcal{W}_i)$
    `// Update local model`
    $\mathcal{W}_i \leftarrow \mathcal{W}_i + \nabla \mathcal{W}i$

return $\mathcal{W}_i$

Improving Accuracy with Adaptive Gradient Descent. In addition to countering MIAs, *PASTEL* aims to improve the model utility FL clients, *i.e.*, by maximizing client model accuracy as defined in §9.1.5. Thus, for each client i, the loss function $\mathcal{L}$ of its local model W_i. The loss function $\mathcal{L}$ is the summation of errors of W_i for each sample of the models training and testing set. It $\mathcal{L}$ is computed with the cross entropy function described in Eq. (8.8), with M the number of output classes of the model, a a binary indicator equals to 1 if class label c is the true class and 0 if not

for observation o, and b is the probability that observation o is of class c. The goal of model training is to minimize L.

$$\mathcal{L}_i = -\sum_{n=0}^{M} a_{o,c} \log b_{o,c} \tag{8.8}$$

To minimize $\mathcal{L}$, W_i updates its parameters at each training round, according to the learning rate hyperparameter η, a coefficient multiplying the computed gradient values at each FL round, such as $\eta \in [\![0, 1]\!]$. The learning rate is one the main impacting factors regarding model accuracy and loss convergence issues in ML, as shown by [96]. A too small learning rate value can cause overfitting, while a too high value may cause model accuracy instability, despite of improving the model training speed.

The adaptive gradient descent technique allows to address issues related to local minima and saddle-points [97, 3], and provides stronger guarantees against overfitting, as explained in section 6.2.6. First, when it comes to training complex models such as CNNs on a high number of iterations, this technique mitigates overfitting risks as the algorithm converges slower than Adam and RMSProp, notably in its first iterations, as shown by [87]. Second, given the high-dimensional properties of neural network optimization problems, this adaptive gradient descent method also presents the advantage of iteratively adjusting the learning rate separately for each dimension. Eq. (8.11) describes adaptive gradient descent, Eq. (8.9) initializes the variable v to zero. v is used to accumulate the squared gradients over time. Eq. (8.10) updates v in each iteration $t + 1$ by adding the squared gradient of the loss function $\mathcal{L}$ with respect to the model parameters $W_{i,t}$. The gradient is squared to emphasize larger gradients and dampen smaller ones. Eq. (8.11) updates the model parameters W_i for the i_{th} parameter in iteration $t + 1$. As the denominator in Eq. (8.11) is a sum of square gradients increasing at each epoch, the algorithm will then attenuate the parameter updates with a too large delta, and accentuate the updates with a too small delta. This property allows to tackle saddle points and local minima, and smoothes the convergence of $\mathcal{L}$ over all its dimensions. Adding a constant to the denominator in Eq. (8.11) prevents from divisions by zero at first iteration t_0.

$$v_0 = 0 \tag{8.9}$$

$$v_{t+1} = v_t + \nabla_W \mathcal{L}(x, W_{i,t})^2 \tag{8.10}$$

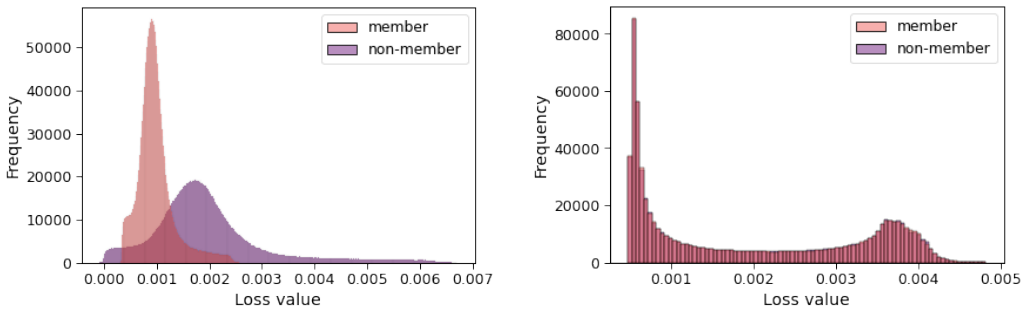
$$W_{i,t+1} = W_i - \eta \frac{\nabla_W \mathcal{L}(x, W_{i,t})}{\sqrt{v_{t+1}} + 1e^{-5}} \tag{8.11}$$

To tackle the aforementioned convergence issues, *PASTEL* combines properties preventing $\mathcal{L}$ from being stuck at local minima and saddle points of the loss function over different dimensions, mitigating the risks of overfitting, and consequently improving the convergence of $\mathcal{L}$ and the accuracy of W_i. We provide a model utility evaluation for different adaptive gradient descent optimizers in §9.3 to illustrate these statements and motivate our proposal.

8.4.3 Analytical Insights

In this section, we analyze the key properties that explain the effectiveness of *PASTEL*. We provide both analytical and empirical evidence explaining the impact of *PASTEL* on the generalization gap.

Generalization Gap Reduction with *PASTEL*. *PASTEL* is a regularization method that aims to reduce the internal generalization gap between member and non-member data. This is achieved by adding a regularization term to the loss function that encourages the model to produce similar outputs for member and non-member inputs. By doing so, *PASTEL* helps to prevent the model from overfitting to the member data and leaking sensitive information. In our study, we apply *PASTEL* to Purchase100 and plot the resulting loss histograms on member and non-member data in Figure 8.4. As shown in the figure, *PASTEL* blurs the shift between member and non-member loss distributions, indicating a reduction in the internal generalization gap, which naturally leads to a internal generalization gap and reduced privacy leakage [114, 25]. To further enhance the effectiveness of *PASTEL*, we used adaptive gradient descent to train the model and minimize the internal generalization gap. This combination of techniques resulted in a more robust and privacy-preserving model that can better handle real-world scenarios. Overall, our results demonstrate the effectiveness of *PASTEL* in reducing the internal generalization gap and improving the privacy-preserving capabilities of machine learning models.



(a) Without *PASTEL* (b) with *PASTEL*

FIGURE 8.4: Loss histogram with Purchase100 with fully connected neural network

***PASTEL* Convergence Analysis.** In this section, we analyze the convergence of *PASTEL* using our loss function defined in §8.4.2. The *PASTEL* loss function is defined as $L = L_{priv} + L_{acc}$.

To demonstrate that the loss function with *Jensen-Shannon Divergence* converges, we need to show that it satisfies the conditions of a convex optimization problem [61]. Let $G_W(X)$, model gradients of a batch of member data. and $G_w(X')$, model gradients of a batch of a non-member data, where W are the learnable parameters of the model. The JSD between $G_w(X)$ and $G_w(X')$ is defined in Eq. (8.6) The loss function with JSD can be written as:

$$L_{priv} = JSD(G_w(X), G_w(X')) \tag{8.12}$$

Our goal is to minimize this loss function with respect to the parameters W, such that model gradients of member data $G_w(X)$ converges to the model gradients of non-member data $G_w(X')$. To show that the loss function with *Jensen-Shannon Divergence* converges with references, we need to prove the following conditions: The loss function L_{priv} is convex with respect to W. The gradient of L_{priv} with respect to W exists and is continuous. The global minimum of L_{priv} corresponds to the model gradients of member data $G_w(X)$. Let's start with the first condition. We can write the JSD as a convex combination of KL divergences:

$$JSD(G_w(X), G_w(X')) = \frac{KL(G_w(X), m) + KL(G_w(X'), m)}{2} \tag{8.13}$$

where

$$m = \frac{G_w(X) + G_w(X')}{2} \tag{8.14}$$

Since KL divergence is a convex function, JSD is also a convex function. Therefore, the loss function L_{priv} is convex with respect to W. For the second condition, we need to compute the gradient of L_{priv} with respect to W. Using the chain rule, we have:

$$\nabla L(W) = \nabla G_w(X') JSD(G_w(X), G_w(X')) \tag{8.15}$$

where $\nabla G_w(X')$ is the gradient of $G_w(X')$ with respect to W. To compute the gradient of JSD with respect to $G_w(X')$, we can use the following formula:

$$\nabla G_w(X') JSD(G_w(X), G_w(X')) = \frac{\nabla G_w(X') KL(G_w(X), m) + \nabla G_w(X') KL(G_w(X'), m)}{2} \tag{8.16}$$

Since KL divergence is differentiable, we can compute its gradient as:

$$\nabla G_w(X') KL(G_w(X), G_w(X')) = -\frac{G_w(X)}{G_w(X')} + 1 \tag{8.17}$$

Using the above formula, we can compute the gradient of JSD with respect to $G_w(X')$ and then the gradient of L_{priv} with respect to W. It can be shown that the gradient of L_{priv} with respect to W exists and is continuous. Finally, for the third condition, we need to show that the global minimum of L_{priv} corresponds to the reference distribution $G_w(X)$. Since JSD is symmetric and bounded, the minimum value of $JSD(G_w(X), G_w(X'))$ is 0, which occurs when p = $G_w(X')$. Therefore, the global minimum of L_{priv} corresponds to the reference $G_w(X)$.
In summary, we have shown that the loss function with *Jensen-Shannon Divergence* converges with references under the conditions of convexity, continuity, and global minimum.

8.4.4 Summary

In this chapter, we have introduced *PASTEL*, a novel approach to thwart membership inference attacks. Unlike existing methods, *PASTEL* rely on narrowing the generalization gap to bolster defenses against such attacks. Furthermore, it uses adaptive gradient descent to enhance model accuracy. In the next chapter, we'll conduct a practical evaluation of *PASTEL* and gauge its performance against existing methods in the field.

Chapter 9

Experimental Evaluation of *PASTEL*

9.1 Implementation and Experimental Setup

9.1.1 Software and Hardware Environment.

All our classifiers and methods are implemented using PyTorch 1.13. We use an existing implementation of membership inference attack [93], and the Opacus library [116] for running differential privacy-based protection methods, as detailed in §9.1.4. Our experiments are performed using NVIDIA A40 GPUs. The software prototype of *PASTEL* is publicly available at[1]:
https://anonymous.4open.science/r/pastel–CF24/

9.1.2 Datasets

To evaluate the performance of *PASTEL* across multiple applications, we consider various datasets, including four image datasets (Cifar-10, Cifar-100, GTSRB, and CelebA), three tabular datasets (MotionSense, Purchase100 and Texas100), and one raw audio waveform dataset (Speech Commands).

TABLE 9.1: Used datasets and models

Dataset	#Records	#Features	Model
MotionSense	345,890	10	CNN with 3 convolutional layers and 4 FC layers
CelebA	202,599	64x64	VGG11 (CNN) with batchnorm layers [95]
GTSRB	51,389	3x48x48	VGG11 (CNN) with batchnorm layers [95]
Cifar-10	50,000	3x32x32	ResNet20 (CNN) [44]
Cifar-100	50,000	3x32x32	ResNet20 (CNN) [44]
Speech Commands	64,727	16,000	M18: 17 layers CNN with a batchnorm layer after each convolutional layer, and a fully-connected classification layer
Purchase100	97,324	600	6 hidden layers FCNN with respective sizes 4096, 2048, 1024, 512, 256 and 128
Texas100	67,330	6,170	6 hidden layers FCNN with respective sizes 4096, 2048, 1024, 512, 256 and 128

[1]The anonymous link will be replaced by the public git link.

9.1.3 FL Experimental Setup

We run our experiments in a FL configuration with 5 clients for images and audio datasets, and 10 clients for Purchase100 and Texas100. For all datasets, we split the data into 4 disjoint splits of equal size (in accordance with the Membership Inference attack's optimal efficiency conditions [93]) respectively corresponding to the target model's training and testing set, and the shadow model's training and testing set. We train a single shadow model used for running the attack against all models, while the target model's training set is split into disjoint slices of equal sizes, according to the number of clients, in an IID configuration. As classically done in privacy-preserving FL experiments, we use a test set that is the union of the test sets of all clients, and apply that union test set to all clients in a first time. We also compare to a more realistic scenario where each client makes use of its own test set. We conducted such a scenario with 3 different datasets and with the different baselines. This shows a negligible difference with the former case, below 1% for both attack AUC and model accuracy. Furthermore, we use the Adam [57] optimizer by default in the No-defense scenario and the Cross Entropy loss function in all scenarios. The model learning rate is 10^{-3} when using Resnet20, VGG and M18 (for images and audio datasets), and 10^{-4} for FCNNs with tabular datasets. For experiments using FCNNs on Purchase100 and Texas100, we run the experiments on 300 FL rounds with 10 local epochs per client, and we throw the membership inference attack starting from the 290th round. With VGG and ResNet on CelebA, GTSRB, Cifar-10 and Cifar-100, we run 50 FL rounds with 5 local epochs per clients, and we throw the attack from the 40th round. For the M18 model trained on Speech Commands, we run 80 FL rounds and we throw the attack from the 70th round.

9.1.4 Baselines

We compare *PASTEL* with a no-defense baseline scenario, and with five state-of-the-art defense solutions, among which one is based on gradient compression (GC) method [34], a cryptographic-based solution with secure aggregation (SA) [121], and three other methods based on differential privacy, namely local differential privacy (LDP) [20], central differential privacy (CDP) [82], weak differential privacy (WDP) [82]. With LDP, the noise is applied locally by participants before aggregation. Each participant runs a random perturbation algorithm and sends its parameters to the server. With CDP, the server clips the L2 norm of clients updates, then aggregates the updates and adds Gaussian Noise, before sending the noisy model parameters to the clients. We use Opacus [116], an efficient and ready-to-use differential privacy framework. With CDP and LDP, ϵ parameter is set to 2.2, and the probability of privacy leakage δ is set to 10^{-5}. With WDP [99], the server applies norm bounding with a norm set to 5, and adds gaussian noise with a standard deviation $\sigma = 0.025$.

9.1.5 Evaluation Metrics

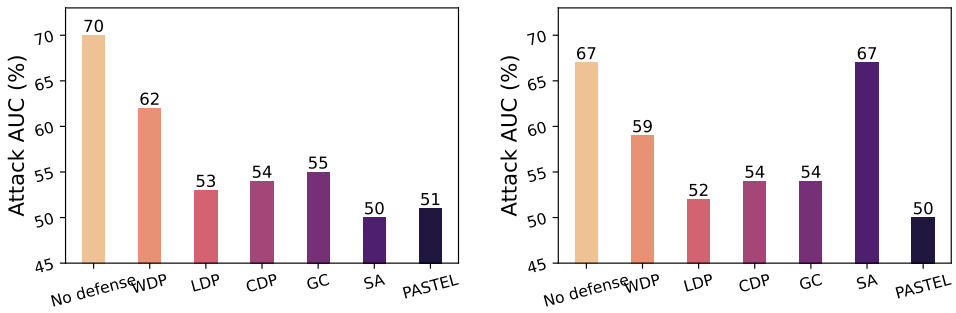
PASTEL's goal as a privacy protection mechanism for FL systems is to maximize the data privacy against potential malicious participants within the FL system (either on

server or client side) and to ensure that the privacy protection has the least negative impact on the utility of the protected models. We focus on the evaluation of *PASTEL* by measuring the trade-off between the AUC of a Membership Inference Attack against models (on both client and server side) [93], and the utility of the models. For each dataset and model case, we compare *PASTEL* with the baselines described in §9.1.4 according to the metrics described in 7.1.4. The evaluation of computational costs is provided in §9.3.2.

9.2 Evaluation of Privacy-Preserving Federated Learning

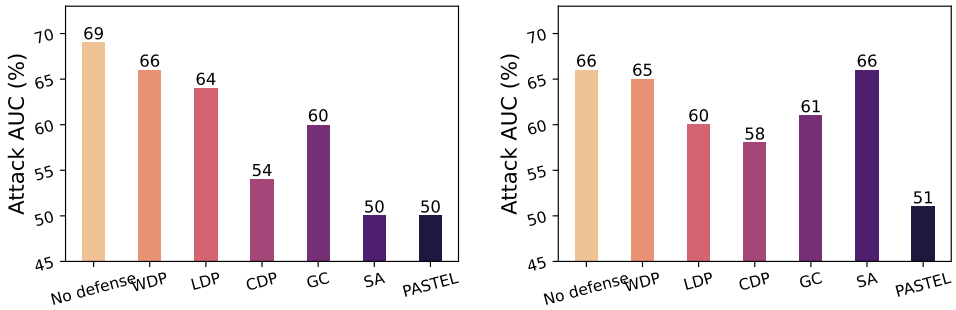
We compare *PASTEL* with different other methods, including a baseline with non-defended models, as well as the aforementioned defense techniques in §9.1.4 against white box membership inference attack described in [93]. We use seven datasets and target models which are widely used in prior works on MIA and defenses. We consider MotionSense with CNN described in Table9.1, Purchase100 with a 6 hidden-layers fully connected neural network. For CelebA, GTSRB, Cifar-10 and Cifar-100 we use ResNet18 and VGG9, finally, for Speech Commands we consider M18, a model specifically designed for this dataset. For each dataset, half of the data is used as prior knowledge of the attacker to run MIAs, and the other half is used for training and testing the FL system. For each dataset, we systematically evaluate *PASTEL*, *WDP*, *LDP*, *CDP*, *GC*, and *SA*, by considering both the utility and membership inference attack AUC on clients' local models and global FL model.

Figure 9.2 depicts the results of the privacy leakage analysis for all datasets presented in Table 9.1. The privacy leakage is evaluated by means of the area under the curve (AUC) metric, which is used to assess the efficacy of membership inference attack. Our findings indicate that for both global and local model inference attacks, *PASTEL* offers mitigation rates that are in close proximity to the 50% mark across all datasets. A mitigation rate of 50% suggests that the attack is no better than random guessing, indicating that the privacy protection provided by *PASTEL* is strong. Furthermore, our results indicate that *PASTEL* performs similarly to differential privacy techniques, which are known to be effective in preserving privacy. In some cases, *PASTEL* even outperforms differential privacy methods in terms of privacy preservation. Secure aggregation serves as a robust safeguard for individual local models by employing encryption, thereby mitigating potential inference attacks, where the AUC becomes akin to that of a random guessing scenario, resulting in a statistically neutral 50% AUC. However, this protective efficacy does not uniformly extend to ensuring privacy for the overarching global model, given its accessibility.In contrast, gradient compression proves more adept at ensuring privacy for the global model as opposed to local models. This is attributed to the intrinsic function of gradient compression in curtailing data transmission to the server, thereby reducing the viability of inference attacks These findings demonstrate the potential of *PASTEL* as a privacy-preserving technique, particularly in scenarios where differential privacy techniques may not be applicable due to their limitations.



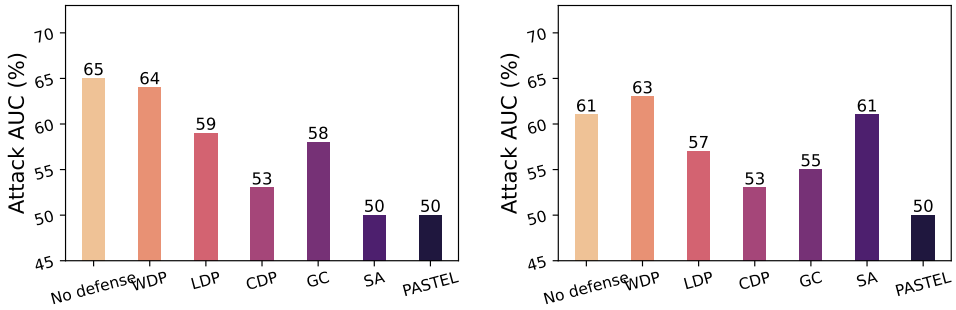
(a) MotionSense– local models

(b) MotionSense– global model



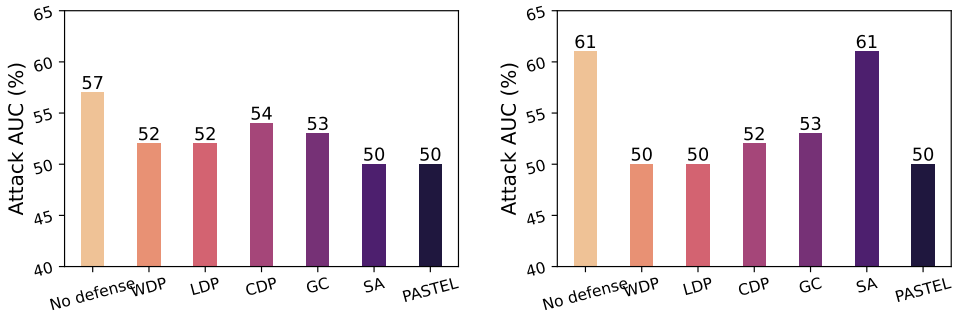
(c) Cifar-10– local models

(d) Cifar-10– global model



(e) Cifar-100– local models

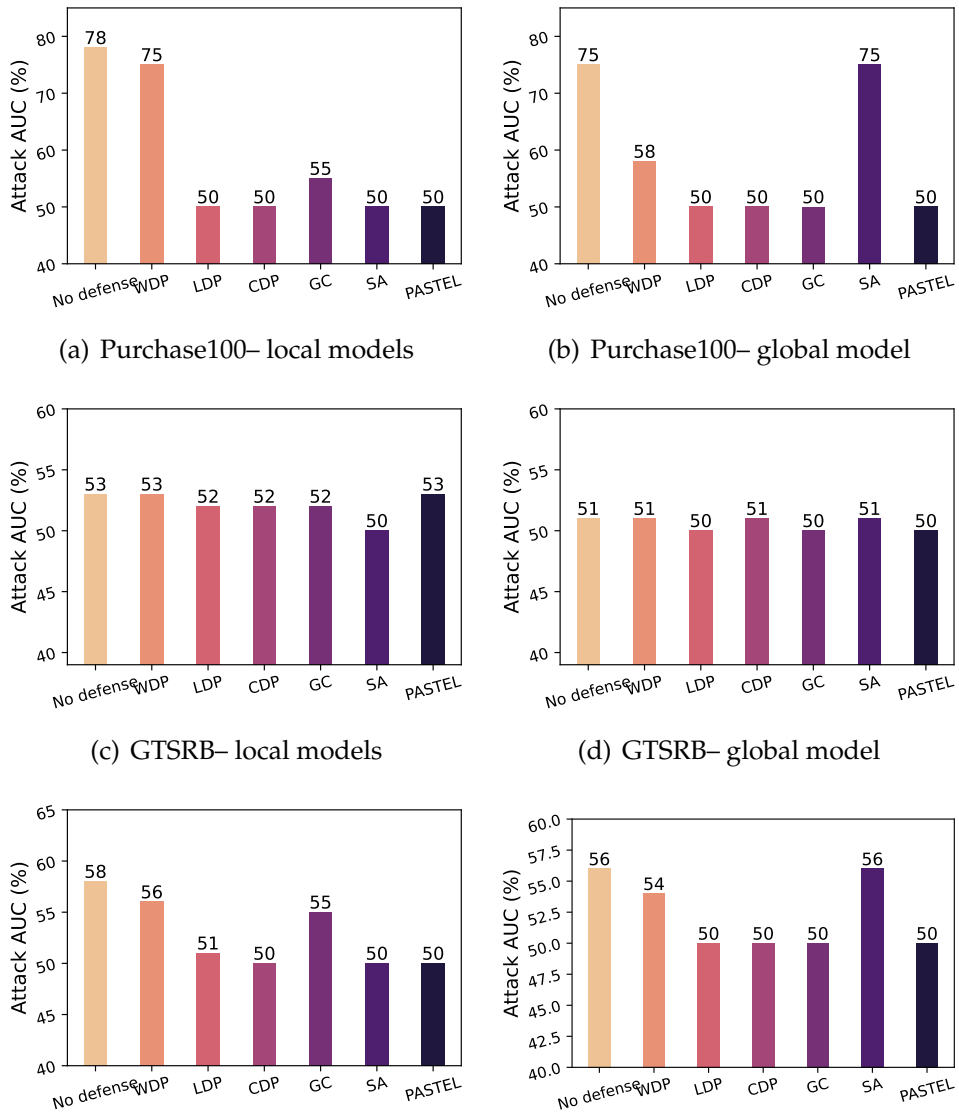
(f) Cifar-100– global model



(g) CelebA– local models

(h) CelebA– global model

FIGURE 9.1: Privacy leakage with *PASTEL* and state-of-the-art protection mechanisms - Image Dataset



(a) Purchase100– local models

(b) Purchase100– global model

(c) GTSRB– local models

(d) GTSRB– global model

(e) Speech Commands– local models

(f) Speech Commands– global model

FIGURE 9.2: Privacy leakage with *PASTEL* and state-of-the-art protection mechanisms - Image Dataset

9.3 Tradeoff Between Privacy and Utility

Figure 9.4 depicts the tradeoff between privacy and utility for *PASTEL* and other state-of-the-art mechanisms across a range of datasets. We evaluated the performance of these mechanisms on both local models (client-side) and global models (server-side). The x-axis of the figure represents the model's accuracy, while the y-axis shows the AUC of the membership inference attack. The ideal mechanism would be located at the bottom-right corner of the graph, indicating high accuracy and low AUC. Our analysis reveals that state-of-the-art mechanisms such as *WDP* and *LDP* offer reasonable mitigation rates against the attack, but they also exhibit a negative impact on the model's utility. For instance, on the Cifar-10 dataset, *WDP* only reduces the attack AUC by 2% on the global model and 3% on the local models.

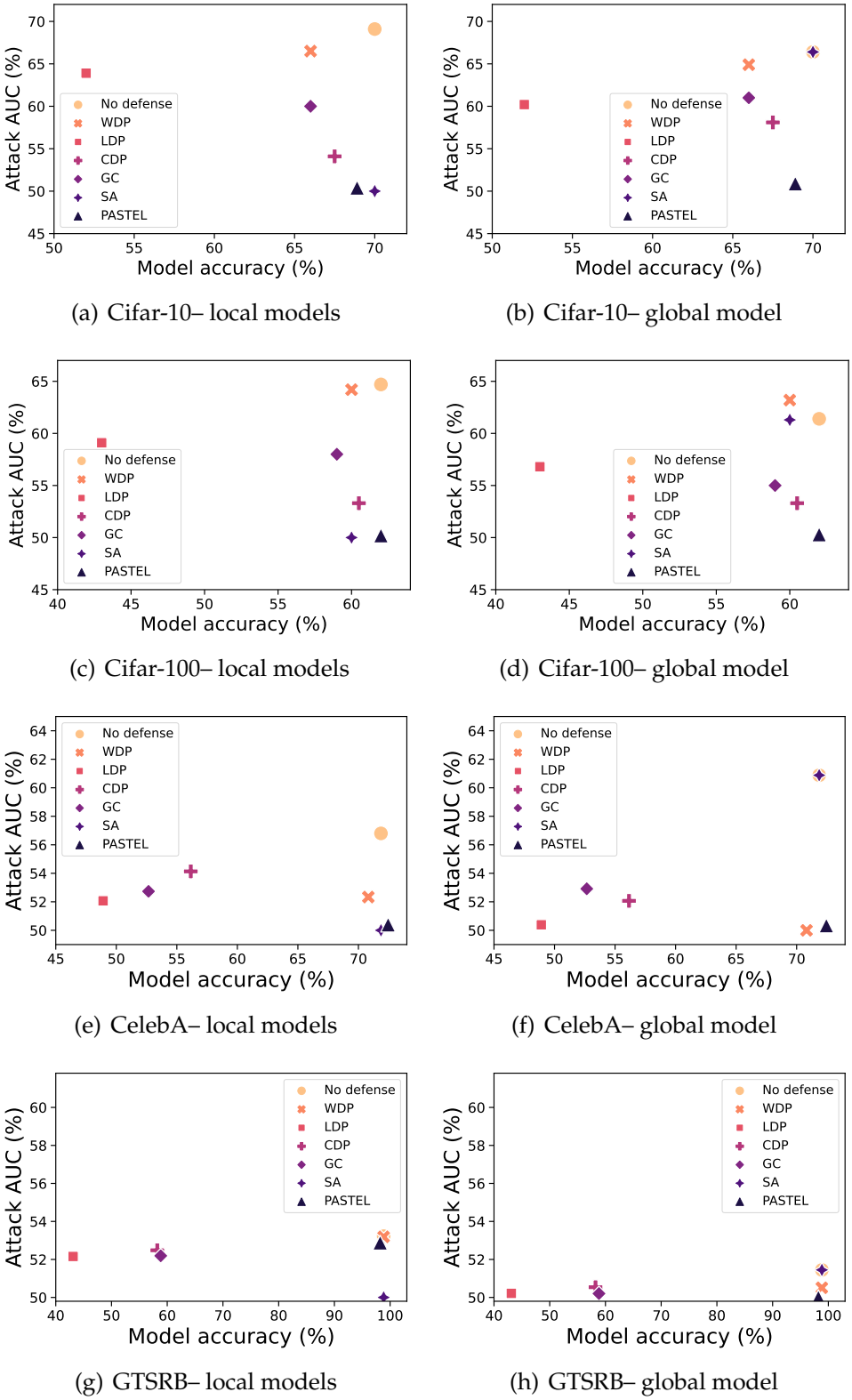
Similarly, *CDP* reduces the attack AUC by only 6%, while the model's utility drops by almost 20%.

Our experiments also indicate that *LDP* can effectively reduce the AUC of the membership inference attack. However, it does so at a significant cost to the model's utility. For example, on the Speech Commands dataset, the attack AUC is reduced to 50%, but the model's utility drops by almost 25%. Secure aggregation employs encryption to defend local models, rendering inference attacks as effective as random chance (50% accuracy). However, this level of protection does not translate equally to the global model, which retains certain vulnerabilities. In contrast, gradient compression is more effective at preserving privacy for the global model compared to local models. This is due to gradient compression's ability to reduce data transmission to the server, thus diminishing the viability of inference attacks and enhancing overall privacy. In the case of gradient compression, while it effectively enhances privacy for the global model, it also results in reduced accuracy. For instance, on the Speech Commands dataset, accuracy dropped from 86% for the baseline to 60% with gradient compression. In contrast, *PASTEL* offers the same level of mitigation as the undefended model while preserving better model utility. For example, on the same Speech Commands dataset, *PASTEL* guarantees the same level of mitigation against the attack as the undefended model while simultaneously maintaining better model utility. *PASTEL* allows to mitigate the attack on the different datasets and presents the most competitive results. The AUC attack does not go far from 50% on the local model or the global models while maintaining a fairly high model accuracy rate equal to the baseline one. In all the experiments, the model utility with *PASTEL*, is not lower than the one with baseline, in certain cases *PASTEL* also improves the model utility, for instance with Speech Commands in Figure 9.4(f), where the accuracy of the baseline is 86% vs. 90% with *PASTEL*. This suggests that *PASTEL* can effectively mitigate privacy leakage while preserving model accuracy, making it a promising alternative to traditional privacy-preserving methods like differential privacy.

In 9.5, we present an empirical analysis of the impact of the loss function introduced by the *PASTEL* mechanism on the loss values incurred by the model on the member and non-member data. Specifically, we evaluate the effectiveness of *PASTEL* in minimizing the difference in distribution between member and non-member data while avoiding the generation of larger loss values. Our results confirm the findings of the previous sections, where we showed that state-of-the-art mechanisms that focus solely on minimizing the loss between member and non-member data often result in larger loss values.

In contrast, the multi-objective function utilized in *PASTEL* allows for the minimization of both the classification loss and the difference between member and non-member data. Our analysis shows that this approach successfully minimizes the difference in distribution between member and non-member data, while incurring minimal loss values, thereby striking a balance between privacy and utility.

For instance, if we compare the results of PASTEL and LDP in Figure 7, we observe that loss values are generally higher for LDP than for PASTEL. This difference in loss values between the two systems can be attributed to the underlying optimization strategy and the trade-off they make between privacy and utility. LDP



(a) Cifar-10– local models (b) Cifar-10– global model

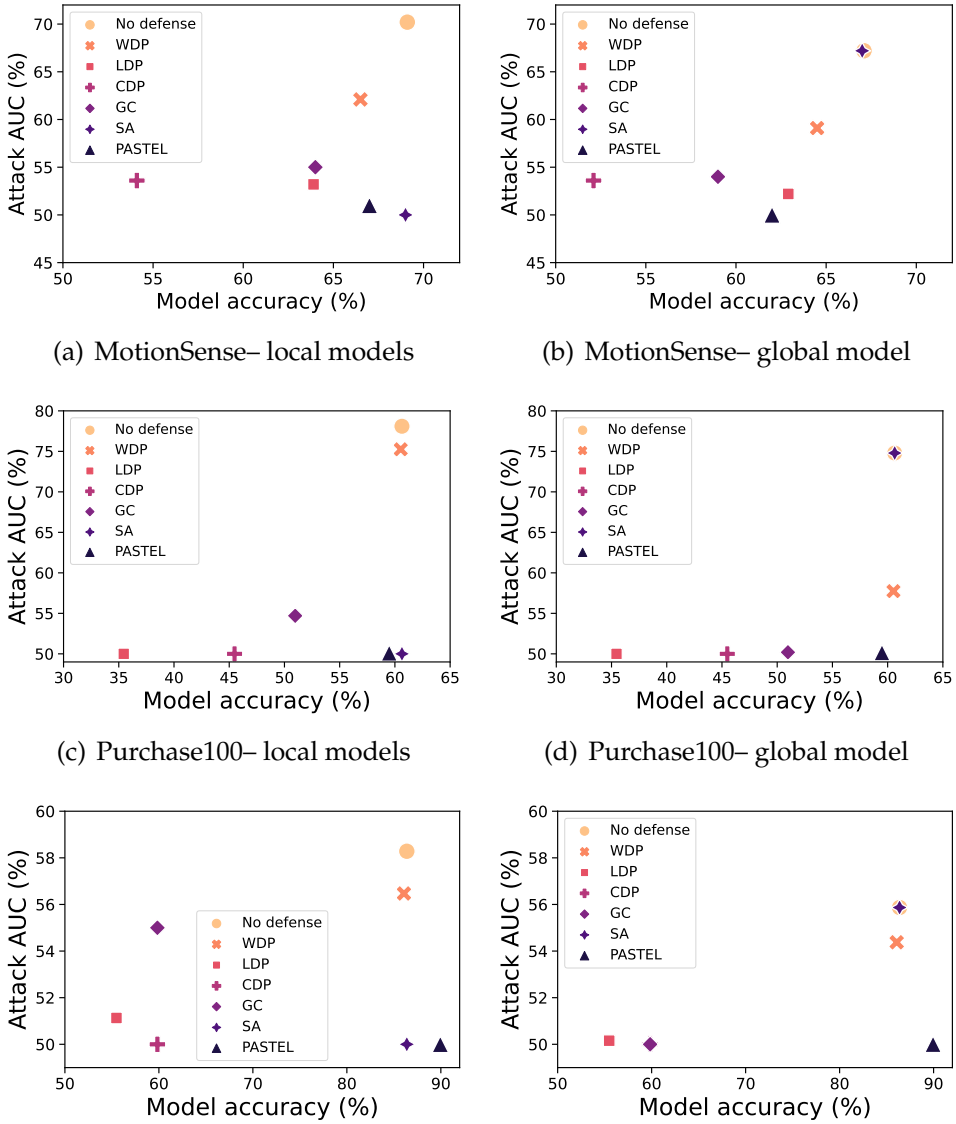
(c) Cifar-100– local models (d) Cifar-100– global model

(e) CelebA– local models (f) CelebA– global model

(g) GTSRB– local models (h) GTSRB– global model

FIGURE 9.3: Privacy-Utility tradeoff with *PASTEL* and state-of-the-art protection mechanisms – Image Dataset

primarily focuses on reducing the risk of memorization by adding noise to the gradients, which can lead to increased loss values due to the added perturbations (6



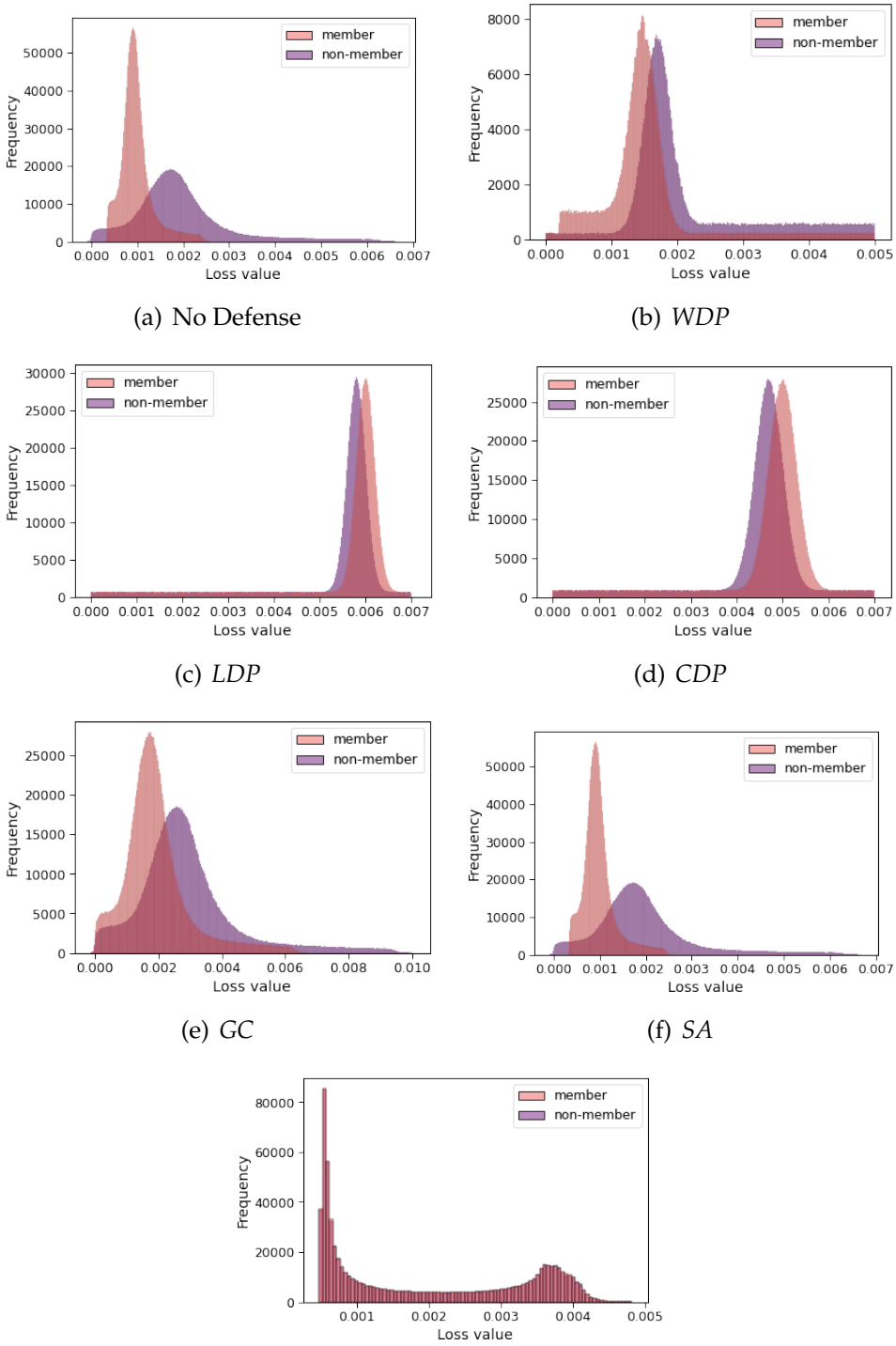
(a) MotionSense– local models (b) MotionSense– global model

(c) Purchase100– local models (d) Purchase100– global model

(e) Speech Commands– local models (f) Speech Commands– global model

FIGURE 9.4: Privacy-Utility tradeoff with *PASTEL* and state-of-the-art protection mechanisms

times bigger than no defended gradient). On the other hand, PASTEL incorporates a multi-objective function that strikes a balance between minimizing the distributional gap between member and non-member data and managing loss values. By considering both objectives, PASTEL manages to achieve competitive loss values while effectively addressing the distributional gap. Furthermore, the fact that the loss curves of members and non-members overlap for PASTEL, whereas they do not fully overlap for WDP, can be explained by the way PASTEL balances its objectives. PASTEL's multi-objective function is designed to ensure that the model learns meaningful representations for both member and non-member data while minimizing the distributional disparity. This balance allows the model to generalize better across different data types, resulting in similar loss values for both member and non-member data. In contrast, the disparity between member and non-member



(a) No Defense (b) *WDP*

(c) *LDP* (d) *CDP*

(e) *GC* (f) *SA*

(g) *PASTEL*

FIGURE 9.5: Loss histogram with Purchase100 with fully connected neural network

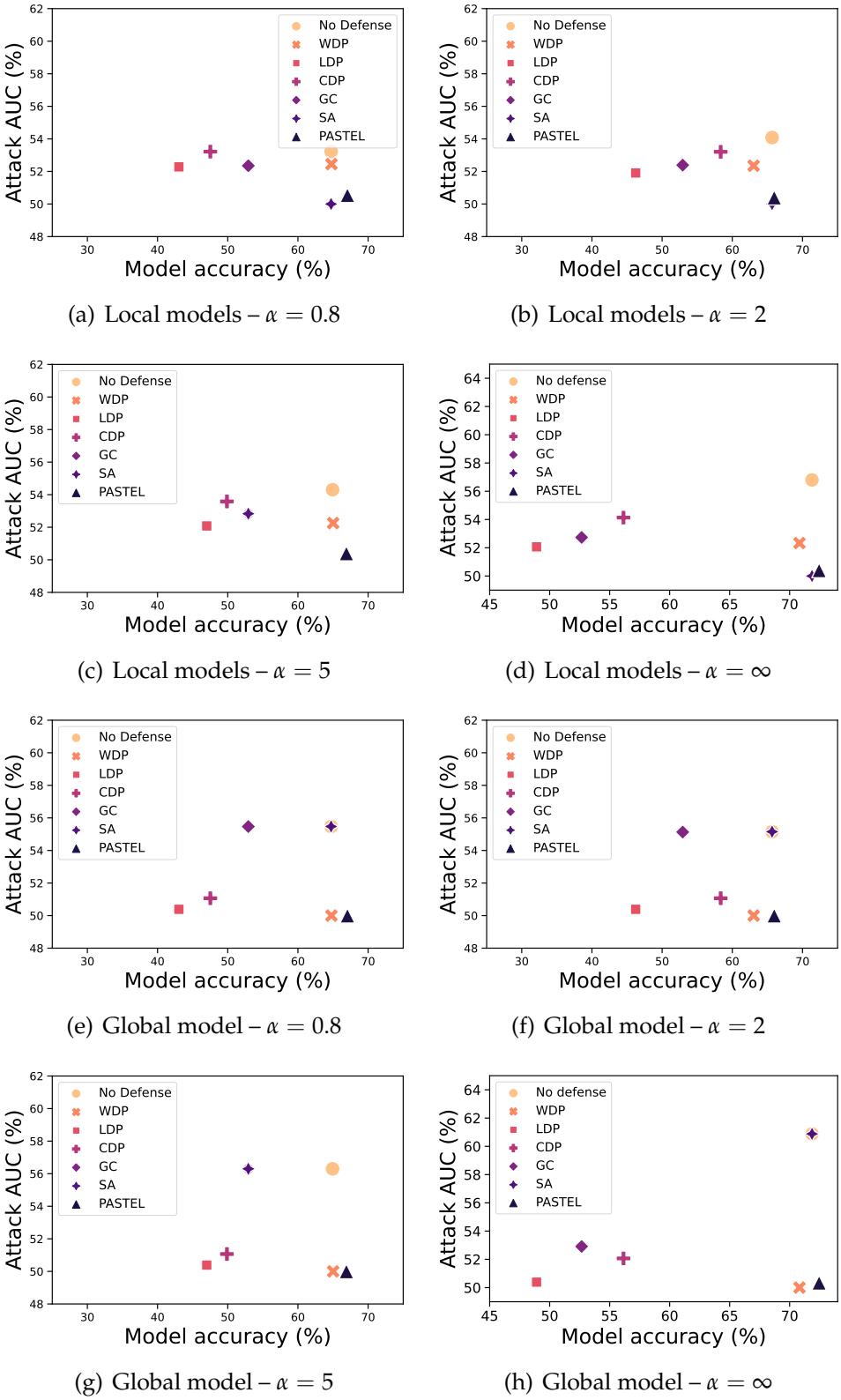
loss values for methods like WDP is proportional to the noise added during the differential privacy process. For techniques employing weak differential privacy, where the noise added is relatively small, the resulting loss values remain modest, and a discernible difference between the two curves can be observed. However,

for strong differential privacy, where a higher level of noise is introduced, the loss curves tend to overlap significantly, with one curve nearly overlaying the other. This phenomenon occurs because the added noise disrupts the optimization process substantially, leading to notably higher loss values. In summary, our findings highlight the nuanced impact of differential privacy techniques on loss values and curve overlap. *PASTEL* strategy of balancing objectives provides a favorable outcome, achieving both minimized distributional disparities and controlled loss values, while the noise introduced by strong differential privacy techniques can lead to significant overlap between loss curves and higher overall loss values. Thus, our results corroborate the findings presented in previous sections, where we demonstrated that state-of-the-art methods focusing solely on loss minimization between member and non-member data may lead to increased loss values. *PASTEL*, however, stands out by skillfully navigating this trade-off. It demonstrates remarkable success in minimizing the distributional gap between member and non-member data while effectively managing and mitigating loss values.

9.3.1 Evaluation of Privacy Protection in Non-IID Settings

Our previous experiments consider an IID data distribution (*i.e.*, the same quantity of data and classes distribution for all clients). In practice, FL systems use to involve setups with clients respectively owning more or less disparate quantities of data and non-identical class distributions for each [120]. With the objective of evaluating *PASTEL* against other state-of-the-art defense mechanisms in realistic FL conditions, we then propose to reproduce the evaluation scenario performed in §9.2, while considering a non-IID data partitioning across FL clients. We apply the Dirichlet distribution function to generate data distribution across clients [58]. Dirichlet function 's α parameter defines the data distribution ratio between clients for each class. A smaller α value induces a more non-IID distribution. We then implement a FL system using various non-IID distributions, by considering three five of α for the Dirichlet function, respectively 0.8, 2, 5 and ∞ (*i.e.*, a high value, equivalent to an IID setting), in order to evaluate *PASTEL* against state-of-the-art mechanisms in more or less non-IID distributions. We run experiments with different non-IID distributions for the CelebA dataset with VGG11 model, in the same experimental conditions than in §9.3.

Figure 9.6 depicts our experimental results for each α value considered, comparing *PASTEL* with all state-of-the-art defense mechanisms, and shows their respective behaviour. Figure 9.6(a) to Figure 9.6(h) show the tradeoff between model utility and privacy for each α value, for both global and local models. We first notice that *PASTEL* remains the most performing defense mechanism compared to considered state-of-the-art mechanisms, for both client and server models, as the Attack AUC remains close to 50%, with a low variation depending on the α value, including the lower values that involve a stronger non-IID distribution. In each case, the tendencies between baselines and *PASTEL* are similar to what we observed in §9.3, meaning that introducing non-IID distributions does not significantly impact the behaviour of *PASTEL* versus state-of-the-art defense mechanisms. We may still notice that a lower α value induces a lower model accuracy; this is an expected behaviour



(a) Local models – $\alpha = 0.8$

(b) Local models – $\alpha = 2$

(c) Local models – $\alpha = 5$

(d) Local models – $\alpha = \infty$

(e) Global model – $\alpha = 0.8$

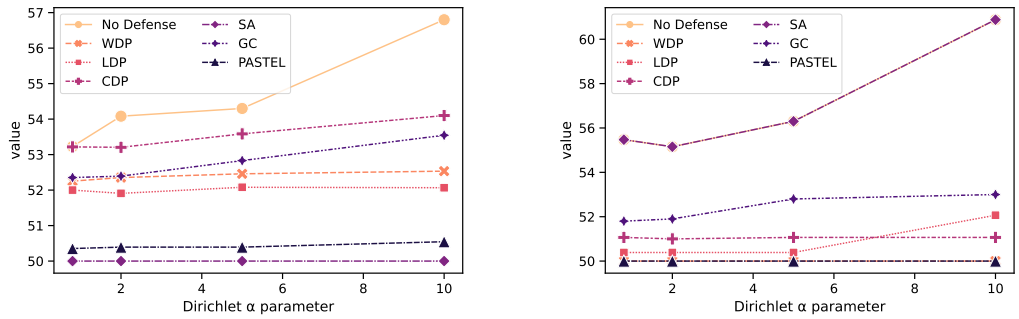
(f) Global model – $\alpha = 2$

(g) Global model – $\alpha = 5$

(h) Global model – $\alpha = \infty$

FIGURE 9.6: Privacy leakage and model utility under different non-IID settings

as non-IID distribution use to impact the effectiveness of Federated Learning systems at optimizing the local models' accuracy.



(a) Scalability of protection on local models (b) Scalability of protection on global model

FIGURE 9.7: privacy scalability with regard to non-IID settings

Figure 9.7(a) and Figure 9.7(b) present the scalability of the privacy (*i.e.*, resilience to MIA attacks) of *PASTEL* and state-of-the-art solutions depending on how far is a FL setting non-IID. Overall, and espacillay without defense, higher is the α Dirichlet parameter, higher is the attack success. This behavior remains true with CDP defense as well. Interestingly, with *PASTEL* and other defense mechanisms such as LDP and WDP, the defense success is poorly impacted by the degree of non-IID setting. Thus, *PASTEL* is able to maintain a good privacy protection against MIA attacks (cf. Figures 9.6(a) - 9.6(h)) in non-IID FL settings.

9.3.2 Cost of Privacy-Preserving Mechanisms

We further analyze the computational costs of *PASTEL*, to evaluate to what extends our solution shows reasonable performances and tackles the high computation costs issues of protection mechanisms based on differential privacy. We provide an evaluation of *PASTEL*'s performance with respect to the cost metrics defined in §9.1.5, *i.e.*, the average training duration of local models, the average server aggregation duration, and the peak value of GPU memory usage. For each dataset and model considered in our evauation, we compare *PASTEL* with all the baselines defined in §9.1.4. All measures are performed with the experimental setup described in §9.1.

Client Local Training Time. We evaluate the average training time per round for each client in various scenarios. This duration represents the total time taken for all the local training epochs of a single client during a FL round. Our results are summarized in Table 9.2. We observe that differential privacy mechanisms such as LDP, CDP, and WDP have a negative impact on the training time. Although the Opacus framework [116] significantly improves the speed of differential privacy, it still incurs a cost of up to 4.5 times in the worst-case scenario, such as when using the FCNN classifier. By the same way, the *SA* mechanism may highly impact the local training time, as shown by [121]. The gradient compression process of *GC* implies a variable computational overhead, increasing the training time of 23% in the most favorable case, but it may reach an overhead up to 55%. On the other hand, *PASTEL* effectively addresses the computational extra-cost issue of differential privacy in terms of training time without notably affecting the system's average performance.

TABLE 9.2: Cost of FL privacy-preserving mechanisms in terms of training time and FL aggregation time and memory usage

System	Client-side training time(s)	Server-side FL aggregation (ms)	Client-side memory usage (MB)
MotionSense			
No Defense	6.1	53	1200
WDP	+51%	0%	+9%
LDP	+44%	0%	+9%
CDP	+19%	+4,328%	+7%
SA *	+163% to 245%	+4,719% to 5,660%	N / A
GC	+34%	0%	+0%
PASTEL	+12%	0%	0%
Cifar-10			
No Defense	9.4	14	2300
WDP	+54%	0%	+78%
LDP	+42%	0%	+72%
CDP	+51%	+8,195%	+69%
SA *	159%	21,428%	N / A
GC	+38%	0%	+0%
PASTEL	+21%	0%	0%
Cifar-100			
No Defense	11.3	68	2300
WDP	+41%	0%	+90%
LDP	+60%	0%	+91%
CDP	+39%	+19,789%	+90%
SA *	+88% to 132%	+3,671% to 4,412%	N / A
GC	+55%	0%	+0%
PASTEL	+21%	0%	0%
CelebA			
No Defense	13.4	69	2900
WDP	+51%	0%	+163%
LDP	+44%	0%	+173%
CDP	+50%	+6,559%	+163%
SA *	+74%	+3,623%	N / A
GC	+28%	0%	+0%
PASTEL	+21%	0%	0%
GTSRB			
No Defense	7.4	67	6100
WDP	+51%	0%	+94%
LDP	+44%	0%	+94%
CDP	+50%	+3,000%	+94%
SA *	+135% to 202%	3,731% to 4,477%	N / A
GC	+23%	0%	+0%
PASTEL	+21%	0%	0%
Purchase100			
No Defense	6.3	68	2100
WDP	+281%	0%	+336%
LDP	+283%	0%	+391%
CDP	+213%	+1,199%	+336%
SA *	+135% to 238%	3,676% to 4,411%	N / A
GC	+47%	0%	+0%
PASTEL	+21%	0%	0%
Speech Commands			
No Defense	19.4	146	18100
WDP	+27%	0%	+78%
LDP	+25%	0%	+78%
CDP	+26%	+6,942%	+78%
SA *	+51% to 77%	1,712% to 2,054%	N / A
GC	+29%	0%	+0%
PASTEL	+21%	0%	0%

* The computational costs of the Secure Aggregation mechanism are estimated on the basis of the experimental results provided in [121].

Server-Side Aggregation Time. In each scenario, we measure the average time for server aggregation per round. Specifically, we record the time elapsed between the moment the server received all the weights to be aggregated and the moment it sends the aggregated weights. We aim to compare defense mechanisms actually impacting the aggregation server, such as CDP and SA, against *PASTEL*. As seen in 9.2, the use of CDP increases the aggregation time by a factor of up to 20 in the case of CelebA with VGG. This additional cost can be attributed to the design principle of CDP, which involves adding noise to the parameter aggregate before transmitting it to clients. *SA* causes extra computational costs on server side by definition [121], with an overhead up to 21,48% in the most extreme cases. These processes significantly increases the aggregation time, taking seconds in our case. However, with *PASTEL*, *LDP*, and *WDP*, the times were of the same order of magnitude as the scenario without any baseline.

Memory Usage. In each scenario, we computed the average value of GPU memory usage during the local model training step. Our findings indicate that the use of LDP, CDP, and WDP systematically increased the memory usage. The rise in memory usage was relatively small in scenarios using ResNet (an increase of approximately 8% for Cifar-10 with ResNet); however, it reached factors of up to 4 in the case of Purchase100 for these baselines, as shown in 9.2. As discussed in [116], the Opacus DP framework is optimized for batched per-sample gradient computation and speeds up the model training with differential privacy, but it inevitably impacts the memory usage. Furthermore, the memory increase depends on both the data feature size and the number of training parameters. For Fully Connected models, the memory usage can reach factors of up to 334, as observed in the FCNN classifier architecture with six layers of sizes 4096, 2048, 1024, 512, 256, and 128 and 600 features in the Purchase dataset. In contrast, *PASTEL* is a privacy-preserving optimization algorithm that does not have any impact on GPU memory usage during local model training. The training process *PASTEL* uses adaptive gradient descent, which is a variant of stochastic gradient descent (SGD) that adjusts the learning rate on a per-parameter basis. Unlike other differential privacy mechanisms, such as LDP, CDP, and WDP, *PASTEL* does not require any additional computations to be performed during local model training. Instead, it relies on a straightforward privacy-preserving update to the gradient estimates that reduces the amount of information that is leaked by each client's update. Because *PASTEL* does not require any additional computations during local model training, it has no impact on GPU memory usage. This makes it an attractive privacy-preserving optimization algorithm for federated learning scenarios in which GPU memory is a scarce resource.

9.3.3 Discussion

Different future directions of *PASTEL* could be studied, including further exploring the threat model, the underlying datasets and models, or the impact of data bias on privacy protection.

Indeed, it could be interesting to consider a similar multi-objective approach for taking into account both model privacy and accuracy, in order to explore other threat models and counter other types of privacy attacks, such as property inference, or model inversion. Property inference involves attackers deducing sensitive attributes

of individuals by querying a trained machine learning model, thereby compromising privacy [106]. Model inversion [123], on the other hand, pertains to adversaries reconstructing original training data by analyzing a model's outputs or gradients, potentially unveiling confidential information. While *PASTEL*'s primary focus lies in countering membership inference attacks by reducing the internal generalization gap, its specialized design may limit its applicability to other types of attacks. Unlike more versatile solutions such as Differential Privacy (DP), which offer a broader scope of defense against various privacy threats, *PASTEL*'s effectiveness against attack types beyond membership inference remains uncertain. It's worth noting that DP, due to its inherent noise injection mechanisms, has demonstrated robustness against a wider array of attack strategies, making it a more comprehensive option for defending against diverse privacy breaches. To thoroughly understand *PASTEL*'s strengths and limitations, future research should undertake rigorous comparative analyses, evaluating its performance against different attack vectors, including but not limited to attribute inference, model inversion, and adversarial attacks. Such investigations will shed light on the extent to which *PASTEL*'s capabilities generalize across various privacy threatening scenarios and provide insights into its suitability as a holistic privacy-preserving solution.

In addition, it could also be interesting to further study the robustness of *PASTEL* with larger and more complex models. Indeed, *PASTEL* relies on the notion of a distance function between member data and non-member data. How such a distance function can handle more or less complex models, and whether some distance functions are more appropriate to some types of models could be interesting to explore. The curse of dimensionality, a well-known problem in high-dimensional data analysis, poses a significant and multifaceted obstacle to the effective application of *PASTEL*, designed to enhance the privacy and security of machine learning models, encounters challenges when dealing with datasets characterized by an increasingly large number of features or dimensions. As the dimensionality of the data grows, a crucial concern emerges: the available data points become progressively sparse within the expansive high-dimensional space. This sparsity fundamentally impacts the accuracy of probability and distribution estimations, which form the bedrock of techniques like *PASTEL*. The reliance on precise measurements of divergence between distributions, such as the Jensen-Shannon divergence often employed in *PASTEL*, can become compromised [49]. In high-dimensional scenarios, the Jensen-Shannon divergence may lose its reliability and informativeness, potentially leading to suboptimal regularization strategies and thereby diminishing *PASTEL*'s effectiveness in bridging the internal generalization gap.

Furthermore, FL privacy protection encounters various challenges when dealing with some types of datasets. Biased datasets pose a significant obstacle [21], as the model defending against membership inference attacks can inherit biases from its training data, rendering standard mitigation strategies inadequate. Moreover, datasets exhibiting pronounced patterns also undermine *PASTEL*'s efficacy in countering membership inference attacks. These conspicuous patterns, whether attributed to attributes, features, or classes, can be exploited by adversaries to deduce membership more easily. future research directions include evaluating *PASTEL* with biased dataset. In the event of unsatisfactory outcomes, potential strategies could encompass a combination of approaches aimed at mitigating bias. Enhancing

fairness and evaluating the defended model's equity could be key in reducing exploitable patterns. Moreover, to counter strong patterns within data, *PASTEL* could adopt techniques disrupting inherent pattern structures while preserving data utility, including advanced data augmentation, generative models for diverse synthetic data, and targeted regularization methods. The aim is to introduce uncertainty, obscuring membership-related information while upholding the model's predictive accuracy. In summary, the challenges*PASTEL* faces with various types of datasets require innovative and tailored solutions. By addressing the complexities of dataset characteristics, biases, and strong patterns, *PASTEL* can be enhanced to provide robust defense against membership inference attacks. This necessitates a comprehensive approach that combines advanced techniques, adaptation to high dimensional spaces, and a deeper understanding of the underlying data dynamics.

9.4 Summary

This chapter evaluates *PASTEL*, an innovative defense method in federated learning designed to counter membership inference attacks. Our empirical assessment was conducted on seven real-world datasets, encompassing four image datasets (Cifar-10, Cifar-100, GTSRB, and CelebA), three tabular datasets (MotionSense, Purchase100, and Texas100), and one raw audio waveform dataset (Speech Commands). We employed widely recognized neural network architectures for this evaluation. Our results show that *PASTEL* outperforms state-of-the-art FL defense mechanisms. Three main benefits can be highlighted, namely, best privacy protection, high model accuracy and no perceptible computational overhead. This work opens a set of interesting research directions. First, it could be interesting to study the effectiveness of the proposed approach in protecting against other types of privacy attacks, such as property inference attacks and model inversion attacks [46]. Property inference attacks leverage machine learning models' output probabilities to infer sensitive information about specific data attributes, and model inversion attacks aim to reconstruct training data by reverse engineering models. Another research direction is to enrich the multi-objective approach with additional interesting properties of FL systems, such as FL fairness for providing non-biased FL models [23], and FL robustness to better protect FL systems against byzantine participants [13]. Furthermore, *PASTEL* might be challenged by larger-scale and more complex models, which would require the exploration of other types of distance functions that would be more appropriate for reducing the generalization gap of such complex models.

Part III

Conclusion and Perspectives

Chapter 10

Conclusion and Perspectives

10.1 Conclusion

Federated Learning has emerged as a groundbreaking paradigm for privacy-preserving machine learning in ubiquitous computing systems. It allows multiple decentralized data owners, to collaboratively train models without exposing their raw data to external entities. Instead, clients send only their model updates to a central server, which aggregates them to produce a global model. This approach has found applications in various domains, including healthcare, autonomous driving, and predictive text input. In this thesis, we thoroughly investigated federated learning, focusing on addressing two main challenges: robustness and privacy. We began by enhancing the resilience of FL, specifically against poisoning attacks. Then, we transitioned to protecting privacy in FL, with a focus on mitigating inference attacks. We introduced three innovative mechanisms in the field of Federated Learning to enhance robustness and privacy preservation. These mechanisms are known as *ARMOR* for robust FL, *PASTEL* and DINAR for privacy preservation in FL.

ARMOR focuses on mitigating edge-case backdoor attacks, a particularly challenging class of poisoning attacks in FL. These attacks aim to corrupt the global FL model with subtle misclassifications of rare and underrepresented data points. Existing FL defense mechanisms struggle to detect and counter these attacks effectively. *ARMOR* introduces a novel Generative Adversarial Network architecture to synthesize class representatives for the global model, allowing for the detection of edge-case backdoors. It achieves a remarkable 95% resilience to such attacks without compromising model quality. *PASTEL* is a pioneering FL defense mechanism designed to counter membership inference attacks while preserving model accuracy and minimizing computational overhead. It employs a multi-objective learning approach to simultaneously reduce model loss and narrow the generalization gap between member and non-member data. *PASTEL* significantly reduces the success rate of MIAs by up to -28% across various datasets and neural network architectures, offering robust privacy protection. DINAR focuses on fine-grained privacy protection in FL, and combines fine-grained obfuscation of private layer parameters, client model personalization, with adaptive gradient descent for maximizing model accuracy, while efficiently protecting the models against MIAs in a non-intrusive way.

10.2 Perspectives

In the following, we outline potential research directions stemming from the investigations into privacy preservation and robustness covered in this thesis. These research avenues extend our understanding and open new possibilities for further advancement in the field of federated learning privacy and security.

10.2.1 Enhancing Robustness in Federated Learning

ARMOR, our defense mechanism against edge-case backdoors, demonstrates its effectiveness in countering such attacks in FL systems. This novel approach has significant potential across a spectrum of learning tasks in FL. For instance, it can be applied to binary classification tasks, such as diagnosing diseases or detecting spam emails, as well as multi-class classification tasks, encompassing domains like agriculture, medicine, and chemistry. While *ARMOR*'s applicability is broad, it may face practical limitations when dealing with extremely complex classifiers with a vast number of classes, such as those found in natural language processing and text classification.

One promising research direction is to optimize the computational cost associated with *ARMOR*. While parallel GPU computations and advancements in GAN convergence speed may help mitigate this cost, further investigations into efficient training techniques and optimizations are warranted. Additionally, assessing *ARMOR*'s efficacy in protecting against different types of privacy attacks, such as property inference and model inversion attacks, could broaden its scope of applicability.

10.2.2 Enhancing Privacy in Federated Learning

PASTEL, our multi-objective approach for privacy protection against membership inference attacks, has demonstrated its effectiveness without introducing significant computational overhead. This research direction offers several intriguing avenues for further exploration. Firstly, it could be valuable to investigate the suitability of *PASTEL* in safeguarding FL systems against other types of privacy attacks, such as property inference attacks and model inversion attacks.

Furthermore, enriching the multi-objective approach of *PASTEL* with additional dimensions of FL, such as fairness and robustness, presents exciting prospects. Research into FL fairness aims to provide non-biased FL models, and incorporating fairness considerations into *PASTEL* could contribute to more equitable FL outcomes. Additionally, exploring ways to bolster FL robustness against byzantine participants, building upon the principles of *PASTEL*, could enhance the security of FL systems.

As *PASTEL* continues to prove its efficacy, it may encounter challenges when applied to larger-scale and more complex models. Investigating alternative distance functions that are better suited for reducing the generalization gap in such scenarios could be a promising avenue for future research.

DINAR's focus on fine-grained privacy protection in FL positions it as a versatile defense mechanism. Beyond its role in mitigating membership inference attacks, DINAR can be leveraged to address various privacy concerns. Research directions

in this context include evaluating DINAR's effectiveness against different privacy attack types, such as property inference and model inversion attacks.

Another intriguing avenue for exploration is the development of automated methods for identifying the most privacy-sensitive neural network layers. Such methods could adapt to specific threat models, privacy attack types, and FL model architectures, enhancing the precision and efficiency of privacy protection mechanisms like DINAR.

Bibliography

[1] Mohammed Abaoud, Muqrin A. Almuqrin, and Mohammad Faisal Khan. "Advancing Federated Learning Through Novel Mechanism for Privacy Preservation in Healthcare Applications". In: *IEEE Access* 11 (2023), pp. 83562–83579. DOI: 10.1109/ACCESS.2023.3301162. URL: https://doi.org/10.1109/ACCESS.2023.3301162.

[2] Fayez Alharbi, Lahcen Ouarbya, and Jamie A. Ward. "Comparing Sampling Strategies for Tackling Imbalanced Data in Human Activity Recognition". In: *Sensors* 22.4 (2022), p. 1373.

[3] Kimon Antonakopoulos et al. "AdaGrad Avoids Saddle Points". In: *International Conference on Machine Learning, ICML 2022*. Vol. 162. Proceedings of Machine Learning Research. Baltimore, Maryland, USA, 2022, pp. 731–771.

[4] Georgios Arvanitidis, Lars Kai Hansen, and Søren Hauberg. *Latent Space Oddity: on the Curvature of Deep Generative Models*. 2017.

[5] Giuseppe Ateniese et al. "Hacking smart machines with smarter ones: How to extract meaningful data from machine learning classifiers". In: *Int. J. Secur. Networks* 10.3 (2015), pp. 137–150. DOI: 10.1504/IJSN.2015.071829. URL: https://doi.org/10.1504/IJSN.2015.071829.

[6] Giuseppe Ateniese et al. "Hacking smart machines with smarter ones: How to extract meaningful data from machine learning classifiers". In: *Int. J. Secur. Networks* 10 (2015), pp. 137–150.

[7] Eugene Bagdasaryan et al. "How To Backdoor Federated Learning". In: *The 23rd International Conference on Artificial Intelligence and Statistics, AISTATS 2020, 26-28 August 2020, Online [Palermo, Sicily, Italy]*. Ed. by Silvia Chiappa and Roberto Calandra. Vol. 108. Proceedings of Machine Learning Research. Palermo, Sicily, Italy: PMLR, 2020, pp. 2938–2948. URL: http://proceedings.mlr.press/v108/bagdasaryan20a.html.

[8] Xiaolong Bai et al. "Learning ECOC Code Matrix for Multiclass Classification with Application to Glaucoma Diagnosis". In: *J. Medical Syst.* 40.4 (2016), 78:1–78:10. DOI: 10.1007/s10916-016-0436-2. URL: https://doi.org/10.1007/s10916-016-0436-2.

[9] Gilad Baruch, Moran Baruch, and Yoav Goldberg. "A Little Is Enough: Circumventing Defenses For Distributed Learning". In: *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019*, ed. by Hanna M. Wallach et al. Vancouver, BC, Canada: NIPS, 2019, pp. 8632–8642. URL: https://proceedings.neurips.cc/paper/2019/hash/ec1c59141046cd1866bbbcdfb6ae31d4-Abstract.html.

[10] David Bau et al. "Rewriting a Deep Generative Model". In: *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part I*. Ed. by Andrea Vedaldi et al. Vol. 12346. Lecture Notes in Computer Science. Springer, 2020, pp. 351–369. DOI: 10.1007/978-3-030-58452-8_21. URL: https://doi.org/10.1007/978-3-030-58452-8_21.

[11] Yassine Ben-Aboud et al. "On Adaptive Sampling Algorithms for IoT Devices". In: *IEEE International Conference on Communications (ICC 2021)*. Montreal, QC, Canada, June 2021, pp. 1–7.

[12] Arjun Nitin Bhagoji et al. "Analyzing Federated Learning through an Adversarial Lens". In: *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. Long Beach, California, USA: PMLR, 2019, pp. 634–643. URL: http://proceedings.mlr.press/v97/bhagoji19a.html.

[13] Peva Blanchard et al. "Machine Learning with Adversaries: Byzantine Tolerant Gradient Descent". In: *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017*, ed. by Isabelle Guyon et al. Long Beach, CA, USA: NIPS, 2017, pp. 119–129. URL: https://proceedings.neurips.cc/paper/2017/hash/f4b9ec30ad9f68f89b29639786cb62ef-Abstract.html.

[14] Peva Blanchard et al. "Machine Learning with Adversaries: Byzantine Tolerant Gradient Descent". In: *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*. Ed. by Isabelle Guyon et al. Long Beach, CA, USA: Advances in Neural Information Processing Systems, 2017, pp. 119–129.

[15] K. A. Bonawitz et al. "Practical Secure Aggregation for Federated Learning on User-Held Data". In: *NIPS Workshop on Private Multi-Party Machine Learning*. Barcelona, Spain: NIPS, 2016.

[16] Kallista A. Bonawitz et al. "Practical Secure Aggregation for Privacy-Preserving Machine Learning". In: *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, October 30 - November 03, 2017*. Ed. by Bhavani M. Thuraisingham et al. Dallas, TX, USA: ACM, 2017, pp. 1175–1191. DOI: 10.1145/3133956.3133982. URL: https://doi.org/10.1145/3133956.3133982.

[17] Kallista A. Bonawitz et al. "Practical Secure Aggregation for Privacy-Preserving Machine Learning". In: *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, October 30 - November 03, 2017*. Ed. by Bhavani Thuraisingham et al. ACM, 2017, pp. 1175–1191. DOI: 10.1145/3133956.3133982. URL: https://doi.org/10.1145/3133956.3133982.

[18] Xiaoyu Cao et al. "FLTrust: Byzantine-robust Federated Learning via Trust Bootstrapping". In: *28th Annual Network and Distributed System Security Symposium, NDSS 2021*, virtually, February 21-25, 2021: The Internet Society, 2021. URL: https://www.ndss-symposium.org/ndss-paper/fltrust-byzantine-robust-federated-learning-via-trust-bootstrapping/.

[19] Xiaoyu Cao et al. "FLTrust: Byzantine-robust Federated Learning via Trust Bootstrapping". In: *28th Annual Network and Distributed System Security Symposium, NDSS 2021, February 21-25, 2021*. virtually: The Internet Society, 2021.

[20] Mahawaga Arachchige Pathum Chamikara et al. "Local Differential Privacy for Federated Learning". In: *Computer Security - ESORICS 2022 - 27th European Symposium on Research in Computer Security, Copenhagen, Denmark, September 26-30, 2022, Proceedings, Part I*. Ed. by Vijayalakshmi Atluri et al. Vol. 13554. Lecture Notes in Computer Science. Springer, 2022, pp. 195–216. DOI: 10.1007/978-3-031-17140-6_10. URL: https://doi.org/10.1007/978-3-031-17140-6_10.

[21] Hongyan Chang and Reza Shokri. "Bias Propagation in Federated Learning". In: *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL: https://openreview.net/pdf?id=V7CYzdruWdm.

[22] Archana Chaudhary, Savita Kolhe, and Raj Kamal. "A Hybrid Ensemble for Classification in Multiclass Datasets: An Application to Oilseed Disease Dataset". In: *Comput. Electron. Agric.* 124 (2016), pp. 65–72. DOI: 10.1016/j.compag.2016.03.026. URL: https://doi.org/10.1016/j.compag.2016.03.026.

[23] Bhaskar Ray Chaudhury et al. "Fairness in Federated Learning via Core-Stability". In: *NeurIPS*. 2022. URL: http://papers.nips.cc/paper_files/paper/2022/hash/25e92e33ac8c35fd49f394c37f21b6da-Abstract-Conference.html.

[24] Dingfan Chen, Ning Yu, and Mario Fritz. "RelaxLoss: Defending Membership Inference Attacks without Losing Utility". In: *The Tenth International Conference on Learning Representations, ICLR 2022, April 25-29, 2022*. Virtual Event: OpenReview.net, 2022.

[25] Dingfan Chen, Ning Yu, and Mario Fritz. "RelaxLoss: Defending Membership Inference Attacks without Losing Utility". In: *The Tenth International Conference on Learning Representations, ICLR 2022, April 25-29, 2022*. Virtual Event: OpenReview.net, 2022.

[26] Zhenzhu Chen et al. "Secure Collaborative Deep Learning Against GAN Attacks in the Internet of Things". In: *IEEE Internet Things J.* 8.7 (2021), pp. 5839–5849.

[27] Wei Dai et al. "Very Deep Convolutional Neural Networks for Raw Waveforms". In: *2017 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2017*. New Orleans, LA, USA, 2017, pp. 421–425.

[28] Ozden O. Dalgic et al. "Mapping of Critical Events in Disease Progression Through Binary Classification: Application to Amyotrophic Lateral Sclerosis". In: *J. Biomed. Informatics* 123 (2021), p. 103895. DOI: 10.1016/j.jbi.2021.103895. URL: https://doi.org/10.1016/j.jbi.2021.103895.

[29] Bao Gia Doan, Ehsan Abbasnejad, and Damith C. Ranasinghe. "Februus: Input Purification Defense Against Trojan Attacks on Deep Neural Network Systems". In: *ACSAC '20: Annual Computer Security Applications Conference, Virtual Event / Austin, TX, USA, 7-11 December, 2020*. ACM, 2020, pp. 897–912. DOI: 10.1145/3427228.3427264. URL: https://doi.org/10.1145/3427228.3427264.

[30] John Duchi and Elad Hazan. "Adaptive Subgradient Methods for Online Learning and Stochastic Optimization". In: *Journal of Machine Learning Research* 12 (2011), pp. 2121–2159.

[31] Haokun Fang and Quan Qian. "Privacy Preserving Machine Learning with Homomorphic Encryption and Federated Learning". In: *Future Internet* 13.4 (2021), p. 94.

[32] Minghong Fang et al. "Local Model Poisoning Attacks to Byzantine-Robust Federated Learning". In: *29th USENIX Security Symposium, USENIX Security 2020, August 12-14, 2020*. Ed. by Srdjan Capkun and Franziska Roesner. Virtual: USENIX Association, 2020, pp. 1605–1622. URL: https://www.usenix.org/conference/usenixsecurity20/presentation/fang.

[33] Yan Feng et al. "sqSGD: Locally Private and Communication Efficient Federated Learning". In: *ArXiv* abs/2206.10565 (2022). arXiv: 2206.10565.

[34] Chong Fu et al. "Label Inference Attacks Against Vertical Federated Learning". In: *31st USENIX Security Symposium, USENIX Security 2022, Boston, MA, USA, August 10-12, 2022*. Ed. by Kevin R. B. Butler and Kurt Thomas. USENIX Association, 2022, pp. 1397–1414. URL: https://www.usenix.org/conference/usenixsecurity22/presentation/fu-chong.

[35] Shuhao Fu et al. *Attack-Resistant Federated Learning with Residual-Based Reweighting*. https://github.com/fushuhao6/Attack-Resistant-Federated-Learning. 2019.

[36] Clement Fung, Chris J. M. Yoon, and Ivan Beschastnikh. "The Limitations of Federated Learning in Sybil Settings". In: *23rd International Symposium on Research in Attacks, Intrusions and Defenses (RAID 2020)*. San Sebastian: USENIX, Oct. 2020. ISBN: 978-1-939133-18-2. URL: https://www.usenix.org/conference/raid2020/presentation/fung.

[37] Sébastien Rouault Georgios Damaskinos Arsany Guirguis. *AggregaThor*. https://github.com/LPD-EPFL/AggregaThor. 2019.

[38] Ian J. Goodfellow et al. "Generative Adversarial Nets". In: *Annual Conference on Neural Information Processing Systems (NeurIPS 2014)*. Montreal, Canada: NIPS, 2014. URL: https://proceedings.neurips.cc/paper/2014/hash/5ca3e9b122f61f8f06494c97b1afccf3-Abstract.html.

[39] Tianyu Gu et al. "BadNets: Evaluating Backdooring Attacks on Deep Neural Networks". In: *IEEE Access* 7 (2019), pp. 47230–47244. DOI: 10.1109/ACCESS.2019.2909068.

[40] Alejandro Guerra-Manzanares et al. "Privacy-Preserving Machine Learning for Healthcare: Open Challenges and Future Perspectives". In: *Trustworthy Machine Learning for Healthcare - First International Workshop, TML4H 2023, Virtual Event, May 4, 2023, Proceedings*. Ed. by Hao Chen and Luyang Luo. Vol. 13932. Lecture Notes in Computer Science. Springer, 2023, pp. 25–40. DOI: 10.1007/978-3-031-39539-0_3. URL: https://doi.org/10.1007/978-3-031-39539-0_3.

[41] Farzin Haddadpour et al. "Federated Learning with Compression: Unified Analysis and Sharp Guarantees". In: *The 24th International Conference on Artificial Intelligence and Statistics, AISTATS 2021, April 13-15, 2021*. Ed. by Arindam Banerjee and Kenji Fukumizu. Vol. 130. Proceedings of Machine Learning Research. Virtual Event: PMLR, 2021, pp. 2350–2358.

[42] Björn Hanneke, Lorenz Baum, and Oliver Hinz. "GDPR Privacy Type Clustering: Motivational Factors for Consumer Data Sharing". In: *31st European Conference on Information Systems - Co-creating Sustainable Digital Futures, ECIS 2023, Kristiansan, Norway, June 11-16, 2023*. Ed. by Margunn Aanestad et al. 2023. URL: https://aisel.aisnet.org/ecis2023_rp/409.

[43] Andrew Hawkins. "Tesla didn't fix an Autopilot problem for three years, and now another person is dead". https://www.theverge.com/2019/5/17/18629214/tesla-autopilot-crash-death-josh-brown-jeremy-banner. 2019.

[44] Kaiming He et al. "Deep Residual Learning for Image Recognition". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016*. Las Vegas, NV, USA: IEEE Computer Society, 2016, pp. 770–778.

[45] Lie He, Sai Praneeth Karimireddy, and Martin Jaggi. "Byzantine-Robust Learning on Heterogeneous Datasets via Resampling". In: *arXiv:2006.09365* 1 (June 2020).

[46] Seira Hidano et al. "Model Inversion Attacks for Online Prediction Systems: Without Knowledge of Non-Sensitive Attributes". In: *IEICE Trans. Inf. Syst.* 101-D.11 (2018), pp. 2665–2676.

[47] Briland Hitaj, Giuseppe Ateniese, and Fernando Pérez-Cruz. "Deep Models Under the GAN: Information Leakage from Collaborative Deep Learning". In: *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, October 30 - November 03, 2017*. Ed. by Bhavani Thuraisingham et al. Dallas, TX, USA: ACM, 2017, pp. 603–618.

[48] Sebastian Houben et al. "Detection of Traffic Signs in Real-World Images: The German Traffic Sign Detection Benchmark". In: *The 2013 International Joint Conference on Neural Networks, IJCNN 2013*. Dallas, TX, USA, 2013, pp. 1–8.

[49] Jhoan Keider Hoyos-Osorio and Luis G. Sánchez Giraldo. "The Representation Jensen-Shannon Divergence". In: *CoRR* abs/2305.16446 (2023). DOI: 10.48550/arXiv.2305.16446. arXiv: 2305.16446. URL: https://doi.org/10.48550/arXiv.2305.16446.

[50] Tzu-Ming Harry Hsu, Hang Qi, and Matthew Brown. "Measuring the Effects of Non-identical Data Distribution for Federated Visual Classification". In: *arXiv:1909.06335* 1 (Sept. 2019).

[51] Dongjun Hwang, Hyunsu Mun, and Youngseok Lee. "Improving response time of home IoT services in federated learning". In: *SAC '22: The 37th ACM/SIGAPP Symposium on Applied Computing, Virtual Event, April 25 - 29, 2022*. Ed. by Jiman Hong et al. Virtual Event: ACM, 2022, pp. 157–163. DOI: 10.1145/3477314.3508380. URL: https://doi.org/10.1145/3477314.3508380.

[52] Jinyuan Jia et al. "MemGuard: Defending against Black-Box Membership Inference Attacks via Adversarial Examples". In: *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, CCS*. London, UK, 2019, pp. 259–274.

[53] Peter Kairouz et al. "Advances and Open Problems in Federated Learning". In: *ArXiv* abs/1912.04977 (2019). arXiv: 1912.04977.

[54] Hongyi Wang Kartik Sreenivasan. *OOD Federated Learning*. https://github.com/ksreenivasan/OOD-Federated-Learning. 2020.

[55] Yigitcan Kaya and Tudor Dumitras. "When Does Data Augmentation Help With Membership Inference Attacks?" In: *Proceedings of the 38th International Conference on Machine Learning*. Ed. by Marina Meila and Tong Zhang. Vol. 139. Proceedings of Machine Learning Research. PMLR, 2021, pp. 5345–5355. URL: https://proceedings.mlr.press/v139/kaya21a.html.

[56] Diederik P. Kingma and Jimmy Ba. "Adam: A Method for Stochastic Optimization". In: *3rd International Conference on Learning Representations*. San Diego, CA, USA, 2015.

[57] Diederik P. Kingma and Jimmy Ba. "Adam: A Method for Stochastic Optimization". In: *ArXiv* abs/1412.6980 (2015).

[58] Samuel Kotz, Narayanaswamy Balakrishnan, and Norman L Johnson. *Continuous multivariate distributions, Volume 1: Models and applications*. Vol. 1. John Wiley & Sons, 2004.

[59] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. *CIFAR-10 (Canadian Institute for Advanced Research)*. http://www.cs.toronto.edu/ kriz/cifar.html. 2010.

[60] Pradeep Kumar. "Predictive Analytics for Spam Email Classification Using Machine Learning techniques". In: *Int. J. Comput. Appl. Technol.* 64.3 (2020), pp. 282–296. DOI: 10.1504/IJCAT.2020.111844. URL: https://doi.org/10.1504/IJCAT.2020.111844.

[61] Harold J Kushner and George G Yin. *Stochastic Approximation and Optimization of Random Systems*. Philadelphia: SIAM Series on Optimization, Society for Industrial and Applied Mathematics, 1997.

[62] Thomas Lebrun et al. "MixNN: Protection of Federated Learning against Inference Attacks by Mixing Neural Network Layers". In: *Middleware '22: 23rd International Middleware Conference*. Quebec, Canada, 2022, pp. 135–147.

[63] Yann LeCun and Corinna Cortes. *MNIST handwritten digit database*. https://yann.lecun.com/exdb/mnist. 2010.

[64] Jiacheng Li, Ninghui Li, and Bruno Ribeiro. "Membership Inference Attacks and Defenses in Supervised Learning via Generalization Gap". In: *ArXiv* abs/2002.12062 (2020). arXiv: 2002.12062.

[65] Yong Li et al. "Privacy-Preserving Federated Learning Framework Based on Chained Secure Multiparty Computing". In: *IEEE Internet of Things Journal* 8.8 (2021), pp. 6178–6186.

[66] Pengrui Liu, Xiangrui Xu, and Wei Wang. "Threats, attacks and defenses to federated learning: issues, taxonomy and perspectives". In: *Cybersecur.* 5.1 (2022), p. 4.

[67] Ziwei Liu et al. "Deep Learning Face Attributes in the Wild". In: *2015 IEEE International Conference on Computer Vision, ICCV 2015*. Santiago, Chile, 2015, pp. 3730–3738.

[68] Gavin R. Lloyd et al. "Learning Vector Quantization for Multiclass Classification: Application to Characterization of Plastics". In: *J. Chem. Inf. Model.* 47.4 (2007), pp. 1553–1563. DOI: 10.1021/ci700019q. URL: https://doi.org/10.1021/ci700019q.

[69] Lingjuan Lyu, Han Yu, and Qiang Yang. "Threats to Federated Learning: A Survey". In: *ArXiv* abs/2003.02133 (2020). arXiv: 2003.02133.

[70] Mohammad Malekzadeh et al. "Mobile Sensor Data Anonymization". In: *Proceedings of the International Conference on Internet of Things Design and Implementation*. IoTDI '19. Montreal, Quebec, Canada: ACM, 2019, pp. 49–58. ISBN: 978-1-4503-6283-2.

[71] Brendan McMahan et al. "Communication-Efficient Learning of Deep Networks from Decentralized Data". In: *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, AISTATS 2017*. Vol. 54. Proceedings of Machine Learning Research. Fort Lauderdale, FL, USA, 2017, pp. 1273–1282.

[72] H. Brendan McMahan et al. "Learning Differentially Private Recurrent Language Models". In: *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL: https://openreview.net/forum?id=BJ0hF1Z0b.

[73] H. Brendan McMahan et al. "Learning Differentially Private Recurrent Language Models". In: *6th International Conference on Learning Representations, ICLR 2018*. Vancouver, BC, Canada: OpenReview.net, Apr. 2018. URL: https://openreview.net/forum?id=BJ0hF1Z0b.

[74] Frank McSherry. "Privacy integrated queries: an extensible platform for privacy-preserving data analysis". In: *Commun. ACM* 53.9 (2010), pp. 89–97.

[75] Luca Melis et al. "Exploiting Unintended Feature Leakage in Collaborative Learning". In: *2019 IEEE Symposium on Security and Privacy (SP 2019)* 10.1109 (2019). DOI: 10.1109/sp.2019.00029. URL: http://dx.doi.org/10.1109/SP.2019.00029.

[76] M.L. Menendez et al. "The Jensen-Shannon divergence". In: *Journal of the Franklin Institute* 334.2 (1997), pp. 307–318. ISSN: 0016-0032.

[77] Aghiles Ait Messaoud et al. "Shielding Federated Learning Systems against Inference Attacks with ARM TrustZone". In: *Middleware '22: 23rd International Middleware Conference*. QC, Canada, 2022, pp. 335–348.

[78] Mehdi Mirza and Simon Osindero. "Conditional Generative Adversarial Nets". In: *CoRR* abs/1411.1784 (2014). arXiv: 1411.1784. URL: http://arxiv.org/abs/1411.1784.

[79] Fan Mo et al. "Layer-wise Characterization of Latent Information Leakage in Federated Learning". In: *The International Conference on Learning Representations (ICLR) - Workshop on Distributed and Private Machine Learning*. Virtual, 2021.

[80] Fan Mo et al. "Quantifying Information Leakage from Gradients". In: *ArXiv* abs/2105.13929 (2021). arXiv: 2105.13929.

[81] Mahesh Chandra Mukkamala and Matthias Hein. "Variants of RMSProp and Adagrad with Logarithmic Regret Bounds". In: *Proceedings of the 34th International Conference on Machine Learning, ICML 2017*. Vol. 70. Proceedings of Machine Learning Research. Sydney, NSW, Australia, 2017, pp. 2545–2553.

[82] Mohammad Naseri, Jamie Hayes, and Emiliano De Cristofaro. "Toward Robustness and Privacy in Federated Learning: Experimenting with Local and Central Differential Privacy". In: *ArXiv* abs/2009.03561 (2020). arXiv: 2009.03561.

[83] Milad Nasr, Reza Shokri, and Amir Houmansadr. "Comprehensive Privacy Analysis of Deep Learning: Passive and Active White-box Inference Attacks against Centralized and Federated Learning". In: *IEEE Symposium on Security and Privacy, SP 2019*. San Francisco, CA, USA: IEEE, May 2019, pp. 739–753. DOI: 10.1109/SP.2019.00065. URL: https://doi.org/10.1109/SP.2019.00065.

[84] Milad Nasr, Reza Shokri, and Amir Houmansadr. "Comprehensive Privacy Analysis of Deep Learning: Passive and Active White-box Inference Attacks against Centralized and Federated Learning". In: *2019 IEEE Symposium on Security and Privacy*. San Francisco, CA, USA, 2019, pp. 739–753.

[85] Adam Palanica and Yan Fossat. "Medication Name Comprehension of Intelligent Virtual Assistants: A Comparison of Amazon Alexa, Google Assistant, and Apple Siri Between 2019 and 2021". In: *Frontiers Digit. Health* 3 (2021), p. 669971. DOI: 10.3389/fdgth.2021.669971. URL: https://doi.org/10.3389/fdgth.2021.669971.

[86] Adam Paszke et al. "PyTorch: An Imperative Style, High-Performance Deep Learning Library". In: *Advances in Neural Information Processing Systems (NeurIPS 2019)*. Vancouver Convention Centre, CA: NIPS, Dec. 2019.

[87] Guilherme Perin and Stjepan Picek. *On the Influence of Optimizers in Deep Learning-based Side-channel Analysis*. Cryptology ePrint Archive, Paper 2020/977. https://eprint.iacr.org/2020/977. 2020.

[88] Venkata Krishna Pillutla, Sham M. Kakade, and Zaid Harchaoui. "Robust Aggregation for Federated Learning". In: *CoRR* abs/1912.13445 (2019). arXiv: 1912.13445. URL: http://arxiv.org/abs/1912.13445.

[89] Sameera Ramasinghe et al. *Rethinking conditional GAN training: An approach using geometrically structured latent manifolds*. 2020. arXiv: 2011.13055 [cs.CV].

[90] Nicola Rieke et al. "The future of digital health with federated learning". In: *npj Digit. Medicine* 3 (2020). DOI: 10.1038/s41746-020-00323-1. URL: https://doi.org/10.1038/s41746-020-00323-1.

[91] Ahmed Salem et al. "ML-Leaks: Model and Data Independent Membership Inference Attacks and Defenses on Machine Learning Models". In: *26th Annual Network and Distributed System Security Symposium, NDSS 2019, February 24-27, 2019*. San Diego, California, USA: The Internet Society, 2019.

[92] Virat Shejwalkar and Amir Houmansadr. "Manipulating the Byzantine: Optimizing Model Poisoning Attacks and Defenses for Federated Learning". In: *Network and Distributed Systems Security Symposium (NDSS 2021)*. Virtual Conference: The Internet Society, Feb. 2021.

[93] Reza Shokri et al. "Membership Inference Attacks Against Machine Learning Models". In: *IEEE Symposium on Security and Privacy, S&P 2017*. San Jose, CA, USA, 2017, pp. 3–18.

[94] Shrebox. "Privacy Attacks in Machine Learning". https://github.com/shrebox/Privacy-Attacks-in-Machine-Learning. 2017.

[95] Karen Simonyan and Andrew Zisserman. "Very Deep Convolutional Networks for Large-Scale Image Recognition". In: *3rd International Conference on Learning Representations, ICLR 2015*. San Diego, CA, USA, 2015.

[96] Leslie N. Smith. "A Disciplined Approach to Neural Network Hyper-Parameters: Part 1 - Learning Rate, Batch Size, Momentum, and Weight Decay". In: *ArXiv* abs/1803.09820 (2018). arXiv: 1803.09820.

[97] Matthew Staib et al. "Escaping Saddle Points with Adaptive Gradient Methods". In: *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019*, ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. Long Beach, California, USA: PMLR, 2019, pp. 5956–5965.

[98] Ziteng Sun et al. "Can You Really Backdoor Federated Learning?" In: *CoRR* abs/1911.07963 (2019). arXiv: 1911.07963. URL: http://arxiv.org/abs/1911.07963.

[99] Ziteng Sun et al. "Can You Really Backdoor Federated Learning?" In: *ArXiv* abs/1911.07963 (2019). ArXiv: 1911.07963.

[100] Elham Tabassi et al. *A Taxonomy and Terminology of Adversarial Machine Learning*. 2019.

[101] Vale Tolpegin et al. "Data Poisoning Attacks Against Federated Learning Systems". In: *25th European Symposium on Research in Computer Security, (ESORICS 2020)*. July 2020.

[102] Aleksei Triastcyn and Boi Faltings. "Federated Generative Privacy". In: *IEEE Intell. Syst.* 35.4 (2020), pp. 50–57.

[103] Stacey Truex et al. "Demystifying Membership Inference Attacks in Machine Learning as a Service". In: *IEEE Transactions on Services Computing* 14.6 (2021), pp. 2073–2089.

[104] Hongyi Wang et al. "Attack of the Tails: Yes, You Really Can Backdoor Federated Learning". In: *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020*. Ed. by Hugo Larochelle et al. virtual: NIPS, 2020. URL: https://proceedings.neurips.cc/paper/2020/hash/b8ffa41d4e492f0fad2f13e29e1762eb-Abstract.html.

[105] Jiacheng Wang et al. "Approx-SMOTE Federated Learning Credit Card Fraud Detection System". In: *47th IEEE Annual Computers, Software, and Applications Conference, COMPSAC 2023, Torino, Italy, June 26-30, 2023*. Ed. by Hossain Shahriar et al. IEEE, 2023, pp. 1370–1375. DOI: 10.1109/COMPSAC57700.2023.00208. URL: https://doi.org/10.1109/COMPSAC57700.2023.00208.

[106] Zhibo Wang et al. "Poisoning-Assisted Property Inference Attack Against Federated Learning". In: *IEEE Trans. Dependable Secur. Comput.* 20.4 (2023), pp. 3328–3340. DOI: 10.1109/TDSC.2022.3196646. URL: https://doi.org/10.1109/TDSC.2022.3196646.

[107] Pete Warden. "Speech Commands: A Dataset for Limited-Vocabulary Speech Recognition". In: *ArXiv* abs/1804.03209 (2018).

[108] Di Wu et al. "Understanding and Defending Against White-box Membership Inference Attack in Deep Learning". In: *Knowl. Based Syst.* 259 (2023), p. 110014.

[109] Han Xiao, Kashif Rasul, and Roland Vollgraf. "Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms". In: *ArXiv* abs/1708.07747 (2017). arXiv: 1708.07747.

[110] Cong Xie, Oluwasanmi Koyejo, and Indranil Gupta. "Generalized Byzantine-tolerant SGD". In: *CoRR* abs/1802.10116 (2018). arXiv: 1802.10116. URL: http://arxiv.org/abs/1802.10116.

[111] Runhua Xu et al. "HybridAlpha: An Efficient Approach for Privacy-Preserving Federated Learning". In: *Proceedings of the 12th ACM workshop on artificial intelligence and security*. 2019, pp. 13–23.

[112] Timothy Yang et al. "Applied Federated Learning: Improving Google Keyboard Query Suggestions". In: *arXiv:1812.02903* 1 (Dec. 2018).

[113] Xue Yang et al. "An Accuracy-Lossless Perturbation Method for Defending Privacy Attacks in Federated Learning". In: *WWW '22: The ACM Web Conference 2022, Virtual Event, Lyon, France, April 25 - 29, 2022*. Ed. by Frédérique Laforest et al. ACM, 2022, pp. 732–742. DOI: 10.1145/3485447.3512233. URL: https://doi.org/10.1145/3485447.3512233.

[114] Samuel Yeom et al. "Privacy Risk in Machine Learning: Analyzing the Connection to Overfitting". In: *31st IEEE Computer Security Foundations Symposium, CSF 2018, July 9-12, 2018*. Oxford, United Kingdom: IEEE Computer Society, 2018, pp. 268–282.

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

[116] Ashkan Yousefpour et al. "Opacus: User-Friendly Differential Privacy Library in PyTorch". In: *ArXiv* abs/2109.12298 (2021). arXiv: 2109.12298.

[117] Hongyi Zhang, Jan Bosch, and Helena Holmström Olsson. "End-to-End Federated Learning for Autonomous Driving Vehicles". In: *International Joint Conference on Neural Networks, IJCNN 2021*. Shenzhen, China: IEEE, 2021, pp. 1–8. DOI: 10.1109/IJCNN52387.2021.9533808. URL: https://doi.org/10.1109/IJCNN52387.2021.9533808.

[118] Jiale Zhang et al. "PEFL: A Privacy-Enhanced Federated Learning Scheme for Big Data Analytics". In: *2019 IEEE Global Communications Conference, GLOBECOM 2019, Waikoloa*, HI, USA, 2019, pp. 1–6.

[119] Junpeng Zhang et al. "A survey on security and privacy threats to federated learning". In: *2021 International Conference on Networking and Network Applications, NaNA 2021, Lijiang City, China, October 29 - Nov. 1, 2021*. IEEE, 2021, pp. 319–326. DOI: 10.1109/NaNA53684.2021.00062. URL: https://doi.org/10.1109/NaNA53684.2021.00062.

[120] Yue Zhao et al. "Federated Learning with Non-IID Data". In: *ArXiv* abs/1806.00582 (2018). URL: https://api.semanticscholar.org/CorpusID:46936175.

[121] Yifeng Zheng et al. "Aggregation Service for Federated Learning: An Efficient, Secure, and More Resilient Realization". In: *IEEE Transactions on Dependable and Secure Computing* 20.2 (2023), pp. 988–1001. DOI: 10.1109/TDSC.2022.3146448.

[122] Jiachen Zhong, Xuanqing Liu, and Cho-Jui Hsieh. "Improving the Speed and Quality of GAN by Adversarial Training". In: *ArXiv* abs/2008.03364 (2020). arXiv: 2008.03364.

[123] Ligeng Zhu and Song Han. "Deep Leakage from Gradients". In: *Springer*. Lecture Notes in Computer Science 12500 (2020). Ed. by Qiang Yang, Lixin Fan, and Han Yu, pp. 17–31.

FOLIO ADMINISTRATIF

THESE DE L'INSA LYON, MEMBRE DE L'UNIVERSITE DE LYON

NOM : ELHATTAB
(avec précision du nom de jeune fille, le cas échéant)

DATE de SOUTENANCE : 27/10/2023

Prénoms : FATIMA

TITRE : Robust and Privacy Preserving Federated Learning

NATURE : Doctorat

Numéro d'ordre : AAAAINSALXXXX

Ecole doctorale : InfosMaths

Spécialité : Informatique

RESUME :

Dans le monde numérique en perpétuelle mutation d'aujourd'hui, l'apprentissage automatique est désormais une puissance essentielle et révolutionnaire, comme le démontrent de multiples recherches. Son impact profond s'étend à travers diverses industries, offrant des solutions et des innovations révolutionnaires qui ont remod- elé la manière dont nous interagissons avec la technologie et prenons des décisions. Des systèmes de recommandation améliorant la diffusion de contenu sur les plate- formes à la présence d'assistants personnels virtuels comme Siri et Alexa, capables de comprendre et de répondre à des commandes en langage naturel, les applica- tions de l'apprentissage automatique sont à la fois diverses et impactantes. Dans des domaines tels que la santé, il contribue au diagnostic des maladies, tandis que dans la finance, il renforce la détection de la fraude et l'évaluation des risques. Cette ubiquité de l'apprentissage automatique signifie non seulement une tendance tech- nologique, mais aussi un changement fondamental dans les approches de résolu- tion de problèmes et de prise de décisions. Cependant, cette vague d'innovation axée sur les données a soulevé une préoccupation primordiale : la protection de la vie privée des individus et de leurs données personnelles. Le Règlement général sur la protection des données (RGPD) illustre l'importance accrue de la protection des données à l'ère moderne. À mesure que l'apprentissage automatique s'intègre de plus en plus dans notre vie quotidienne, trouver un équilibre délicat entre les avancées technologiques et la protection de la vie privée individuelle est devenu im- pératif. De plus, L'attention portée à ces préoccupations a donné naissance au con- cept de l'apprentissage automatique préservant la vie privée, avec l'apprentissage fédéré émergeant comme une technique cruciale, redéfinissant l'apprentissage au- tomatique collaboratif en permettant à plusieurs parties de construire un modèle partagé sans partager leurs données brutes. L'apprentissage fédéré représente un paradigme prometteur en apprentissage au- tomatique, permettant la formation collaborative de modèles entre des appareils décentralisés dans des systèmes de calcul en périphérie. Cependant, il présente une vulnérabilité à diverses attaques. Cette recherche est divisée en deux axes princi- paux, chacun abordant des défis cruciaux en matière de sécurité et de confidentialité dans le contexte de l'apprentissage fédéré. Le premier axe se concentre sur la lutte contre les attaques d'empoisonnement pour un apprentissage fédéré robuste, où les adversaires cherchent à introduire des tâches nuisibles dans les modèles fédérés en plus de leurs tâches principales. Pour détecter ces attaques, on introduit ARMOR, un nouveau système de détection d'attaque basé sur GAN qui analyse les infor- mations intégrées dans les mises à jour du modèle. Le deuxième axe concerne la lutte contre les attaques d'inférence pour l'apprentissage fédéré préservant la vie privée, en particulier les attaques d'inférence d'appartenance. Pour renforcer la confidentialité en apprentissage fédéré, deux approches novatrices sont introduites : PASTEL, qui améliore la résilience des systèmes d'apprentissage fédéré contre les MIAs en minimisant la différence de généralisation interne, et DINAR, une méth- ode d'apprentissage fédéré préservant la confidentialité à grain fin qui obscurcit les couches sensibles à la

confidentialité et utilise une descente de gradient adaptative pour améliorer l'utilité du modèle. Ces objectifs de recherche visent collectivement à relever les défis en matière de sécurité et de confidentialité et à faire progresser le domaine de l'apprentissage fédéré.

MOTS-CLÉS :

Préservation de la vie privée, robustesse, apprentissage automatique distribué, apprentissage fédéré, attaques par empoisonnement, confidentialité, attaques d'inférence.

Laboratoire (s) de recherche : LIRIS

Directeur de thèse: Pr. Sara BOUCHENAK

Président de jury : Véronique Eglin

Composition du jury :

Romain Rouvoy Professeur, Université de Lille Rapporteur
François Taïani Professeur, Université de Rennes Rapporteur
Aurelien Bellet Chargé de Recherche, Université de Lille Examinateur
Véronique Eglin Professeur, INSA Lyon Examinatrice
Sara Bouchenak Professeur, INSA Lyon Directrice de thèse