University of Zurich[UZH]

BACHELOR THESIS – Communication Systems Group, Prof. Dr. Burkhard Stiller

Implementation of Membership Inference Attack Affecting Federated Learning-based Anomaly Detection System

Filip Trendafilov
Luzern, Emmenbrücke
Student ID: 16-107-567

Supervisor: Chao Feng, Dr. Alberto Huertas Celdran
Date of Submission: May 14, 2023

University of Zurich
Department of Informatics (IFI)
Binzmühlestrasse 14, CH-8050 Zürich, Switzerland

ifi

Bachelor Thesis
Communication Systems Group (CSG)
Department of Informatics (IFI)
University of Zurich
Binzmühlestrasse 14, CH-8050 Zürich, Switzerland
URL: http://www.csg.uzh.ch/

Abstract

In dieser Arbeit werden die Fähigkeiten von Federated Learning (FL) zur Wahrung des Datenschutzes untersucht, wobei der Schwerpunkt auf den konfigurationen Centralized Federated Learning (CFL) und Decentralized Federated Learning (DFL) liegt. Trotz der bestehenden Datenschutzvorteile haben sich sowohl CFL als auch DFL als anfällig für Angriffe durch Angreifer erwiesen, einschließlich Membership Inference Attacks (MIA). Diese Arbeit vergleicht die Datenschutz-Fähigkeiten von CFL und DFL, die auf MNIST, FashionMNIST und CIFAR-10 trainiert wurden, gegen White-Box und Black-Box MIA mit verschiedenen Leistungsmetriken. Darüber hinaus werden die am häufigsten verwendeten Abwehrtechniken gegen MIA erörtert, wie z. B. Differential Privacy (DP), Regularisierung und Wissensdestillation. Die Ergebnisse deuten darauf hin, dass FL-Modelle im Allgemeinen einen besseren Datenschutz bieten als ML-Modelle, wobei CFL die beste datenschutzfreundliche Föderation gegen Shadow Models ist, die binäre klassifikatorbasierte MIA verwenden, und DFL-Modelle mit einer vollständig verbundenen Netzwerktopologie eine starke Resistenz gegen MIA zeigen, die einen prognosebasierten Klassifikator verwenden. Diese Arbeit bietet wertvolle Einblicke in die Datenschutz-Fähigkeiten von CFL und DFL in verschiedenen Szenarien und unterstreicht die Bedeutung weiterer Forschung im Bereich des Datenschutzes bei kollaborativer ML.

This thesis investigates the data privacy preservation capabilities of Federated Learning (FL), specifically focusing on Centralized Federated Learning (CFL) and Decentralized Federated Learning (DFL) settings. Despite their existing data privacy advantages, both CFL and DFL have been shown to be vulnerable to adversarial attacks, including Membership Inference Attacks (MIA). This thesis compares the data privacy-preserving capabilities of CFL and DFL, trained on MNIST, FashionMNIST, and CIFAR-10, against White-Box and Black-Box MIA across various performance metrics. Furthermore, the most commonly used defense techniques used against MIA are discussed, such as Differential Privacy (DP), Regularization, and Knowledge Distillation. The findings suggest that FL models generally provide better data privacy than ML models, with CFL being the best data privacy preserving federation against shadow models using binary classifier-based MIA and DFL models with a fully connected network topology, showing strong resistance against MIA using a prediction-based classifier. This work offers valuable insights into the data privacy-preserving abilities of CFL and DFL in different scenarios and underlines the importance of further research in the domain of data privacy in collaborative ML.

Acknowledgments

The first acknowledgment I would like to speak out to, are my two supervisors Dr. Alberto Huertas Celdran and Chao Feng. With their constant support and deep understanding of the topic, they were able to guide me in the right direction and provide invaluable input at any time.
Second, I would like to thank Prof. Dr. Burkhard Stiller for giving me the opportunity to write my bachelor's thesis at the Communication Systems Group (CSG).
Third, I would like to thank and express my deepest appreciation to my family, and close friends. They have given me so much support during the last six months.

Contents

Abstract i

Acknowledgments iii

1 Introduction 1

1.1 Motivation . . . 2

1.2 Description of Work . . . 3

1.3 Thesis Outline . . . 4

2 Background 5

2.1 Machine Learning . . . 5

2.1.1 Supervised vs. Unsupervised Learning . . . 6

2.2 Centralized and Decentralized Federated Learning . . . 7

2.2.1 Open Source Framework for FL . . . 7

2.2.2 Network Topology . . . 8

2.2.3 Security and Privacy . . . 10

2.3 Membership Inference Attacks . . . 10

2.3.1 Adversarial Knowledge . . . 11

2.3.2 Classification Methods . . . 11

2.3.3 Inference Targets . . . 13

2.3.4 Inference Mode . . . 13

2.3.5 Evaluation Metrics . . . 14

2.4 Defense Techniques . . . 15

2.4.1 Differential Privacy 15
2.4.2 Regularization 15
2.4.3 Knowledge Distillation 16

3 Related Work **17**

3.1 Model Inversion 17
3.2 Property Inference 18
3.3 Reconstruction Attacks 19
3.3.1 GAN Attacks 19
3.3.2 DLG attacks 20
3.4 Source Inference Attack 20
3.5 Discussion 21

4 Scenario Setup **23**

4.1 Federated Learning Framework 23
4.1.1 Datasets 24
4.1.2 Training Architecture 27
4.1.3 Training Device Setup 27
4.1.4 Hyperparameters 27
4.2 Membership Inference Framework 29
4.2.1 Population Metric 29
4.2.2 Shadow Metric Inference Process 30
4.3 Discussion 31

5 Evaluation **33**

5.1 Model Training and Membership Inference Evaluation 33
5.2 ML vs CFL 34
5.2.1 MNIST 34
5.2.2 FashionMNIST 35

5.2.3 CIFAR-10 . . . 36
5.3 CFL vs DFL Fully Connected . . . 38
5.3.1 MNIST . . . 38
5.3.2 FashionMNIST . . . 39
5.3.3 CIFAR-10 . . . 41
5.4 DFL: Fully-Connected vs Ring . . . 42
5.4.1 MNIST . . . 42
5.4.2 FashionMNIST . . . 43
5.4.3 CIFAR-10 . . . 44
5.5 Conclusion . . . 45
5.5.1 MNIST . . . 45
5.5.2 FashionMNIST . . . 45
5.5.3 CIFAR-10 . . . 46

6 Summary, and Conclusions **47**
6.1 Future Work . . . 48

Bibliography **49**

Abbreviations **57**

List of Figures **57**

List of Tables **60**

A Installation Guidelines **63**

B Contents of the zip File **65**

Chapter 1

Introduction

Machine Learning (ML) has gained significant traction in the past decade. Especially, in recent years ML has been more relevant than ever. Industries like healthcare, finance, Internet of Things (IoT)[1], and marketing are taking advantage of ML-aided decision-making, improving automated systems, increasing efficiency in workflows[2], and analysis of user data [3]. The International Data Corporation (IDC) prognosis for the year 2025, states that there will be 41.6 billion IoT devices, generating over 79 zettabytes(ZB) of data [4]. These estimations are based on data stored in the private and public sectors on devices like personal computing devices (laptops, Personal Computers (PC), tablets, smartphones), and IoT devices. As the volume of data increases with the increase of participants, wanting to take advantage of the benefits ML can provide, and the increase of user data, so does the need for processing and learning from this information.

However, this also brings attention to the directly related topic of data privacy. As sensitive information is brought into ML training algorithms for the stated industries—Healthcare, health data, personal user information; Finance, transactions, and user addresses; IoT, personal user information; Marketing, social relationships, and geospatial data points—prioritizing privacy preservation, and responsible data practices is essential to safeguard individual privacy and build user trust.

Google first introduced the idea of Federated Learning (FL) to address the importance of data privacy preservation in ML[2] and ever since it has emerged as a critical approach within the domain of collaborative ML. This technique enables multiple participants to contribute to the development of a shared global model, it also preserves data privacy by eliminating the need for direct training data sharing.
As IoT devices usually suffer from high latency due to low network bandwidth, centralizing large volumes of data for training ML models resulted in unavoidable bottlenecks at the resources of network capacity and computational capacity of the devices.
FL also addresses the issue of bottlenecks with IoT devices by reducing constant raw training data flow to use the capacity of the individual devices for training [2]. Moreover, sharing only model parameters or gradients with a CS, or neighboring devices reduces the risk of bottlenecks at network throughput capacity. The study of FL settings has moved on quickly to further develop even more robust settings than the standard FL setting, where a CS takes the sole responsibility of aggregating model parameters to set a global

model. This setting is also referred to as Centralized Federated Learning (CFL). A FL setting with split responsibility of aggregating model parameters to all participants is called Decentralized Federated Learning (DFL).
As CFL already indicates with its name, its main feature is having a central server (CS) acting as an aggregator for the training devices, also referred to as participants or nodes. Participants train their model with their local training data until a certain threshold is reached, where they upload their results to the CS. The CS aggregates and combines the received data of the individual participants to then create a so-called global model. After aggregation, the global model is shared with each participant to start another round of training. CFL has been the main research topic in the field of FL and has therefore been studied in depth regarding its limitations and drawbacks. The first drawback addresses the issue of potentially running into high latency and federated network bottlenecks, as a higher number of participants, and the size of the participant's model parameters increases the data that has to be aggregated in the CS. Secondly, participants may not have the same network bandwidth, which hinders the network to synchronize properly and therefore slows down the federated network learning process. As the aggregation is handled solely in the CS, the third drawback introduces the problem of a bottleneck at the CS. The last drawback introduced by the work of Beltran et al. [3], highlights the risk of having a single point of attack regarding that in CFL the CS is handling the aggregation. They point out that sensitive information could be at risk when there is no guarantee to trust the CS.
DFL on the other hand splits the responsibility of aggregating the results to each participant instead of having a CS. This classification of FL came up in 2018 [5] to focus on the drawbacks of CFL. As participants train locally deriving model parameters, they finish a communication round by aggregating their model parameters with neighboring participants. Even though in both classifications, there is no raw data sharing between participants or between participants and CS, they still are susceptible to adversary attacks. Various work has shown that adversarial attacks, like Membership Inference Attacks (MIA)[6], [7] [8], [9], [10] have been proven successful in attacking FL models. MIA aim to infer whether a data record has been used to train the target model, which can become very harmful regarding FL has already found usage in domains like Finance, IoT, and Social Media. Taking advantage of the main drawbacks that FL presents, such as information leakage through model updates, and centralized points of vulnerability, ultimately compromising data privacy and security in collaborative ML environments.

These privacy vulnerabilities will be studied in this work, comparing the stability of CFL and DFL against MIA. This comparison will be conducted by comparing MIA metrics to elicit how serious the privacy leak in CFL and DFL are. Furthermore, the most used and prominently researched defense techniques [8], [11], [12] to guard FL systems against MIA will be presented. The following chapter outlines the motivation for this comparison, sets the scope, and will introduce the structure of this thesis.

1.1 Motivation

DFL has not been studied as rigorously as CFL but presents very similar vulnerabilities, sharing model updates between neighboring nodes, and possibly having a malicious neigh-

bor that could be eavesdropping, or manipulating the training process actively. These threat models can also be classified as active attacks, for manipulation of the training process, and passive attacks, for eavesdropping on neighboring participants or CS.

Comparing the privacy leakage of CFL and DFL conducted via MIA is a novel contribution to the field of MIA in the domain of FL. As there is already existing work that has analyzed MIA on centralized FL, the analysis for DFL has been sparse. Thus, to improve the privacy-preserving capabilities of FL systems, this thesis will measure the performance of MIA affecting both FL settings. Furthermore, the performance of MIA will be measured on three different network topologies applied to the DFL participant configuration.

1.2 Description of Work

To address the motivation of this thesis, it is important to set a clear set of scenarios. That way a structured comparison of MIA metrics can be conducted and may provide further insight on how to defend against them.
As a first step though, an in-depth analysis of the Fedstellar [13] Framework will be made, which was used to train CFL and DFL models for the experiments. Describing the architecture of the model training, using Convolutional Neural Networks (CNN), and the aggregation algorithm FedAvg, the aggregation algorithm used in FL.
Secondly, the fundamentals regarding MIA in FL settings will be set by documenting the research phase of this thesis. For a clear overview of the expanding topic, a taxonomy table is provided and each category will be explained. The performance measure of MIA and the MIA itself will be conducted using the open-source framework Privacy Meter, proposed by Nasr et al.[6], Ye et al. [14], Kumar et al. [15], and Shokri et al.[16]. This way, the result metrics from this thesis can be compared with the experiment result metrics from their work.
The MIA metrics used for the conclusion will consist of comparing the Area Under the Reciever Operator Characteristics Curve (AUC), True Positive (TP) Rate, False Positive (FP) Rate, True Negative (TN), False Negative (FN) Rate, and the Accuracy-, also referred to as Precision Rate (PR).

Thirdly, the aforementioned scenarios have to be set. The models will be trained on four different datasets: MNIST, FashionMNIST, EMNIST (letter split), and CIFAR10. All models are trained using CNN, for the MNIST-based datasets, two layers, and for CIFAR10, three layers were used. For each dataset, four models are trained. One CFL model and three DFL models, which differ in their network topology: Fully-Connected, Ring, and Star.
With these splits, four scenarios for each dataset are created, which will be referred to with the dataset prefix of M for MNIST, F for FashionMnist, E for EMNIST and C for CIFAR10. The scenario name is followed by a dash and the chosen federation: -C, for CFL; -D, for DFL. Furthermore, the DFL setting will have another suffix indicating the network topology: -F, for Fully-Connected; -R, for Ring; and -S, for Star. All 16 scenarios will be evaluated on their privacy-preserving stability, using the Population Metric and Shadow Metric, implemented in the Privacy Meter framework.

1.3 Thesis Outline

The thesis is structured in a way, that builds up all the required knowledge that is needed for the experiments. Chapter 2 will focus on building up the fundamentals of Deep Learning and the differences between CFL and DFL.
The second part of Chapter 2 will cover all the information needed to understand how and why MIA are finding success in breaching FL models. This includes covering the taxonomy of MIA, showing the differences between White-Box and Black-Box, local and global, passive and active attacks, and the two classification approaches: binary classifier, and prediction classifier. To conclude the second part of Chapter 2, an in-depth explanation of the MIA evaluation metrics will draw the transition to the last part. This last part will discuss defense techniques used to protect FL systems from MIA.
Chapter 3 will cover the related work section where innovative and extended forms of MIA from recent works will be presented. Inference techniques that target not only to expose whether a datapoint has been used for training, but also show how to infer the participant that was trained on that datapoint, "Source Inference Attacks" proposed by Hu et al.[10].
Chapter 4 will discuss the details of the scenario setup, covering model training configurations, hyper-parameters, and exact neural network convolution parameters in the first part. In the second part of Chapter 4, the inference attacks used in the experiments: Population Metric, and Shadow Models, will be presented. This also includes going over each step of how the experiments were carried out.
Chapter 5 will compare the results computed from the experiments stated in Chapter 4 and put those evaluations into perspective with the work of the developer of Privacy Meter.
Lastly, Chapter 6 will have a summary of this thesis, draw the conclusion of the experiments, discuss the limitations, and give an outlook into what future directions this research area might be taken.

Chapter 2

Background

The following chapter will introduce the main concepts of FL and MIA. First, there is a brief summary of what FL is and the importance of data privacy in the context of standard ML versus collaborative ML. Secondly, a definition of MIA is provided, followed by the history and development of MIA. The main section 2.2 is centered around building up a taxonomy of MIA to categorize them by their adversarial knowledge, classification methods, inference targets, inference modes, and domains.
To conclude the background chapter, section 2.3 lists the most common defense techniques against MIA.

2.1 Machine Learning

In standard ML the training of, for example, a classification model is carried out by a single participant. Leaving out the need to worry about sharing data with other devices. This means that the efficiency and accuracy solely depend on the computational power and training data of that single training device [17]. As already established, this poses major difficulties when the classification task requires training models on devices with limited computational or network resources like IoT devices. Furthermore, Shokri et al.[16] have shown that even in a limited setting, where the adversary only gets black-box queries that return the model's output on a given input, they predict the membership of a given data record with alarming accuracy.
For training the ML model [16], famous Machine-Learning-as-a-Service platforms were used, including Google Prediction API[18], Amazon Machine Learning [19], Microsoft Azure Machine Learning [20], and BigML [21]. The models could not be downloaded and the training algorithms are hidden from the users, due to the access of the model being limited to only be accessible through the provider's Application Programming Interface (API). As documented in the work of Shokri et al. [16] the model used for training was chosen adaptively by the ML-as-a-Service providers and there was no control over the configuration of regularization techniques, at least for the Google Prediction API. This also leaves out the possibility for the user to influence the training process, in case

the model overfits, and neither does the user get warned regarding the consequences of overfitted models.

What are the consequences of overfitted models, and when is a model overfit? To discuss this matter, it is important to understand the process and technique on how ML models transform labeled input data into scores that enable the classification of the desired task. Also, the following explanations assume the supervised learning setting, the difference between supervised and unsupervised learning will be briefly covered in the upcoming Subchapter 2.1.1.
As the model gets fed labeled training data, each data point is assigned a score, also referred to as a loss. This loss reflects the model's prediction error and therefore aims to minimize this loss. SGD [22] and Adam[23] are well-known algorithms that are used in ML to optimize this objective function. While the loss minimizes on the training data, hence starts to learn the labels of the training data, the goal is to be able to also label unseen data [10], [24]. When the model is called overfitting, this just means that the model has difficulties classifying data that was not used in the training process [25], which is also represented in the output of the loss and the accuracy difference between training data and test data.
This difference in output behavior between training data and unseen test data of the model is one reason why MIA have found success [1], [9], [14], [16], [26]. In the work of Shokri et al. [16], using the Google Prediction API, they trained a model on the Pruchase-100 dataset, with a training accuracy of 99% and a test accuracy of 66%. The accuracy of the MIA got up to over 93%. Although the reason for the alarming MIA accuracy is not only due to the model overfitting on the training data, having a model that centralizes the whole training data onto one participant, greatly impacts the privacy security of possibly sensitive data.

2.1.1 Supervised vs. Unsupervised Learning

In this thesis, only supervised Learning has been considered and used for training the target models. Though it is important to know the differences between the two categories since the adversary attacks would need to be adjusted accordingly.
The difference in provided data and learning task distinguishes the two learning categories. First, supervised learning will be covered and then unsupervised. In supervised ML, the model's goal is to learn a general rule for mapping input data to the output, given the input data is already correctly labeled [27]. Using the difference between the true label and the predicted label [28], the model is able to learn over time, also referred to as epochs. This kind of technique is used in feedforward models for classification tasks. MLP and CNN architectures are two examples of such feedforward models, CNN will be the architecture of choice for this thesis and will be briefly discussed in Chapter 4.
Unsupervised learning on the other hand uses unlabeled data to discover hidden patterns. It is called unsupervised because it does not need human "supervision" to evaluate the proposed solution. This technique is used in the following problems: Clustering, Association, and Dimensionality reduction [28].

2.2 Centralized and Decentralized Federated Learning

To address the privacy issues of centralized ML, FL emerged as a promising new technique to preserve privacy by decentralizing training data across multiple participants[6]. CFL focuses on having a CS that takes on the responsibility to aggregate the participants computed model parameters, to create a global model. DFL on the other hand decentralizes this responsibility of the CS to all participants and therefore eliminates one of the drawbacks of CFL, where the CS represents a single point of failure [29]. There is another categorization of FL which is referred to as Semi-Decentralized Federated Learning (SDFL), which will not be part of the experiments, but for the sake of completeness still be mentioned. In SDFL, the responsibility of aggregation is passed on periodically during training. The order in which the participant is chosen next is either not defined, so at random, or it's based on the performance of either the network capacity, computational power, or general power capacity[30], [31].

2.2.1 Open Source Framework for FL

Opensource frameworks like TFF, PySyft, SecureBoost, and FedML offer both CFL and DFL federations. TFF is developed by Google and is based on TensorFlow. This framework offers three parts: models, a wrapping function to use already existing models; "Federated Computation Builder", functions that help with evaluations and training; Datasets, a collection of data that can be downloaded and accessed via Python [32]. The major drawback of TFF is that it does not include any privacy mechanisms.
PySyft provides an interface that helps developer implement their training algorithm. It also supports both TensorFlow and PyTorch and even includes privacy mechanisms such as differential privacy and secure multiparty computation [33].
SecureBoost is designed to comply with General Data Protection Regulations (GDPR) in the European Union of May 2018. To achieve this goal the framework designed an encryption strategy that ensures secure node communication. Additionally, SecureBoost models are as accurate as "non-federated gradient tree-boosting algorithms that require centralized data" [34].
FedML focuses on having a broad usage regarding hardware requirements. It supports three computing paradigms namely, on-device training, single-device simulations, and distributed computing. Furthermore, it promotes the idea of having diverse algorithms and comprehensive reference implementations for the provided datasets, optimizers, and models [35].

All the stated open-source Frameworks use the well-known aggregation algorithm FedAvg. It is a heuristic-based algorithm used for its coherent and empirical effectiveness. The algorithm expects model parameters from neighboring participants and then aggregates the average of the received updates to then compute a new global model [36]. While participants train during multiple epochs on their local dataset, they gather information and minimize losses. After the participants reach a predefined epoch counter, they upload the information to the aggregator. In CFL this would correspond to the CS, whereas in DFL this is done with connected neighboring nodes.

In CFL, FedAvg is the most used aggregation algorithm, and the above-stated open-source FL frameworks have also been using FedProx in TFF, FedOpt, and FedNova in FedML, and GBDT in SecureBoost [3].
It is important to explore different aggregation algorithms and find ways to improve and optimize this step. Especially for DFL federations, this can become a great boost in efficiency, as the number of aggregations increases with the number of participants training the same model. Furthermore, depending on the network topology used in DFL, the communication costs vary.

2.2.2 Network Topology

In this thesis, two different network topologies were used for the DFL federation: Fully-Connected, see Figure 2.3, and Ring, see Figure 2.2. In the Fully Connected network topology, all participants are connected to each other. This results in a very high communication cost between participants. To put it in mathematical terms, the communication costs increase with the square of the number of participants in the network. Therefore the Fully Connected topology is not suited for cases where scalability is an important key factor. The advantages of Fully Connected network topologies are drawn through their robustness and flexibility, as the failure of one or more participants does not kill the entire communication pipeline because the other participants are interconnected and can still carry on[37].
The Star network topology was only used in CFL, see Figure 2.1, because for DFL federations usually a more complicated topology is chosen. Although this topology is the main usage for CFL, it can also be applied to DFL. Star- and Ring network topology belong to the family of Partially Connected Networks. This family of networks does not provide the same robustness as the Fully-Connected network topology, more specifically the Star topology can be a single point of failure for the whole federation. As each participant is connected to the middle participant, the communication cost grows linearly with the number of participants in the network. Scaleability is therefore limited to the local participant resources [3]. Furthermore, communication between non-center nodes all have to pass through the central node, causing bottlenecks at the central node and possibly straggling the whole communication process of the network[38].
For the last topology used in this thesis, the Ring network topology, the communication cost increases also linearly with the number of participants in the network [3]. Participants are connected to a maximum of two other participants, their neighbors.
For both Ring- and Star network topology, the main drawback occurs in their possibly high communication cost, as an update of one participant has to pass through at least one, and at most one less than the number of participants in the entire network to reach all participants.

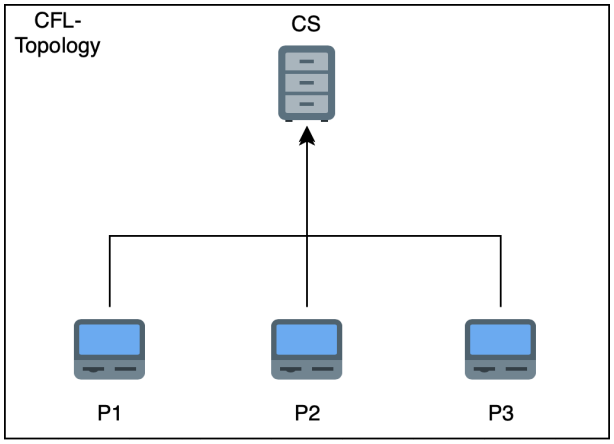


Figure 2.1: CFL-Star Topology

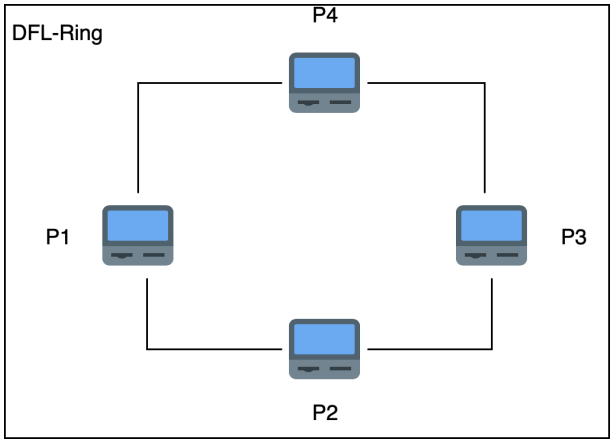


Figure 2.2: DFL: Ring

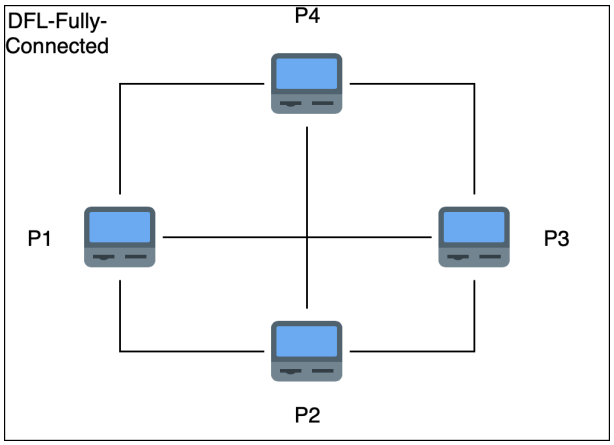


Figure 2.3: DFL: Fully-Connected

2.2.3 Security and Privacy

Although FL originated from the idea to make collaborative ML more secure, analysis from recent works has shown that the preservation of privacy is still at risk [3], [6], [7], [17], [29], [39]. Even though raw data is never directly exposed during training, the exchange of model parameters and the comparison of model losses makes FL models prone to leak sensitive information.
DFL presents an even more prone setting for privacy leakage than CFL, as the nature of DFL is defined to share model parameters more frequently than CFL [3].
Therefore it is important to keep on exploring new methods regarding aggregation algorithms, which directly correlate with the issue of exposing sensitive information during communication. As more algorithms emerge, naturally it would become more difficult for attackers to design tailored attacks.

2.3 Membership Inference Attacks

Gradients are derived from the model parameters, which have their origin from the participant's local data, these gradients are then processed in the layers of deep neural networks. The input of the following layer takes the feature and the error from the previous layer and further refines the output to minimize the error [40]. This small window of exposing only an abstraction of the raw training data can be tracked and used to infer whether a given data point was used in the training process or not [16]. A real-world example can be constituted to show the possible grave impact of an adversary attack like MIA.
Assuming all hospitals of central Switzerland are using an FL-aided system that helps Doctors diagnose lung cancer. Assumption 2: This FL system has been trained on X-ray pictures with the goal to be able to detect lung cancer in an early stage. Making this task an image classification task. Furthermore, for the system to be able to analyze multiple factors like blood levels it might need more than just X-ray images for accurate classifications.
Now coming from an attacker's point of view, knowing that hospitals in central Switzerland have been working together to improve the efficiency of lung cancer diagnosis, and the knowledge of what it takes to diagnose lung cancer, inferring membership of sensitive patient data could have devastating consequences.

Investigating the potential and limitations of existing MIA systems designed to compromise the privacy of FL, particularly in the context of DFL federations, is crucial for understanding and mitigating their consequences.
Inference attacks are classified into two categories, as elaborated in the work of Dwork et al. [41]: tracing attacks, which in this thesis will be referred to as MIA [16], and reconstruction attacks. Reconstruction attacks aim to infer attributes of records from the training data [42], [43]. As the experiments were conducted using MIA, this chapter will be dedicated to creating a taxonomy of MIA. This will help oversee the broad field of already investigated and thoroughly researched areas of MIA.

2.3.1 Adversarial Knowledge

As MIA belongs to the family of adversarial attacks, the first thing that needs to be differentiated is what the adversary assumes to be able to observe, also referred to as the adversarial knowledge [12], [40]. In the context of MIA on ML models, attackers can leverage two types of knowledge: training data and target model knowledge.
Training data knowledge corresponds to the distribution of the data used for training. Generally, it's assumed that an attacker has access to a shadow dataset, which follows the same distribution as the original training data. This is plausible, as the shadow dataset can be obtained through statistics-based or model-based synthesis methods [16]. For a more sophisticated and real-world resembling attack, it is common to assume that the shadow dataset and the training dataset are disjoint.
Target model knowledge involves understanding the learning algorithm, model architecture, and learned parameters. Using these types of adversarial knowledge, one can assess the threat level posed by different attacks [12]. The differentiation for adversarial knowledge is defined as black-box attack, for the most constrained scenario; white-box attack, for scenarios with complete access to the target model and target dataset; and grey-box attack for scenarios with limited access to the target model and target dataset.

In a white-box attack scenario, an attacker can access comprehensive information about the target model. This includes the distribution of training data, the model's training process, its architecture, and learned parameters. This information allows the attacker to perform a tailored and effective privacy breach on the target model[12], [40].

In a black-box attack scenario, the attacker's knowledge is limited to the training data distribution and the ability to query the target model. They do not have direct access to the model's architecture or learned parameters. For instance, when attacking a classification model, the attacker can only query the target classifier and receive prediction outputs for the given input records [12], [14].

In a grey-box attack scenario, the attacker has partial knowledge of the target model. The attacker has access to the training data distribution and some information about the target model, such as its architecture or certain aspects of the training process. However, the attacker may not have full access to the model's learned parameters or specific training details. This intermediate level of knowledge allows the attacker to conduct more informed attacks compared to black-box scenarios but with less effectiveness than white-box attacks [44].

2.3.2 Classification Methods

Deep neural networks are often overparameterized and therefore can memorize information about their training datasets due to repeated training on the same training data [15], [45], [46]. As a result, these models output distinct behaviors for training data records (members) and test data records (non-members), with the model's parameters storing statistically correlated information about specific training data records [6], [15], [16]. This difference in behavior enables the tailored design of attack models to distinguish between

members and non-members of the training dataset. The work of Hu et al. [12] distinguishes between two primary MIA approaches: binary classifier-based attack approaches and metric-based attack approaches.

Binary classifier-based MIA is a classifier that differentiates between a target model's training members and non-members by observing the behavior of the target model. One effective technique for training such a classifier is shadow training, introduced by Shokri et al. [16]. In this approach, an attacker creates multiple so-called shadow models that replicate the target model's behavior, assuming that the attacker knows the structure and learning algorithm.
By utilizing the information of knowing the training and test datasets, the attacker constructs a shadow dataset with "features and the ground truth of membership of the training and test data records" [12]. This shadow dataset is then used to train the binary classifier-based shadow model.
Shadow training is applicable for both white-box and black-box attack models, with the training process being the same for both adversarial knowledge levels. Although the training process does not differ, depending on the adversarial knowledge level, the attacker can use more information for the setting of white-box knowledge, and very limited information for the setting of black-box knowledge. In black-box attacks, the attacker only has access to the target model's prediction vector (output) for a given input data record. When querying shadow models, they collect only the prediction vectors of each data record.
In contrast, white-box attacks grant the attacker full access to the target model, allowing them to observe intermediate computations at hidden layers as well as prediction vectors. Consequently, white-box attackers gather more information to build their attack model compared to their black-box counterparts [12].

Metric-based MIAs on the other hand, offer a simpler, less computationally intensive alternative to binary classifier-based MIA. Instead of training a binary classifier, metric-based MIA use calculated metrics on prediction vectors to then compare these metrics to a preset threshold. The comparison of the computed metrics and the threshold defines the membership of a given data record. In the work of Hu et al. [12], they categorize the following four primary types of metric-based MIAs: prediction correctness-based [25], prediction loss-based [25], prediction confidence-based [47], and prediction entropy-based attacks [26], [47].
Prediction correctness-based MIA, infer membership solely on the performance of the target model. So if the target model predicts an input data record as a member, so does the attacker, and vice versa for non-member predictions [25].
Prediction loss-based MIA rely on the loss difference between the prediction loss, given an input data record, and the average loss of all training members of the target model. This method assumes that the average loss of all training data is smaller than the loss of the test data, which is a fair assumption since the target model tries to minimize the prediction loss of the training data [25].
Prediction confidence-based MIA compare the maximum prediction confidence with a preset threshold, if the maximum prediction confidence is larger than the threshold, the training data record is marked as a member, and non-member otherwise [47].
Prediction entropy-based MIA compare the prediction entropy with a preset threshold to infer membership of a training data record. If the entropy is larger than the threshold, the data record is marked as a member, and non-member otherwise [47].

2.3.3 Inference Targets

In the setting of FL there are multiple approaches on how the adversary can be placed to observe the communication of model parameters between participants and the aggregator. An attacker can either be the CS or one of the participants. As a curious CS, the attacker receives updates from each participant and can therefore infer information about their training data or even manipulate the global model actively. Alternatively, a curious participant can only observe global parameters when the CS sends the updated global model to the participants. This way the curious participant can track the differences between its local parameters and the new global model sent by the CS. In both positions, the attacker is observing the learning process over multiple communication rounds [6].

2.3.4 Inference Mode

MIA can further be categorized into active and passive attacks, though in most MIA applications a passive approach is chosen. A passive inference mode is depicted as an adversary (CS or participant) that observes the learning process without modifying it. Also, the time of the attack in the passive attack mode happens after the training is done. However, active attacks involve an adversary participating during the training process to extract more information about the target model's training set. The inference model architecture remains the same for passive and active attacks.
An active attacker can exploit optimization algorithms, like Adam or SGD, by observing how the optimizer minimizes losses of training data. For a training data record with a big loss, the optimizer will react by reducing this loss. However, if a training data record was not checked during training, the loss change is rather moderate. This difference in the behavior of the loss can be exploited to improve MIA performance [6].
These performances can be captured and analyzed by computing four main metrics, which will be discussed in the last sub-chapter of 2.3.

2.3.5 Evaluation Metrics

To complete Chapter 2.3, it is necessary to discuss the evaluation metrics most used to assess the performance of MIA. The four main metrics that will be used in the experiments are: True Negatives, False Negatives, True Positives, and False Positives. "Positives" refers to the classification of a data record as a member, and "Negative" refers to the classification of a data record being a non-member.
Following, there will be a list of advanced metrics that can be calculated with the just mentioned basic metrics:

1. **Attack Success Rate** (ASR) [16]:

$$\frac{\text{True Positives} + \text{True Negatives}}{\text{True Negatives} + \text{True Positives} + \text{False Negatives} + \text{False Positives}}$$

2. **Attack Precision** (AP) [16]:

$$\frac{\text{True Positives}}{\text{True Positive} + \text{False Positive}}$$

3. **True Positive Rate** (TPR) [16]:

$$\frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

4. **False Positive Rate** (FPR) [48]:

$$\frac{\text{False Positives}}{\text{False Positives} + \text{True Negatives}}$$

5. **Membership Advantage** (MA) [25]:

$$\text{TPR} - \text{FPR}$$

6. **F1-Score** (F1) [49]:

$$\frac{2 \cdot \text{AP} \cdot \text{TPR}}{\text{AP} + \text{TPR}}$$

7. **AUC-ROC** (Area under ROC Curve) [50]: The ROC curve is a plot that lets the TPR go against the FPR at various threshold values. The Area under this curve represents a performance measure of the MIA, where a value of 0.5 corresponds to the classification being as good as a random guess.

2.4 Defense Techniques

This Chapter will cover the main defense techniques used to protect ML systems from MIA. In this thesis, the defense techniques will only be briefly covered theoretically, as none of them were used in the experiments.

2.4.1 Differential Privacy

Differential privacy (DP) [51] is a probabilistically based privacy-preserving technique that offers a privacy guarantee. Many studies have applied DP to ML models to reduce the impact of MIA [3], [6], [9], [14], [16], [17], [25], [26], [29], [52]. When DP is applied to ML models, it naturally limits the success probability of MIA, as the "models do not learn or remember specific user details when the privacy budget ϵ is sufficiently small" [12]. The mechanism behind DP is to make sure that the probability of an outcome should be very similar for any two input datasets, given that they differ in only one data point [53].

DP-infused ML models have shown that the membership advantage (MA) of an attacker is constrained by the threshold function ϵ [25]. DP is generally applied by adding noise to the gradients during training, but it may significantly hinder model performance in certain situations [16]. In trying to ensure the privacy of ML models, the amount of noise that has to be added during training increases, resulting in poor model performance. Therefore it is important to find a balance and set a useful ϵ to ensure that the model performance does not suffer too much while trying to preserve the privacy guarantee. A major advantage of DP pictures the protection it provides while the adversarial knowledge mode does not matter. So whether the attacker has full knowledge (white-box) or is only limited to a black-box setting, DP is effective for both scenarios [12]. Additionally, DP can be used to defend against other privacy attacks like attribute inference attacks [54] and property inference attacks [55].

2.4.2 Regularization

Regularization techniques are implemented to decrease the overfitting of ML models, which can help protect against MIA. Several regularization methods, like L2-norm regularization [14], data augmentation [56], dropout [57], and label smoothing [58] have been proposed to enhance the generalization capabilities of ML models. These techniques were initially designed to decrease overfitting, but have been proven effective in reducing MIA effectiveness, as they help the model to generalize to test data and therefore reduce the difference in the model's output behavior between training and test data [52].
Regularization methods are effective against MIAs in both black-box and white-box settings because they modify the target model's output distribution and its internal parameters. Despite their effectiveness and broad applicability, regularization methods have a potential drawback: they may not provide a beneficial tradeoff between membership privacy and prediction utility [16]. For example, L2-norm regularization can decrease MIA

accuracy to random guesses score of 0.5 when using large regularization factors, but this may lead to a considerable decrease in the target model's prediction accuracy.

2.4.3 Knowledge Distillation

Knowledge distillation [59] is a technique that transfers knowledge from a fully trained model to a smaller model, allowing the smaller model to achieve similar accuracy. This defense technique is a form of the model compression technique. The process of transferring knowledge from a fully trained model to a smaller model happens periodically and saves computational resources. For FL settings this technique can be very beneficial regarding the privacy of training data, by sharing knowledge rather than model parameters during communication between participants [17].

Chapter 3

Related Work

This Chapter will elicit the current research status of FL data privacy and discuss the security threats posed by Inference Attack-related attack methods.

3.1 Model Inversion

In FL models are continuously updated via queries and model revisions from various devices. Should an attacker gain access to these queries, they can reconstruct the model data. This can give the adversary the information needed to understand both the training data and the model [54], [60]–[63]. White-Box model inversion attacks, reconstruct training data from model parameters and can be classified into two types: Deep Leakage from Gradients attacks [64] and Gradient Similarity (GS) attacks [65].
DLG attacks operate under the assumption that the server is malicious and aims to reconstruct a participant's data using their uploaded gradient. The server optimizes the reconstructed data to minimize the Euclidean distance between the original gradients and those generated during the backpropagation of the reconstructed data [66]. Conversely, GS attacks function similarly to DLG attacks, but instead of using Euclidean distance, they employ cosine similarity between the original gradients and the reconstructed gradients. This is utilized to augment the reconstructed data through local updates.
Other works [61] introduced an attack strategy that compromises inference data privacy in collaborative DL. This technique allows a malicious participant to recover any arbitrary input data without the need to have knowledge about other participants in the same network or their model update information. To validate this, a two-device system, with one trusted and one untrusted component, was considered. This validation setup was only used to prove the possible novel attack strategy, but the implemented model is actually applicable to multi-device scenarios. In such a scaled multi-device scenario, all adversarial participants are regarded as untrusted, while all the initial and all intermediate ones are regarded as trusted. To make this attack strategy focus on the objective, the adversary is assumed to follow all inference protocols put on from the system, such that it stays hidden. Additionally, the adversary does not get involved with the trusted devices' inference game or observe any intermediate computations.

The authors explored the model's application in three distinct adversarial knowledge scenarios: White-Box, Black-Box, and "query-free". In a 'white-box' setting, the adversary has the knowledge of the inner structures of the DNN layers, which includes the model architecture and model parameters, similar to a White-Box scenario in MIA. This information is used to comprise input data without any information about the training dataset, unlike the parallel scenario in MIA, where in a White-Box scenario, knowledge about the dataset is expected.
In a Black-Box scenario, the attacker indirectly learns about the model through input-output comparison, and while the adversary may learn about the intermediate computations or the distribution of the training data, it's not necessary for recovering sensitive data.
The third scenario is a unique 'query-free' black-box setting, where the adversary does not even require to compare the input-output values to reconstruct the target model's data. These findings indicate that the model inversion attack can recover data with high accuracy[67], even when the adversary has no knowledge of the model or is even completely restricted and not even allowed to query inputs. This suggests that the requirements for successful data reconstruction are minimal and therefore pose a serious threat model for FL.

3.2 Property Inference

In FL environments, the main objective is to train a classifier that can accurately predict a specific task by learning patterns from the dataset at hand. Occasionally, the model may learn properties not directly related to the main task, which can be exploited through property inference attacks, leading to significant privacy vulnerabilities for sensitive training data [29], [40], [55], [66], [68], [69].

Property inference focuses on inferring properties of the target models' dataset, independent of a subset of training inputs, rather than the entire class. This is particularly relevant in FL, where the data's relevance determines each individual model's contribution to the global model. A particular concern is single-batch property inference, which aims to disclose the unique properties of a data batch [68]. This can lead to serious privacy vulnerabilities, especially when a property is also incorporated in the training data. To effectively execute property inference attacks, an attacker needs supplementary training data, labeled with the specific property that aims to be inferred. Property inference attacks can be categorized by the inference mode, namely, active and passive, similar to MIA [12].

Active property inference attacks can execute more powerful attacks using multi-task learning. The adversary advances the local model using an augmented property classifier, which is connected to the last layer, and trained to perform the classification and recognize batch properties. During collaborative training, the adversary uploads updates based on the consolidated loss, causing the shared model to learn separately for data with and without specific properties. This allows gradients to be distinguishable, enabling the attacker to classify whether the data has certain properties. These adversaries, although more active, are considered honest-but-curious. As in the model inversion example [61],

for the sake of theoretical analysis, the attacker is assumed to adhere to system protocols and not overstimulate the system with messages, that would expose the attacking system [29].

In passive property inference attacks, on the other hand, the attacker only observes the participant's updates and is able to infer without the need for the local or collaborative training process to be influenced. As in the active attack mode, it is assumed that the attacker has supplementary data, containing the target property and points, that are lacking such properties. This data should come from the target participant's class, making it unbound to other classes [68]. The attack name comes from the nature of the attack, capitalizing on global model updates to generate compound updates based on data with specific properties and data without. This enables the attacker to train a binary batch classifier to identify whether updates belong to the data with the property in question. The key difference between active and passive inference attacks is the ability of an active attacker to compute more local computations and share these with the aggregator.

3.3 Reconstruction Attacks

Privacy concerns in FL exist beyond data leakage, as model updates from gradient client updates, Support Vector Machines (SVM), or k-Nearest-Neighbor (KNN), where values are saved, can also be compromised. These methods facilitate reconstruction attacks aimed to recovering the training data or vectors used during model training [70]. In FL settings, even the gradient update can reveal sensitive client information, making the trained models vulnerable to reconstruction attacks. Reconstruction attacks can be categorized into two families: GAN-based attacks, and deep leakage from gradient (DLG) [64] attacks. Both types of attacks utilize the gradient update information leakage for reconstruction, necessitating specific defensive measures like multi-party computation (MPC)[71] and homomorphic encryption (HE) [72], [73]. Unlike MIA these specific defensive measures are needed to effectively inhibit reconstruction attacks.

3.3.1 GAN Attacks

Generative Adversarial Networks(GAN)-based techniques serve as a significant source of data reconstruction attacks in FL settings [7], [9], [74]. These attacks leverage the operational dynamics of the learning process to develop a GAN capable of creating a model sample, which is similar to the so thought, secure private model. The process involves using a GAN discriminator to reconstruct data for a specific class and differentiate the data's source. This strategy drives the model to identify differences between the datasets, thus improving GAN's performance. Existing defense techniques such as DP fall short against such attacks [75], leading to suggestions of using more complex measures like MPC [71] or HE[72].

Another GAN-based approach involves reconstructing the local data using model parameters o the target model[76]. The method uses both the generator and the discriminator

networks of the GAN to launch the attacks. The objective is to produce similar images and data distributions between the two networks. The generator uses the latest global model parameters to generate false data, while the discriminator identifies differences between the generated and original target data. This iterative process minimizes prediction loss, resulting in an efficient model.

Finally, a user-level GAN-based reconstruction method called 'mGAN-AI' [77] is proposed for FL settings, capable of reconstructing real training data and targeting a specific device to break user-level privacy. This method uses a multitask discriminator capable of distinguishing the category, reality, and client identity of input data. The proposed model, which does not interfere with the main training process, has been experimentally proven to deliver stronger privacy breaches compared to general reconstruction.

3.3.2 DLG attacks

The primary focus of DLG is the generation of artificial data and labels using a temporary gradient that tries to match with the gradients that are being shared between participants and aggregators. This technique has been prominently utilized to comprise data leakage attacks in FL settings [66].

The DLG technique has been explored to test the feasibility of acquiring private training data from publicly shared gradients [78]. The approach involves initializing synthetic data and labels, calculating gradients that are close to the real ones on the currently shared model, and reducing the difference between the artificial and real gradients by updating the synthetic data and labels iteratively. Despite its effectiveness, DLG struggles with convergence and repetitive correct label classification [66].

An improved model that addresses the problem of label data leakage during gradient sharing was further explored, named Improved DLG (iDLG) [79]. The authors state that the model is able to comprise the real labels with high accuracy. It works well for a differentiable model trained with a cross-entropy loss function over labeled data if gradients are given for each data point in the training data. However, its ability to identify ground-truth labels is strictly tied to this condition.

3.4 Source Inference Attack

Source Inference Attack (SIF) [12] introduces a novel inference attack in the context of FL. The SIA aims to determine which client owns a training record in FL, extending beyond MIA to identify the source of the data, exploiting devices that are trained on the explored data.

The authors argue that the SIA can be conducted by an honest-but-curious CS, who knows the identities of clients and receives updates from them. The server can infer client-private information without interfering with the FL training or affecting the model prediction performance.

The paper explores the SIA from the Bayesian perspective, demonstrating that a server can achieve the optimal estimate of the source of a training member in an SIA without violating the FL protocol. The prediction loss of local models on the training members is utilized to obtain the source information of the training members effectively and non-intrusively.

The authors conducted empirical evaluations of the SIA in FL trained with one synthetic and five real-world datasets, considering several FL aspects such as data distributions across clients, the number of clients, and the number of local epochs. The experiment results validate the efficacy of the proposed source inference attack under various FL settings. An important finding is that the success of an SIA is directly relevant to the generalizability of local models and the diversity of the local data.

The authors' main contributions include proposing the SIA, a novel inference attack in FL that identifies the source of a training member, adopting the Bayesian perspective to demonstrate that an honest-but-curious central server can launch an effective SIA in a non-intrusive manner, and performing an extensive empirical evaluation on both synthetic and real-world datasets under various FL settings.

3.5 Discussion

Table 3.1 summarizes the related Inference Attacks. It puts all the discussed attacks into a comparison of the described effectiveness, shows countermeasures that were presented and are sorted according to the inference family.

To conclude the related work section and the discussed inference attacks, it's fair to say that the field of FL is very dynamic, since its origin back in 2016 [2]. Inference methods appearing from 2022 on seem to be extending existing approaches and therefore pose even more serious threats. As SIA being an extension of the original MIA on FL models or the novel model inversion attack [67].

Table 3.1: Comparison of different Inference Attack Families

Work Citation	Year	Defense Technique	Effectiveness	Inference Attack Family
[64]	2021	-	Yes	Model Inversion (DLG)
[65]	2020	-	Yes	Model Inversion (GS)
[61]	2019	-	Yes	Model Inversion (WB, BB)
[67]	2023	-	Yes	Model Inversion
[69]	2021	-	Yes	Property Inference
[61]	2019	-	Yes	Property Inference (Active)
[29]	2020	-	Yes	Property Inference (Active)
[68]	2021	-	Yes	Property Inference (Passive)
[64]	2021	[71]–[73]	No	Reconstruction Attack (DLG)
[70]	2021	[71]–[73]	Yes	Reconstruction Attack
[9]	2020	[71]–[73]	Yes	Reconstruction Attack (GAN)
[74]	2018	[71]–[73]	Yes	Reconstruction Attack (GAN)
[7]	2020	[71]–[73]	Yes	Reconstruction Attack (GAN)
[76]	2021	[71]–[73]	Yes	Reconstruction Attack (GAN)
[77]	2019	[71]–[73]	Yes	Reconstruction Attack (GAN)
[78]	2019	[71]–[73]	No	Reconstruction Attack (DLG)
[79]	2020	[71]–[73]	No	Reconstruction Attack (DLG)
[12]	2022	[51], [52]	Yes	Source Inference Attack

Chapter 4

Scenario Setup

The Scenario Setup Chapter is dedicated to discuss the experimental setup and will cover all the architectural details. First, an overview of the FL framework Fedstellar [13] will be given, including what datasets were used, how the training of CFL and DFL models were set up, and lastly, what scenarios are going to be evaluated in the experiments.

Secondly, the Privacy Meter framework [6], [14]–[16] and the selection of MIA that were launched onto the FL models, including its attack architecture will be covered. Additionally, the classification methods will be explained used in the MIA, what metrics were used to evaluate the performances, and how these metrics are calculated.

4.1 Federated Learning Framework

The FL framework Fedstellar provides users with a broad spectrum of modular, adaptable, and extensible functionality. If the user wishes to use a costume model architecture, the framework is easily adaptable as it follows a standardized way of implementation and folder structure. Furthermore, it provides a web interface, making it very accessible. The web interface gives the user an organized selection profile, with the following option: dataset: MNIS, FEMNIST, the training method: MLP, and CNN, the federation setting: CFL, and DFL, the topology for DFL settings: Ring, Fully Connected, Star, and custom topology,

1. **Scenario information**: Scenario Title, Scenario Description
2. **Deployment options**
    - Simulation, Real Devices, Data-Center
    - Accelerator definition: CPU, GPU
3. **Federation architecture**:CFL, DFL, SDFL

4. **Network Topology**:

    - Custom topology: The web interface shows a window with the current topology chosen and represents training and aggregator participants. These participants can be customized by left-clicking them. One can add, remove, or change the role of the selected participant.
    - Predefined topology: Star, Ring, Fully Connected, Random

5. **Number of participants**: Number Input

6. **Datasets**: MNIST, FEMNIST

7. **Communication rounds**: Number Input

8. **Training**:

    - Model: MLP, CNN
    - Number of Epochs: Number input

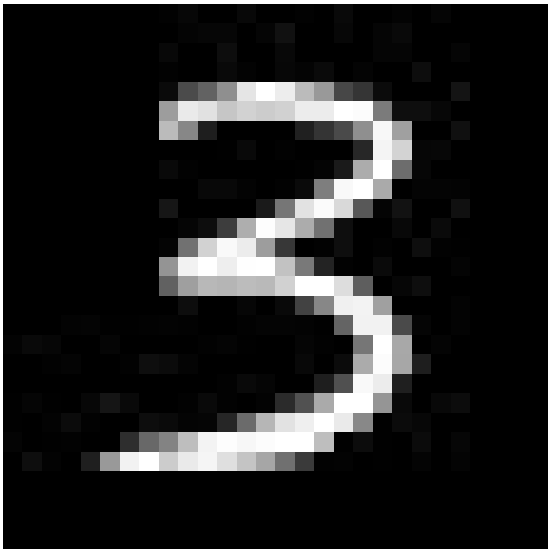
9. **Aggregation algorithm**: FedAvg

After selecting the desired configuration of parameters, the model is run and the learning can be observed via the tensorboard application. The models are saved locally with all the configuration settings stored in a separate folder.

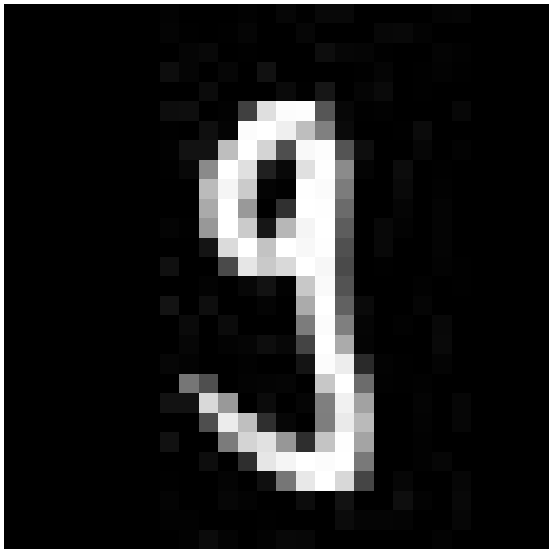
4.1.1 Datasets

For the experiments, the goal was to also explore datasets that are not related to the MNIST family but also explore the other MNIST-related datasets. Therefore, CIFAR-10 was added to the collection, as well as FashionMNIST and EMNIST with the letter split.

MNIST [80]: MNIST is an image dataset, containing handwritten digits. The training set consists of 60'000 grey-scale images and the test set of 10'000 grey-scale images. The digits on the images are centered and have an image size of 28 x 28 pixels. As this dataset only contains digits, there are ten different classes that can be classified, namely: 0, 1, 2, 4, 5, 6, 7, 8, and 9.



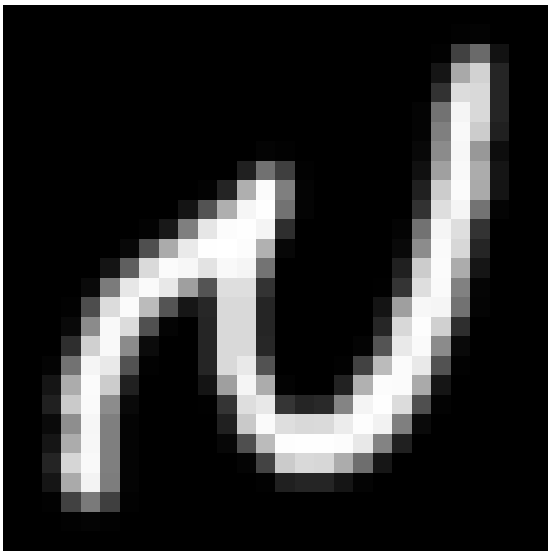
(a) MNIST dataset example: 3



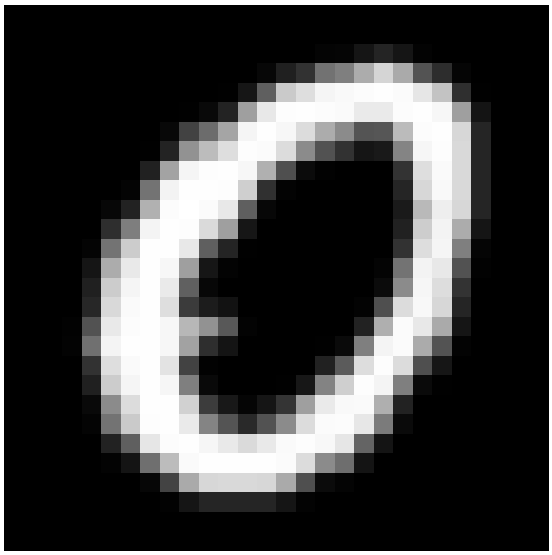
(b) MNIST dataset example: 9

Figure 4.1: Raw MNIST Images

EMNIST (letter split) [81]: EMNIST is an image dataset, containing handwritten letters. The training set consists of 88'800 grey-scale images and the test set of 14'800 grey-scale images. The images have a size of 28 x 28 pixels and contain 26 different classes, referring to the 26 letters of the phonetic NATO alphabet.



(a) EMNIST dataset example: n

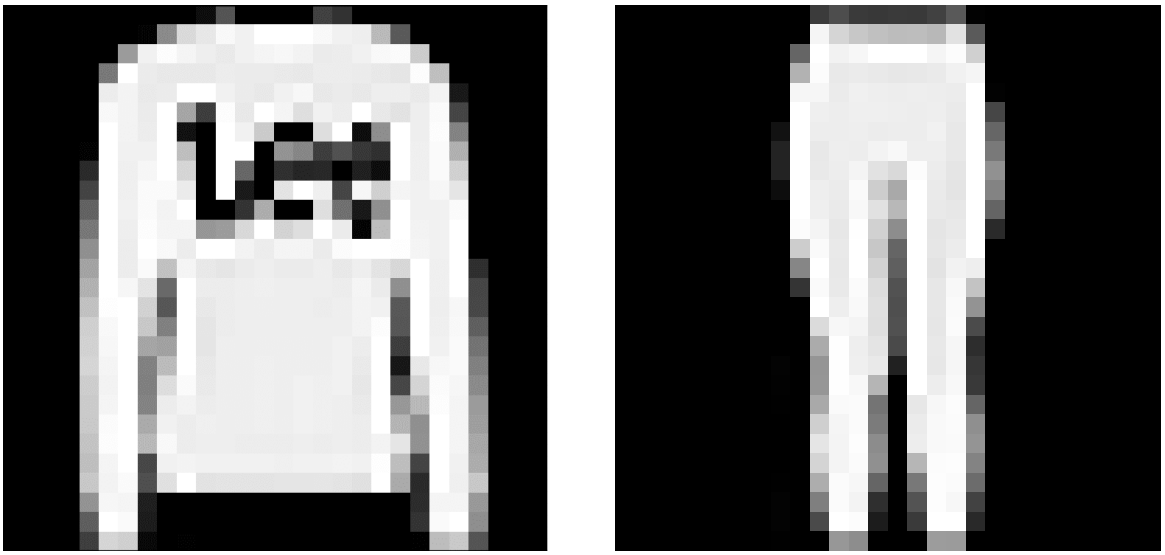


(b) EMNIST dataset example: o

Figure 4.2: Raw EMNIST Images

FashionMnist [82]: FashionMNIST is an image dataset, containing clothing articles from the online retailer Zalando. This dataset was developed to be used as a direct substitute for the MNIST dataset, as it shares the size of the training set and test set. Furthermore, it also has the same image size of 28 x 28 pixels and also shares the number of classes

with the MNIST dataset. The images can be classified into the following ten classes: T-Shirt/top, Trousers, Pullover, Dress, Coat, Sandal, Shirt, Sneaker, Bag, and Ankle boots.



(a) FashionMNIST dataset example: Pullover (b) FashionMNIST dataset example: Trouser

Figure 4.3: Raw FashionMNIST Images

CIFAR-10 [83]: CIFAR-10 is an image dataset, containing means of transportation and animals. The training set consists of 50'000 colored images and the test set of 10'000 colored images. The images have a size of 32 x 32 pixels, and the total amount of classes in CIFAR-10 are ten, which go as follows: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck.

(a) CIFAR-10 dataset example: Automobile (b) CIFAR-10 dataset example: Deer

Figure 4.4: Raw CIFAR-10 Images

4.1.2 Training Architecture

For training the FL models, the datasets were split according to the number of participants defined in the configurator of Fedstellar. As the computational resources were limited, all the models were trained on the minimum amount of nodes, where the topologies, Ring, and Fully Connected have distinct arrangements, see Figure 2.3, and Figure 2.2.
For the models trained on MNIST, and FashionMMNIST, the model definition followed the reference implementation of Fedstellar, including two convolutions. For the CIFAR-10 dataset, a custom implementation was used for the CNN architecture, comprised of 3, double convolutions [84].
Regarding the activation functions in between the layers, all scenarios use the ReLu activation function. Also after each CNN layer, a max pooling layer is used. All models share the same activation function, optimizer, and loss function, being ReLu, ADAM, and Crossentropyloss respectively.

4.1.3 Training Device Setup

The device at hand, during the training process, had the following system configurations:

- **OS:** UBUNTU 22.04.2 LTS
- **GPU:** GeForce RTX 3060
  - VRAM: 12GB
  - CUDA Driver Version: 11.6
- **RAM-Storage:** 16GB
- **HDD:** 1TB
- **Swap-Memory:** 4GB
- **Pyhton Version:** 3.10

4.1.4 Hyperparameters

In the following section, all hyperparameters per dataset will be listed, beginning with MNIST, followed by FashionMNIST and lastly, also CIFAR-10 will be covered.

MNIST-Hyperparameters :

- **Learning Rate**: 0.001
- **Number of Participants**: 4
- **Epochs per Round:** 4

- **Number of Rounds:** 3
- **Batch-Size:** 32
- **Number of workers:** 4
- **In-Channels:** 1
- **Out-Channels:** 10

FashionMNIST-Hyperparameters :

- **Learning Rate**: 0.0005
- **Number of Participants**: 4
- **Epochs per Round:** 4
- **Number of Rounds:** 3
- **Batch-Size:** 32
- **Number of workers:** 4
- **In-Channels:** 1
- **Out-Channels:** 10

CIFAR-10-Hyperparameters :

- **Learning Rate**: 0.0005
- **Number of Participants**: 4
- **Epochs per Round:** 20
- **Number of Rounds:** 7
- **Batch-Size:** 64
- **Number of workers:** 4
- **In-Channels:** 3
- **Out-Channels:** 10

4.2 Membership Inference Framework

Privacy Meter is an open-source framework, designed to assess data privacy within statistical and ML algorithms. It plays a crucial role in the data protection impact assessment process by quantifying fundamental privacy risks associated with ML models. The tool uses state-of-the-art Inference techniques for auditing a variety of ML algorithms used in areas such as classification, regression, computer vision, and natural language processing. It produces detailed reports outlining the privacy risks for data records in the training set, considering both Black-Box and White-Box adversarial knowledge levels [6], [14]–[16].

To assess the full range of privacy risks in FL settings, the jupyter notebook:
white_box_attack.ipynb, and shadow_metric.ipynb, was chosen. Both provide an accessible and transparent usage of the implemented audit systems.
The name 'white_box_attack.ipynb'is a bit misleading since both White-box and Black-box metrics are provided and the inference method used to audit the results is the 'PopulationMetric' attack.

4.2.1 Population Metric

First, there will be a systematic elaboration on how to successfully execute the Population Metric attack on ML models, followed by the process of how to successfully execute the attack on Fedstellar FL models.

Population Metric Inference Process:

1. After starting the jupyter-notebook-server the first thing to do is to run the 'Import' kernel.
2. From there, depending on what dataset the ML model should be trained, the respective hyper-parameter and model-definition kernel should be run.
3. The next steps consist of loading the matching dataset and splitting the data into a train, test, and population set.
4. Then the training and test data is loaded for training the ML model inside of the notebook.
5. Next up, the model is trained on the specified dataset, and hyper-parameters.
6. After completion, the trained model is wrapped to into the target-model variable.
7. Continuing, the dataset is formatted into a tensor.
8. Then, the audit, train, and test data and respectively audit, train, and test targets are defined using the tensor formatting function.
9. These subsets are then passed onto the target dataset, and audit dataset variables.

10. Lastly, the target and the reference objects are defined.
11. For launching the Black-Box attack, the Population Metric object is run, which uses the ModelLoss() class for inference.
12. For launching the White-Box attack, the Population metric object is run, which uses the ModelGradientNorm() class for inference.
13. After some time, the results will be saved into the indicated logs folder, and the reports can be generated, by running the desired report formats: ROC, Signal Histogram, or Confusion Matrix.

For MIA on Fedstellar models, the process is very similar. Step 3 changes, as the indices that were used for the FL setting, have to be considered and must therefore first be imported. Step 4 and 5 can be omitted. In step 6 the Fedstellar model has to be imported and wrapped accordingly. The rest of the steps can be followed as described for a successful audit result.

4.2.2 Shadow Metric Inference Process

1. After starting the jupyter-notebook-server the first thing to do is to run the 'Import' kernel.
2. From there, depending on what dataset the ML model should be trained, the respective hyper-parameter and model-definition kernel should be run.
3. The next steps consist of loading the matching dataset and splitting the data into train, and test sets.
4. Then the split data is preprocessed, such that there are defined splits for the shadow models, and further audit subsplits.
5. Next up, the data is wrapped into a dataset and target dataset object.
6. Then the dataset is subdivided according to the definition of the hyper-parameters, including the number of shadow models that should be trained.
7. Continuing, the subdivisions are enumerated.
8. From there, depending on what dataset is being used, the respective shadow model-definition kernel should be run.
9. Then the loss is defined and the shadow models are trained.
10. After completion, the shadow models are wrapped.
11. Lastly, the target and the reference objects are defined.
12. Before running the inference attack, the ShadowMetric() object and its respective classification method are defined.

13. Then the Shadow Metric attack is run, using the ModelLoss() class for training a binary classifier on the target model.

14. After some time, the results will be saved into the indicated logs folder, and the reports can be generated, by running the desired report formats: ROC, Signal Histogram, or Confusion Matrix.

For MIA on Fedstellar models, the process is very similar. Before preprocessing the dataset, the indices used for the target participant, have to be imported, adding a step to the process between steps 3 and 4. The rest is equivalent.

4.3 Discussion

Concluding the scenario setup, the experiments can be classified according to the taxonomy as described in Chapter 2.3. Both adversarial knowledge levels are covered, as the Population Metric provides both White- and Black-Box inference attacks. Furthermore, also both classification methods are evaluated, with the Shadow Metric using a binary classifier, and the Population Metric using a prediction loss-based classifier.

The inference target is a single target, more specifically, all inference attacks were carried out on the participant with the index 0, making it a global attack for CFL federations, and a local attack for DFL federations.

Lastly, according to various definitions the inference mode can be classified as being a passive attack, as all inference attacks were launched after the training of the models finished.

Chapter 5

Evaluation

Chapter 5 will present the evaluations of the experiments carried out during the process of this thesis as depicted in Chapter 4. All subchapters are further split into sections, where each dataset will be looked at separately. First, in Chapter 5.1, all the results of the model accuracies and inference performances will be presented in three tables, for a quick overview of the results. Secondly, the MIA metrics on ML and CFL models will be compared. Followed by the comparison of MIA performance on CFL and DFL models. Lastly, the different topologies used in the DFL training process will be evaluated on their privacy-preserving differences and put into comparison.

5.1 Model Training and Membership Inference Evaluation

Table 5.1: Training and MIA Performance on MNIST Dataset

MNIST											
Model	Train	Test	White-Box Attack			Black-Box Attack			Shadow Metric		
	Acc	Acc	ROC	F1	AP	ROC	F1	AP	ROC	F1	AP
ML	99.62%	98.72%	50.8%	65.5%	51.3%	51.1%	65.1%	51.2%	-	-	-
CFL	98%	81%	50.4%	62.7%	90%	50.6%	62.3%	90.3%	50.2%	90.8%	83.3%
DFL-FC	98.8%	86.5%	49.4%	62.6%	89.6%	49.1%	62%	88%	50.9%	90.7%	83.9%
DFL-Ring	98.5%	81.2%	48.7%	59.5%	89.6%	48.8%	59.9%	89.7%	50.7%	90.7%	83.9%

Table 5.2: Training and MIA Performance on FashionMNIST Dataset

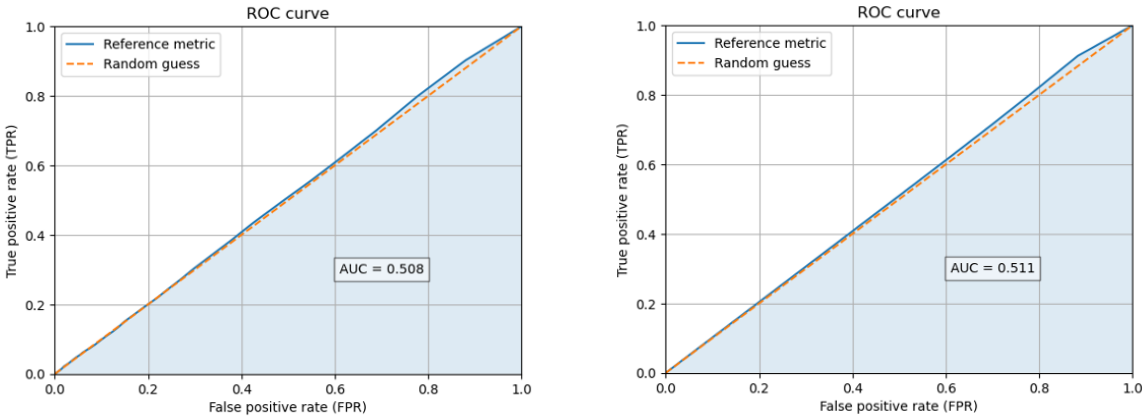
FashionMNIST											
Model	Train	Test	White-Box Attack			Black-Box Attack			Shadow Metric		
	Acc	Acc	ROC	F1	AP	ROC	F1	AP	ROC	F1	AP
ML	98.55%	89.56%	53.4%	66.8%	53%	53.4%	62.8%	52.5%	-	-	-
CFL	89.5%	89.4%	49.6%	60.8%	89.9%	49.6%	60.7%	89.9%	49.8%	85.2%	84.4%
DFL-FC	90.7%	89.9%	49.4%	51.6%	89.5%	49.4%	51.7%	89.6%	51.5%	86.3%	84.8%
DFL-Ring	89.3%	90%	52%	66.2%	90.3%	52%	66.5%	90.4%	53%	86.6%	85.2%

Table 5.3: Training and MIA Performance on CIFAR-10 Dataset

CIFAR-10											
Model	Train	Test	White-Box Attack			Black-Box Attack			Shadow Metric		
	Acc	Acc	ROC	F1	AP	ROC	F1	AP	ROC	F1	AP
ML	92.8%	48.83%	73%	76.2%	66.9%	73.4%	77.2%	65.6%	-	-	-
CFL	78.4%	30%	49.1%	62.5%	89.8%	50.4%	68.5%	90.2%	50.5%	40.1%	81.7%
DFL-FC	82.1%	60%	48.3%	49.2%	89%	48.5%	58.6%	89.8%	51.7%	46.3%	82.8%
DFL-Ring	82.3%	64.6%	51.5%	77.7%	90.4%	51.9%	78.2%	90.3%	51.5%	59%	82.5%

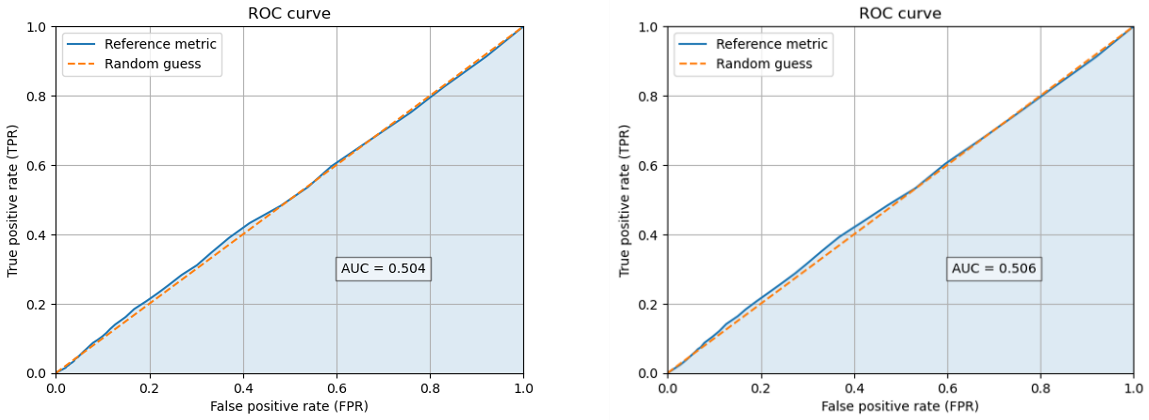
5.2 ML vs CFL

5.2.1 MNIST



(a) White-Box ROC on MNIST ML Model (b) Black-Box ROC on MNIST ML model

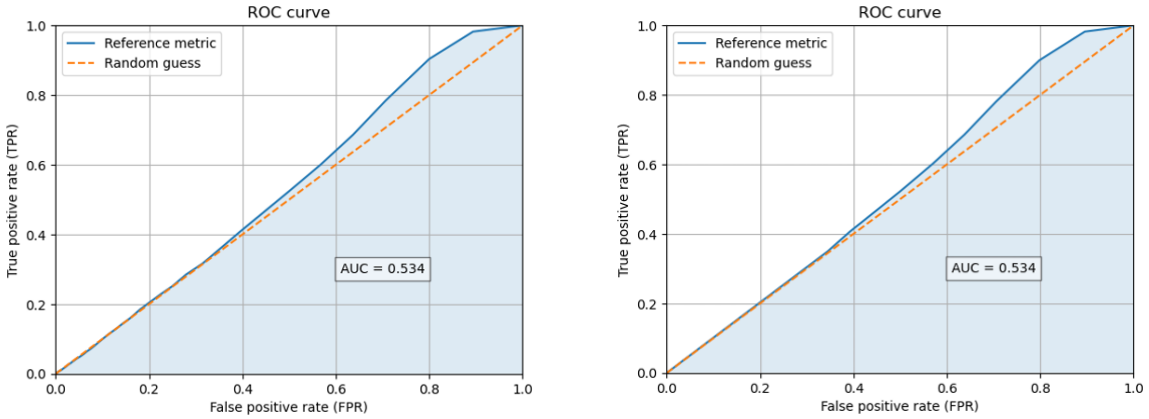
Figure 5.1: MNIST: ROC value comparison of White-Box and Black-Box attack on ML model



(a) White-Box ROC on MNIST CFL model (b) Black-Box ROC on MNIST CFL model

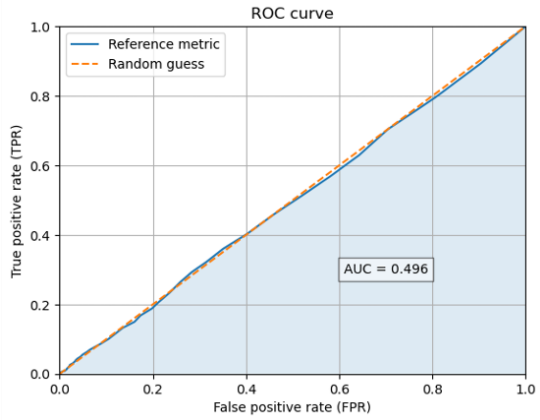
Figure 5.2: MNIST: ROC value comparison of White-Box and Black-Box attack on CFL model

5.2.2 FashionMNIST

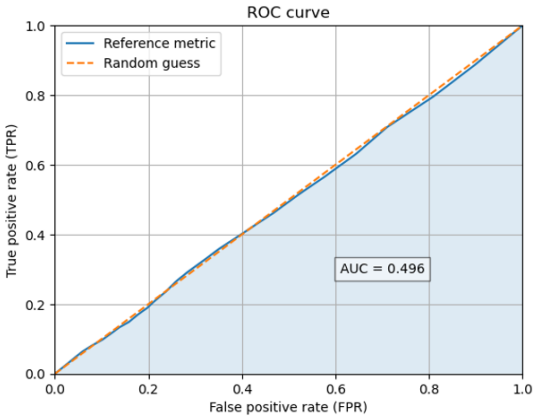


(a) White-Box ROC on FashionMNIST ML Model (b) Black-Box ROC on FashionMNIST ML model

Figure 5.3: FashionMNIST: ROC value comparison of White-Box and Black-Box attack on ML model



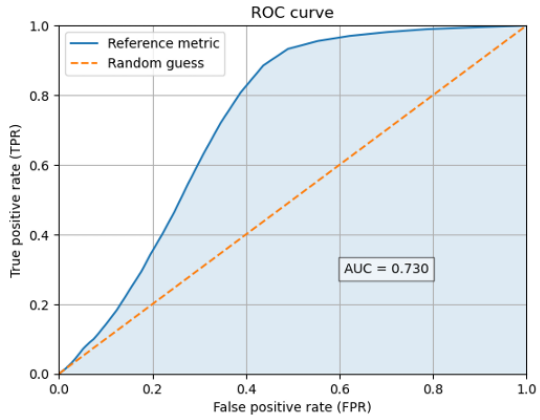
(a) White-Box ROC on MNIST CFL model



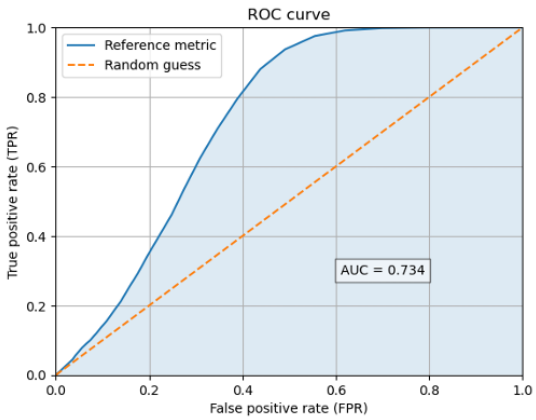
(b) Black-Box ROC on MNIST CFL model

Figure 5.4: FashionMNIST: ROC value comparison of White-Box and Black-Box attack on CFL model

5.2.3 CIFAR-10

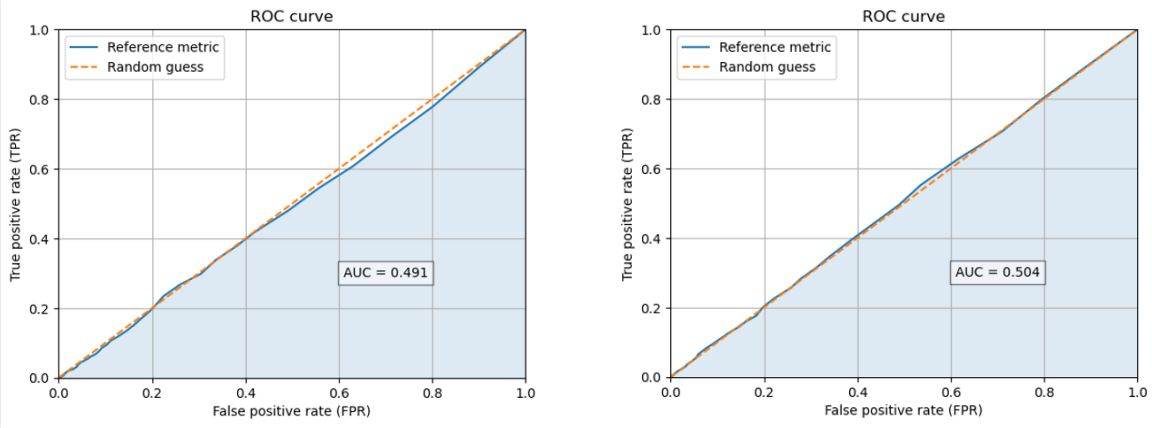


(a) White-Box ROC on CIFAR-10 ML Model



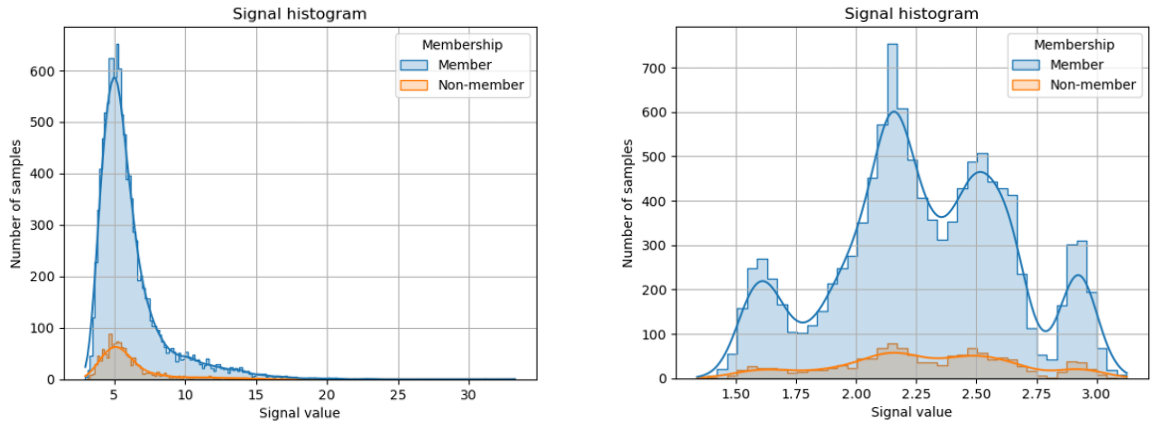
(b) Black-Box ROC on CIFAR-10 ML model

Figure 5.5: CIFAR-10: ROC value comparison of White-Box and Black-Box attack on ML model



(a) White-Box ROC on CIFAR-10 CFL model (b) Black-Box ROC on CIFAR-10 CFL model

Figure 5.6: CIFAR-10: ROC value comparison of White-Box and Black-Box attack on CFL model

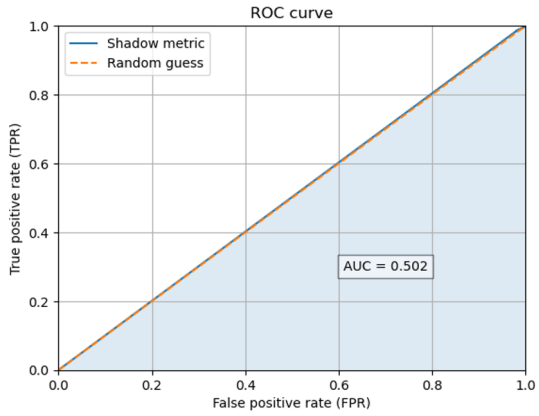


(a) Signal-Histogram of White-Box attack on CIFAR-10 CFL model (b) Signal-Histogram of Black-Box attack on CIFAR-10 CFL model

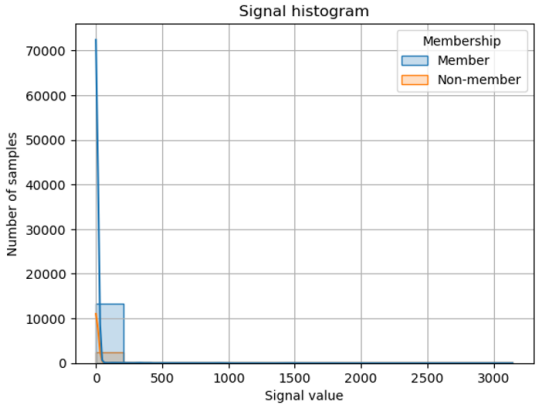
Figure 5.7: CIFAR-10: Signal Histogram comparison of White-Box and Black-Box attack on CFL model

5.3 CFL vs DFL Fully Connected

5.3.1 MNIST

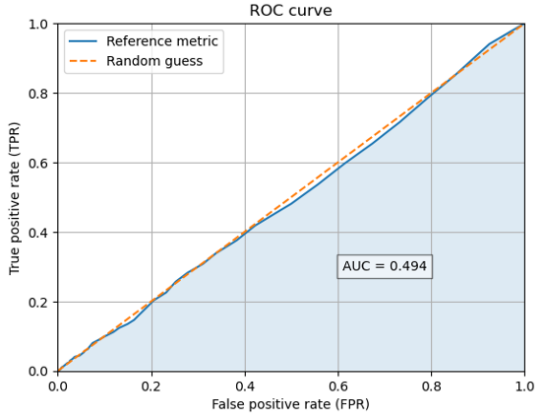


(a) Shadow Metric ROC on MNIST CFL model CFL model

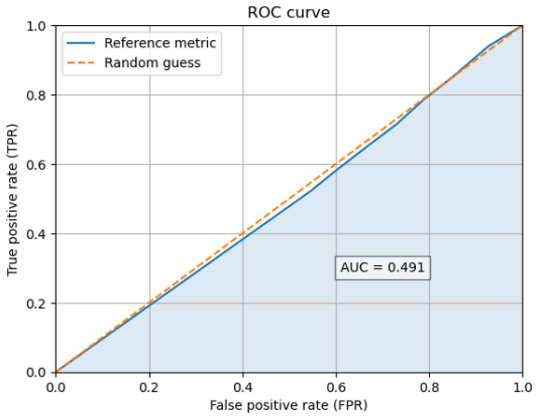


(b) Shadow Metric Signal-Histogram on MNIST

Figure 5.8: MNIST: ROC value comparison of Shadow Metric attack on CFL model

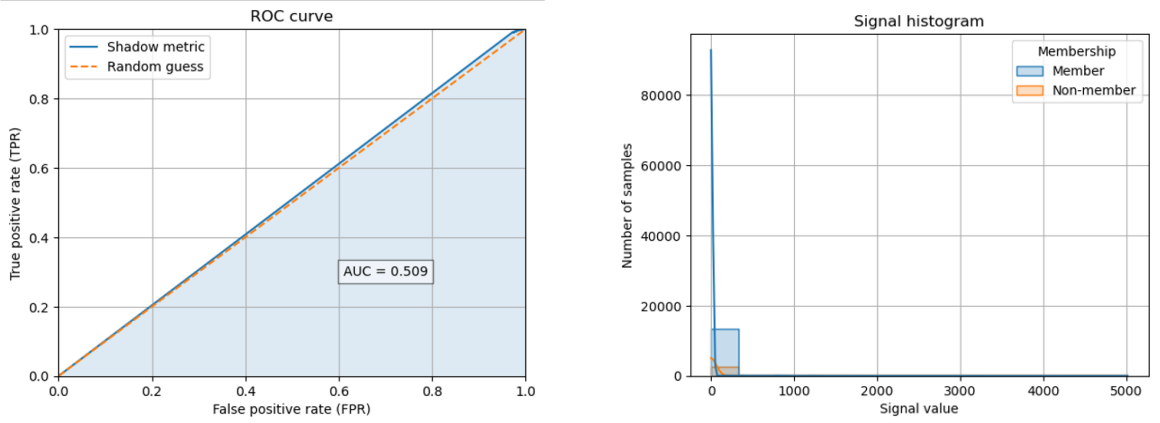


(a) White-Box ROC on MNIST DFL model, Topology: Fully Connected



(b) Black-Box ROC on MNIST DFL model, Topology: Fully Connected

Figure 5.9: MNIST: ROC value comparison of White-Box and Black-Box attack on DFL model, Topology Fully Connected

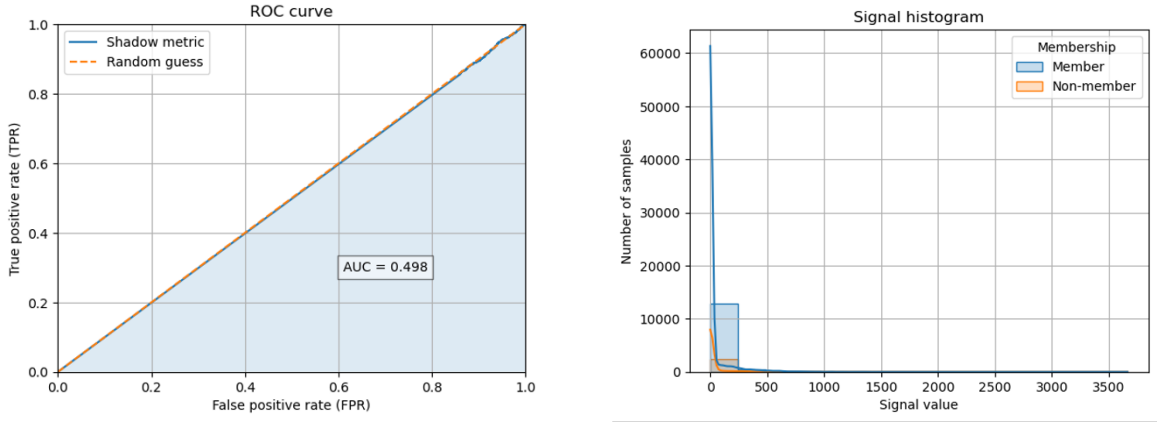


(a) Shadow Metric ROC on MNIST DFL model, Topology: Fully Connected

(b) Shadow Metric Signal Histogram on MNIST DFL model, Topology: Fully Connected

Figure 5.10: MNIST: ROC and Signal-Histogram Graphs of Shadow Metric attack on DFL model, Topology Fully Connected

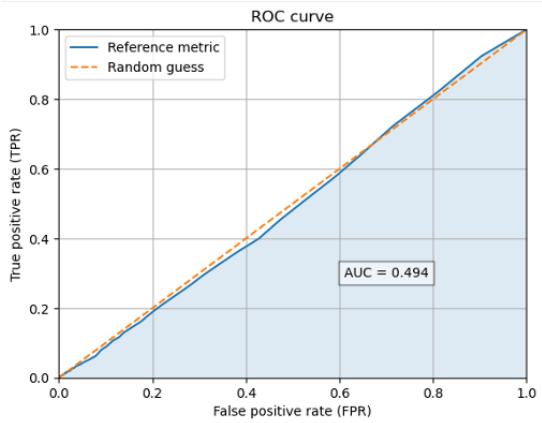
5.3.2 FashionMNIST



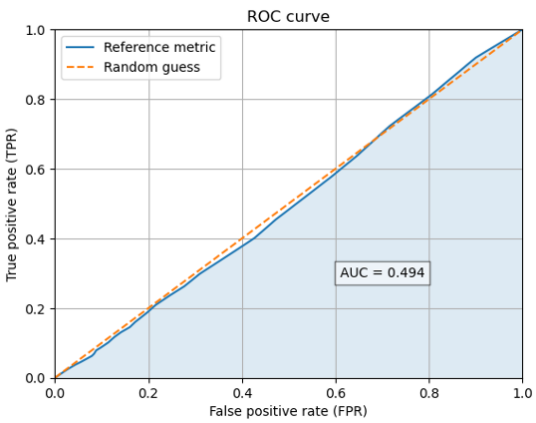
(a) Shadow Metric ROC on FashionMNIST CFL model

(b) Shadow Metric Signal-Histogram on FashionMNIST CFL model

Figure 5.11: FashionMNIST: ROC and Signal-Histogram Graphs of Shadow Metric attack on CFL model

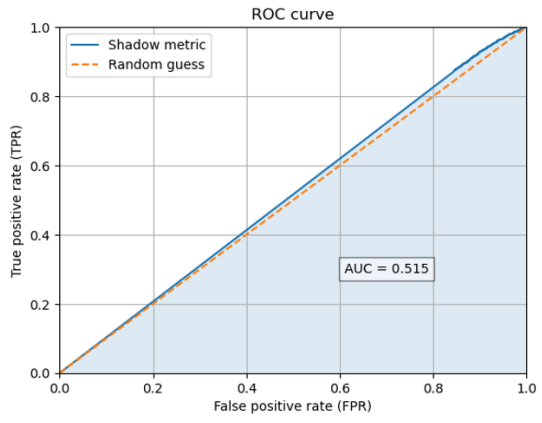


(a) White-Box ROC on FashionMNIST DFL model, Topology: Fully Connected

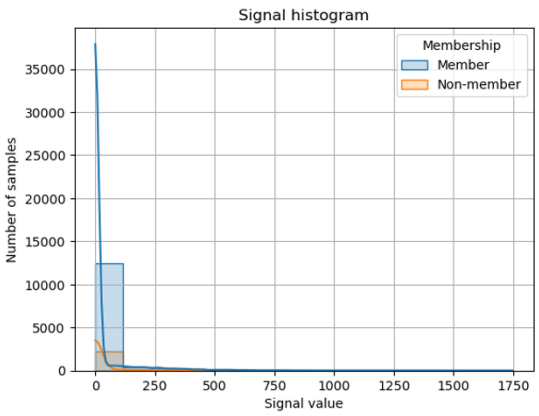


(b) Black-Box ROC on FashionMNIST DFL model, Topology: Fully Connected

Figure 5.12: FashionMNIST: ROC value comparison of White-Box and Black-Box attack on DFL model, Topology Fully Connected



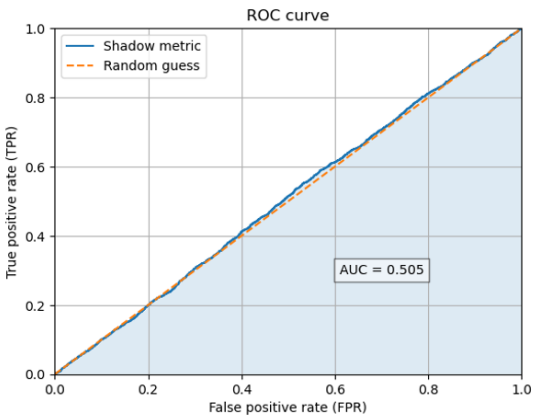
(a) Shadow Metric ROC on FashionMNIST DFL model, Topology: Fully Connected



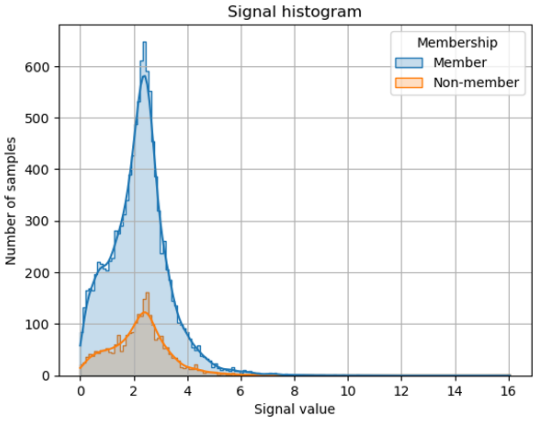
(b) Shadow Metric Signal Histogram on FashionMNIST DFL model, Topology: Fully Connected

Figure 5.13: FashionMNIST: ROC and Signal-Histogram Graphs of Shadow Metric attack on DFL model, Topology Fully Connected

5.3.3 CIFAR-10

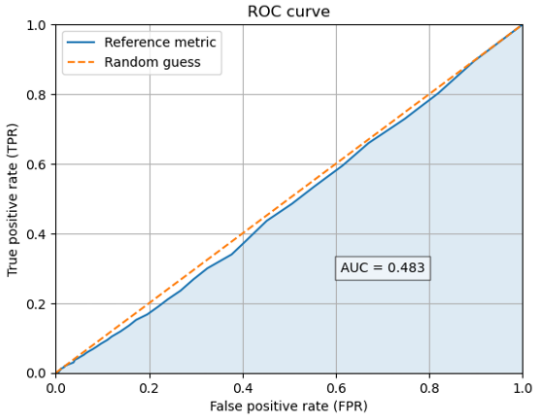


(a) Shadow Metric ROC on CIFAR-10 CFL model

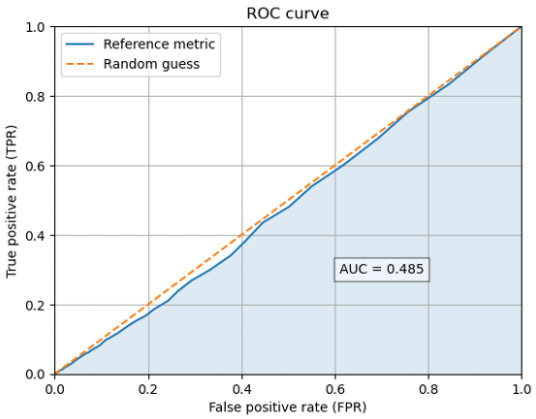


(b) Shadow Metric Signal-Histogram on CIFAR-10 CFL model

Figure 5.14: CIFAR-10: ROC and Signal-Histogram Graphs of Shadow Metric attack on CFL model

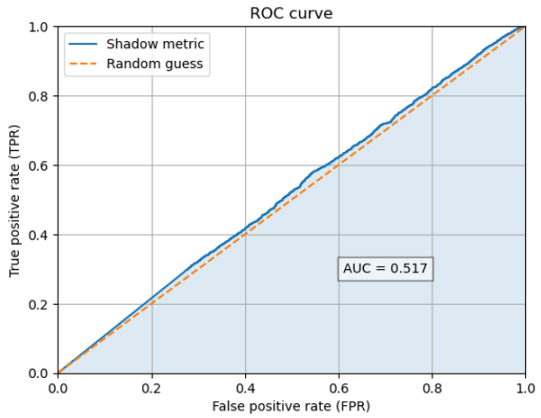


(a) White-Box ROC on CIFAR-10 DFL model, Topology: Fully Connected

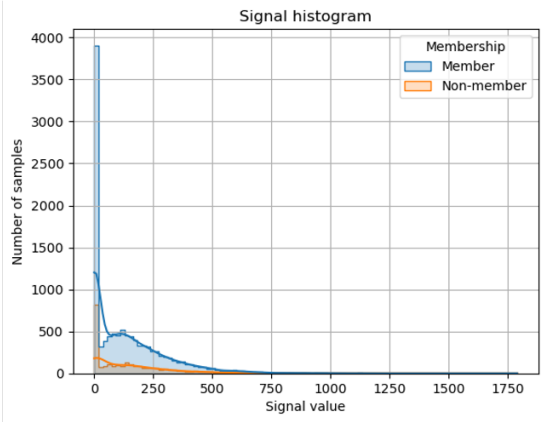


(b) Black-Box ROC on CIFAR-10 DFL model, Topology: Fully Connected

Figure 5.15: CIFAR-10: ROC value comparison of White-Box and Black-Box attack on DFL model, Topology Fully Connected



(a) Shadow Metric ROC on CIFAR-10 DFL model, Topology: Fully Connected

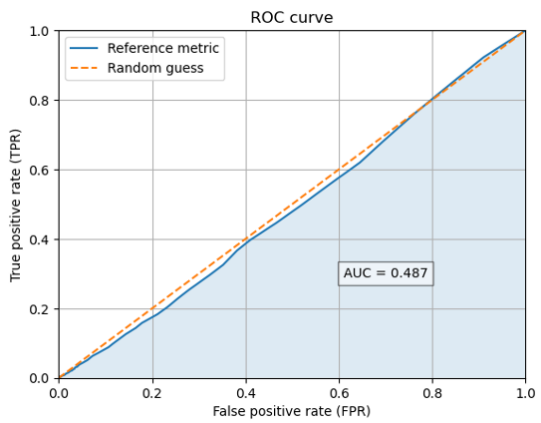


(b) Shadow Metric Signal Histogram on CIFAR-10 DFL model, Topology: Fully Connected

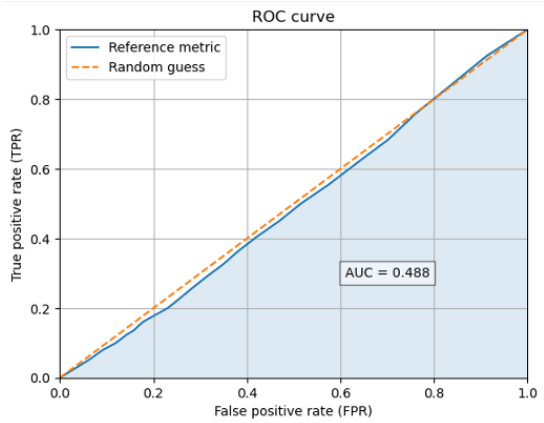
Figure 5.16: CFIAR-10: ROC and Signal-Histogram Graphs of Shadow Metric attack on DFL model, Topology Fully Connected

5.4 DFL: Fully-Connected vs Ring

5.4.1 MNIST

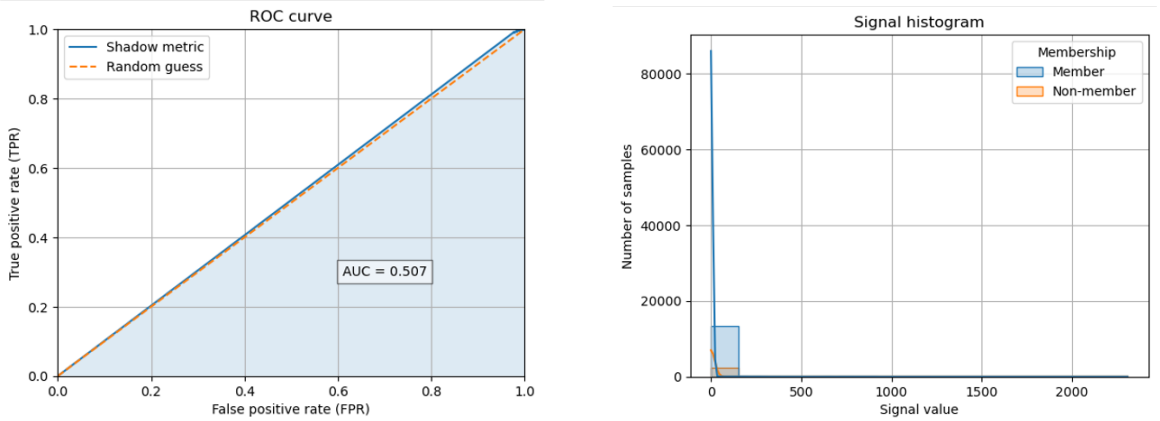


(a) White-Box ROC on MNIST DFL model, Topology: Ring



(b) Black-Box ROC on MNIST DFL model, Topology: Ring

Figure 5.17: MNIST: ROC value comparison of White-Box and Black-Box attack on DFL model, Topology Ring

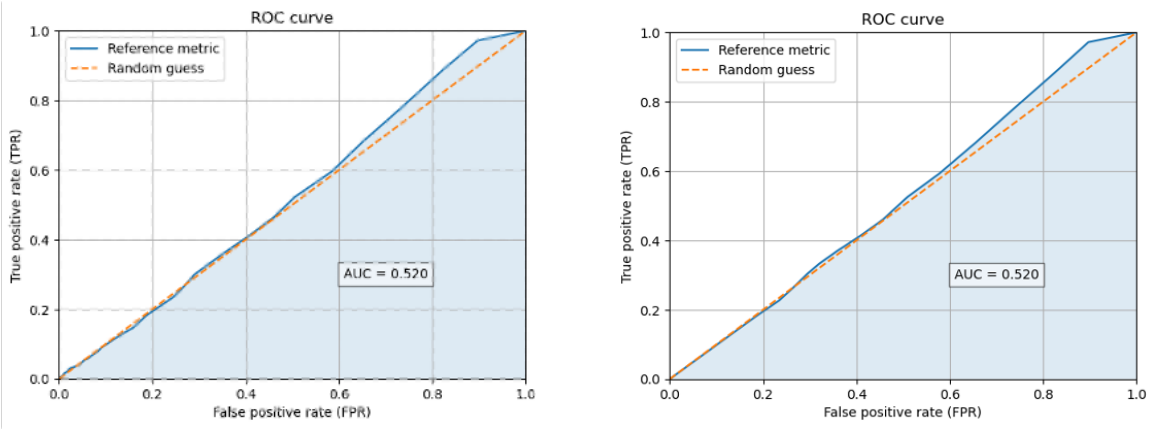


(a) Shadow Metric ROC on MNIST DFL model, Topology: Ring

(b) Shadow Metric Signal Histogram on MNIST DFL model, Topology: Ring

Figure 5.18: MNIST: ROC and Signal-Histogram Graphs of Shadow Metric attack on DFL model, Topology: Ring

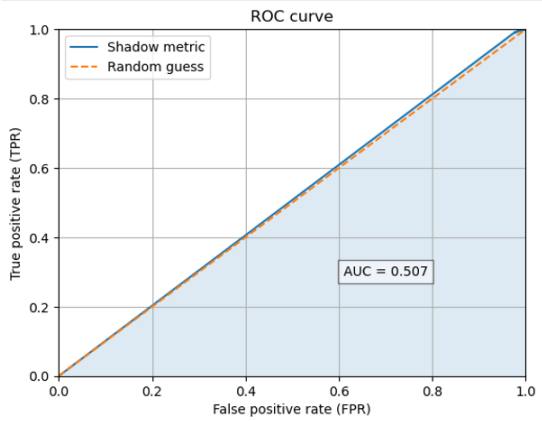
5.4.2 FashionMNIST



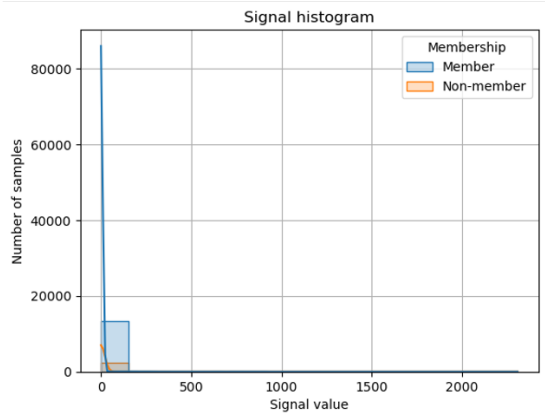
(a) White-Box ROC on FashionMNIST DFL model, Topology: Ring

(b) Black-Box ROC on FashionMNIST DFL model, Topology: Ring

Figure 5.19: FashionMNIST: ROC value comparison of White-Box and Black-Box attack on DFL model, Topology: Ring



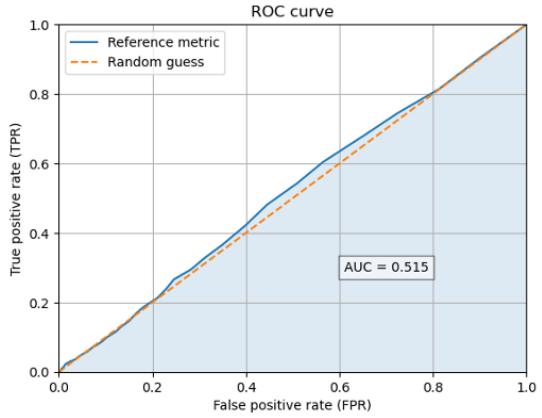
(a) Shadow Metric ROC on FashionMNIST DFL model, Topology: Ring



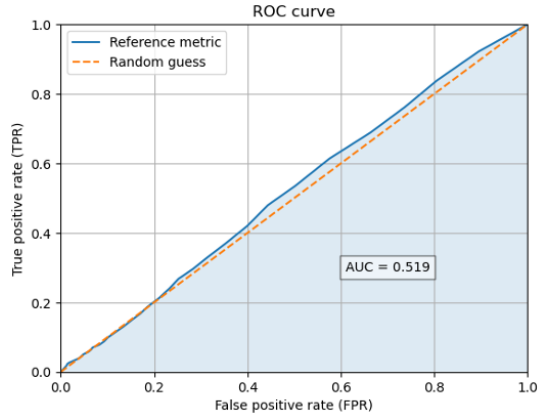
(b) Shadow Metric Signal Histogram on FashionMNIST DFL model, Topology: Ring

Figure 5.20: FashionMNIST: ROC and Signal-Histogram Graphs of Shadow Metric attack on DFL model, Topology: Ring

5.4.3 CIFAR-10

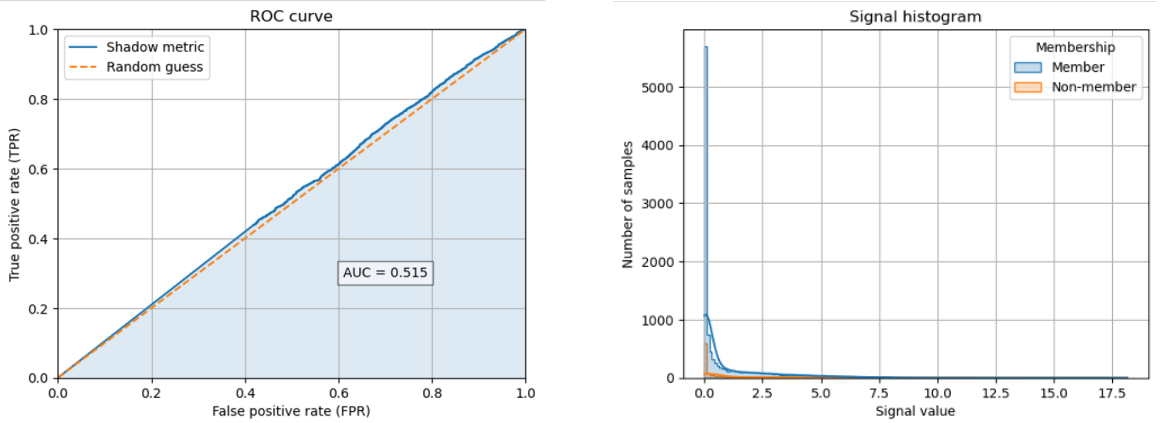


(a) White-Box ROC on CIFAR-10 DFL model, Topology: Ring



(b) Black-Box ROC on CIFAR-10 DFL model, Topology: Ring

Figure 5.21: CIFAR-10: ROC value comparison of White-Box and Black-Box attack on DFL model, Topology: Ring



(a) Shadow Metric ROC on CIFAR-10 DFL model, Topology: Ring

(b) Shadow Metric Signal Histogram on CIFAR-10 DFL model, Topology: Ring

Figure 5.22: CFIAR-10: ROC and Signal-Histogram Graphs of Shadow Metric attack on DFL model, Topology: Ring

5.5 Conclusion

5.5.1 MNIST

Comparing the ROC and F1 values of Table 5.1, the DFL-Ring model is by far the most robust model against White-Box and Black-Box attacks. As expected, the ML model finishes last in terms of privacy-preserving abilities, even though being well generalized. Furthermore, the DFL-FC model seems to have a ROC advantage over the CFL model by over 10% for the Black-Box attack and exactly a 10% advantage for the White-Box attack, while also having the best Test-Accuracy out of all FL settings. Looking at the results of the Shadow Metric, it seems that the CFL model leaks the least amount of information, followed by the DFL-Ring model, and lastly the DFL-FC model.

5.5.2 FashionMNIST

Comparing the ROC and F1 values of Table 5.2, the DFL-FC model seems to be the model with the most robust performance against White- and Black-Box attacks. Again, the ML model finishes last against the Population Metric attacks. Although for the Black-Box attack, the F1 score is slightly better than the DFL-Ring one. In contrast to the models trained on the MNIST dataset, the CFL model on FashionMNIST has a ROC advantage of 14% over the DFL-Ring model against both White- and Black-Box attacks. For the Shadow Model attack, the CFL model is again taking the lead in terms of privacy preservation, followed by the DFL-FC model. Leaving the DFL-Ring model to be the most susceptible, in terms of inference attacks.

5.5.3 CIFAR-10

Comparing the ROC and F1 values of Table 5.3, the DFL-FC model is again taking the lead in terms of performance against White- and Black-Box attacks. As expected the ML model has the highest ROC values for the Population Metric attacks, but looking at the F1-Score, the DFL-Ring model performs worse than the ML model. Again, the DFL-FC model shows an advantage over the CFL model against the Population Metric attacks, while also being way better generalized than the CFL model. Comparing the FL models against the Shadow Metric inference, the CFL model poses to be the most robust model over all the datasets. Looking at the ROC value, the DFL-Ring is slightly better, but much worse if looking at the F1-Score compared to the DFL-FC model.

Chapter 6

Summary, and Conclusions

The final Chapter of this thesis will focus on reflecting on the journey of this project, proposing the findings that have been made, and the implications they hold for the field of FL and data privacy. First, a complete summary of the thesis will be given, including the importance of this work. Secondly, the main contributions of this work will be presented, by taking the findings from the evaluations and putting them into context. Following, the limitations of this work are stated. The last part of this chapter will suggest potential avenues for future projects on this pressing topic.

The integration of sensitive data from various sectors such as healthcare, finance, IoT, and marketing into ML training algorithms necessitates prioritizing privacy preservation and responsible data practices. FL first introduced by Google, offers a promising approach to this end. FL enables the collaborative development of a shared global model across multiple participants while preserving data privacy, as it eliminates the need for direct data sharing. This technique also addresses issues related to network and computational bottlenecks, especially with IoT devices, by reducing the constant flow of raw training data and instead sharing only model parameters or gradients.

Two settings for FL have been studied extensively: CFL and DFL. CFL involves a central server (CS) aggregating model updates from various nodes, forming a global model that is subsequently shared with each participant for further training. Despite its wide use, CFL has several drawbacks, including the potential for high latency, network synchronization problems due to varying bandwidths among participants, bottlenecks at the CS, and heightened security risks as the CS becomes a single point of attack.

DFL, developed to address CFL's shortcomings, distributes the responsibility of aggregation to each participant. Although both settings prevent raw data sharing, they remain susceptible to adversarial attacks, including MIA, which infer whether a data record was used to train a target model. Model inversion, property inference, reconstruction attacks, and source inference attacks also belong to the family of adversarial attacks and have been shown to be as threatening as MIA. This susceptibility raises significant concerns given the application of FL application in sensitive domains.

In this thesis, the privacy vulnerability posed by MIA affecting FL settings were investigated, comparing the stability of CFL and DFL against MIA through various performance

metrics. Furthermore, a set of defense techniques was presented, including DP, regularization methods, and knowledge distillation.

These investigations have led to several concluding points:

- For all datasets, FL models turned out to be strictly better in terms of preserving data privacy, if the ROC value is chosen as the main comparison metric.
- For all datasets, CFL outperformed all other model configurations against shadow model MIA.
- The DFL-Ring model trained on MNIST, had the best data privacy preserving performance against the Population Metric attacks.
- The DFL-FC model trained on FashionMNIST and CIFAR-10, was proven to be the the least affected model against the Population Metric attacks.

Finally, these results reflect the true data privacy-preserving abilities of CFL and DFL in different scenarios. CFL taking the lead against binary classification-based MIA, and DFL models on fully connected network topologies, against MIA, using prediction-based classifier.

6.1 Future Work

Given the widespread application of Federated Learning in various sensitive domains, these findings have significant implications, emphasizing the need for continuous research and development of more secure, privacy-preserving mechanisms in the field of FL. Future directions could be drawn by extending the FL framework Fedstellar with defense strategies against adversarial attacks in general. Furthermore, an extension could be built based on the MIA framework Privacy Meter to incorporate, not only passive attacks but also active MIA attacks, which might pose a greater threat.

Bibliography

[1] H. Chen, H. Li, G. Dong, *et al.*, "Practical membership inference attack against collaborative inference in industrial iot", *IEEE Transactions on Industrial Informatics*, vol. 18, no. 1, pp. 477–487, 2022. DOI: 10.1109/TII.2020.3046648.

[2] H. B. McMahan, E. Moore, D. Ramage, and B. A. y Arcas, "Federated learning of deep networks using model averaging", *CoRR*, vol. abs/1602.05629, 2016. arXiv: 1602.05629. [Online]. Available: http://arxiv.org/abs/1602.05629.

[3] E. T. M. Beltrán, M. Q. Pérez, P. M. S. Sánchez, *et al.*, "Decentralized federated learning: Fundamentals, state-of-the-art, frameworks, trends, and challenges", 2022. arXiv: 2211.08413 [cs.LG].

[4] D. Reinsel, J. Gantz, J. Rydning, *et al.*, "The digitization of the world from edge to core", Tech. Rep., Nov. 2018.

[5] L. He, A. Bian, and M. Jaggi, *Cola: Decentralized linear learning*, 2019. arXiv: 1808.04883 [cs.DC].

[6] M. Nasr, R. Shokri, and A. Houmansadr, "Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning", in *Proceedings of the 2019 IEEE Symposium on Security and Privacy (SP)*, IEEE, May 2019, pp. 1–15. DOI: 10.1109/sp.2019.00065. [Online]. Available: https://doi.org/10.1109/2Fsp.2019.00065.

[7] J. Zhang, J. Zhang, J. Chen, and S. Yu, "Gan enhanced membership inference: A passive local attack in federated learning", in *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, 2020, pp. 1–6. DOI: 10.1109/ICC40277.2020.9148790.

[8] H. Lee, J. Kim, S. Ahn, R. Hussain, S. Cho, and J. Son, "Digestive neural networks: A novel defense strategy against inference attacks in federated learning", *Computers Security*, vol. 109, p. 102 378, 2021, ISSN: 0167-4048. DOI: https://doi.org/10.1016/j.cose.2021.102378. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167404821002029.

[9] J. Chen, J. Zhang, Y. Zhao, H. Han, K. Zhu, and B. Chen, "Beyond model-level membership privacy leakage: An adversarial approach in federated learning", in *2020 29th International Conference on Computer Communications and Networks (ICCCN)*, 2020, pp. 1–9. DOI: 10.1109/ICCCN49398.2020.9209744.

[10] H. Hu, Z. Salcic, L. Sun, G. Dobbie, and X. Zhang, "Source inference attacks in federated learning", *CoRR*, vol. abs/2109.05659, 2021. arXiv: 2109.05659. [Online]. Available: https://arxiv.org/abs/2109.05659.

[11] L. J. Ba and R. Caruana, "Do deep nets really need to be deep?", in *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, ser. NIPS'14, Montreal, Canada: MIT Press, 2014, pp. 2654–2662.

[12] H. Hu, Z. Salcic, L. Sun, G. Dobbie, P. S. Yu, and X. Zhang, *Membership inference attacks on machine learning: A survey*, 2022. arXiv: 2103.07853 [cs.LG].

[13] *Fedstellar: Framework for decentralized federated learning)*, https://federatedlearning.inf.um.es, Accessed: 2023-05-11.

[14] J. Ye, A. Maddi, S. K. Murakonda, V. Bindschaedler, and R. Shokri, "Enhanced membership inference attacks against machine learning models", in *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, 2022, pp. 3093–3106. DOI: 10.48550/arXiv.2111.09679. [Online]. Available: https://doi.org/10.48550/arXiv.2111.09679.

[15] S. Kumar and R. Shokri, "Ml privacy meter: Aiding regulatory compliance by quantifying the privacy risks of machine learning", in *The 20th Privacy Enhancing Technologies Symposium, Workshop on Hot Topics in Privacy Enhancing Technologies (HotPETs)*, 2020. DOI: 10.48550/arXiv.2007.09339. [Online]. Available: https://doi.org/10.48550/arXiv.2007.09339.

[16] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models", in *2017 IEEE Symposium on Security and Privacy (SP)*, IEEE, 2017, pp. 3–18. DOI: 10.1109/SP.2017.41. [Online]. Available: https://doi.org/10.1109/SP.2017.41.

[17] V. Mothukuri, R. M. Parizi, S. Pouriyeh, Y. Huang, A. Dehghantanha, and G. Srivastava, "A survey on security and privacy of federated learning", *Future Generation Computer Systems*, vol. 115, pp. 619–640, 2021, ISSN: 0167-739X. DOI: https://doi.org/10.1016/j.future.2020.10.007. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X20329848.

[18] *Vertex ai*, https://cloud.google.com/vertex-ai, Accessed: 2023-05-07.

[19] *Machine learning on aws*, https://aws.amazon.com/machine-learning, Accessed: 2023-05-07.

[20] *Welcome to machine learning studio (classic)*, https://studio.azureml.net, Accessed: 2010-09-30.

[21] *Machine learning made beautifully simple for everyone*, https://bigml.com, Accessed: 2023-05-07.

[22] D. Saad, "Online algorithms and stochastic approximations", *Online Learning*, 1998.

[23] D. P. Kingma and J. Ba, *Adam: A method for stochastic optimization*, 2017. arXiv: 1412.6980 [cs.LG].

[24] T. Hastie, R. Tibshirani, and J. Friedman, *The elements of statistical learning: data mining, inference and prediction*, 2nd ed. Springer, 2009. [Online]. Available: http://www-stat.stanford.edu/~tibs/ElemStatLearn/.

[25] S. Yeom, I. Giacomelli, M. Fredrikson, and S. Jha, *Privacy risk in machine learning: Analyzing the connection to overfitting*, 2018. arXiv: 1709.01604 [cs.CR].

[26] N. Carlini, S. Chien, M. Nasr, S. Song, A. Terzis, and F. Tramer, *Membership inference attacks from first principles*, 2022. arXiv: 2112.03570 [cs.CR].

[27] R. Sathya and A. Abraham, "Comparison of supervised and unsupervised learning algorithms for pattern classification", *International Journal of Advanced Research in Artificial Intelligence*, vol. 2, no. 2, pp. 34–38, 2013. DOI: 10.14569/IJARAI.2013.020206. [Online]. Available: http://dx.doi.org/10.14569/IJARAI.2013.020206.

[28] *Supervised vs. unsupervised learning: What's the difference?*, https://www.ibm.com/cloud/blog/supervised-vs-unsupervised-learning, Accessed: 2023-05-07.

[29] L. Lyu, H. Yu, and Q. Yang, *Threats to federated learning: A survey*, 2020. arXiv: 2003.02133 [cs.CR].

[30] M. Yemini, R. Saha, E. Ozfatura, D. Gunduz, and A. Goldsmith, "Semi-decentralized federated learning with collaborative relaying", in *2022 IEEE International Symposium on Information Theory, ISIT 2022*, ser. IEEE International Symposium on Information Theory - Proceedings, Institute of Electrical and Electronics Engineers Inc., 2022, pp. 1471–1476. DOI: 10.1109/ISIT50566.2022.9834707.

[31] F. Lin, S. Hosseinalipour, S. Azam, C. Brinton, and N. Michelusi, "Semi-decentralized federated learning with cooperative d2d local model aggregations", English (US), *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 12, pp. 3851–3869, Dec. 2021, Publisher Copyright: © 1983-2012 IEEE., ISSN: 0733-8716. DOI: 10.1109/JSAC.2021.3118344.

[32] *Tensorflor federated: Machine learning on decentralized data*, https://www.tensorflow.org/federated, Accessed: 2023-05-07.

[33] *PySyft: A Library for Easy Federated Learning*. Springer, Cham, 2021, pp. 111–139. DOI: 10.1007/978-3-030-70604-3_5.

[34] K. Cheng, T. Fan, Y. Jin, *et al.*, *Secureboost: A lossless federated learning framework*, 2021. arXiv: 1901.08755 [cs.LG].

[35] C. He, S. Li, J. So, *et al.*, *Fedml: A research library and benchmark for federated machine learning*, 2020. arXiv: 2007.13518 [cs.LG].

[36] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, *Communication-efficient learning of deep networks from decentralized data*, 2023. arXiv: 1602.05629 [cs.LG].

[37] Y. Xiao, S. Huang, M. Xiao, S. Mumtaz, and O. Dobre, "Fully decentralized federated learning based on-board mission for uav swarm system", *IEEE Communications Letters*, vol. 25, no. 10, pp. 3296–3300, Jul. 2021, ISSN: 1089-7798. DOI: 10.1109/LCOMM.2021.3095362.

[38] T. Vogels, H. Hendrikx, and M. Jaggi, *Beyond spectral gap: The role of the topology in decentralized learning*, 2022. arXiv: 2206.03093 [cs.LG].

[39] G. Abad, S. Picek, V. J. Ramírez-Durán, and A. Urbieta, "On the security privacy in federated learning", 2022. arXiv: 2112.05423 [cs.CR].

[40] L. Melis, C. Song, E. De Cristofaro, and V. Shmatikov, "Exploiting unintended feature leakage in collaborative learning", May 2019, pp. 691–706. DOI: 10.1109/SP.2019.00029.

[41] C. Dwork, A. Smith, T. Steinke, and J. Ullman, "Exposed! a survey of attacks on private data", *Annual Review of Statistics and Its Application*, vol. 4, Mar. 2017. DOI: 10.1146/annurev-statistics-060116-054123.

[42] I. Dinur and K. Nissim, "Revealing information while preserving privacy", Jun. 2003, pp. 202–210. DOI: 10.1145/773153.773173.

[43] R. Wang, Y. Li, X. Wang, H. Tang, and X.-y. Zhou, "Learning your identity and disease from research papers: Information leaks in genome wide association study", Nov. 2009, pp. 534–544. DOI: 10.1145/1653662.1653726.

[44] A. Suri, P. Kanani, V. Marathe, and D. Peterson, *Subject membership inference attacks in federated learning*, Jun. 2022. DOI: 10.48550/arXiv.2206.03317.

[45] N. Carlini, C. Liu, Ú. Erlingsson, J. Kos, and D. Song, "The secret sharer: Evaluating and testing unintended memorization in neural networks", in *28th USENIX Security Symposium (USENIX Security 19)*, Santa Clara, CA: USENIX Association, Aug. 2019, pp. 267–284, ISBN: 978-1-939133-06-9. [Online]. Available: https://www.usenix.org/conference/usenixsecurity19/presentation/carlini.

[46] C. Song, T. Ristenpart, and V. Shmatikov, *Machine learning models that remember too much*, 2017. arXiv: 1709.07886 [cs.CR].

[47] A. Salem, Y. Zhang, M. Humbert, P. Berrang, M. Fritz, and M. Backes, *Ml-leaks: Model and data independent membership inference attacks and defenses on machine learning models*, 2018. arXiv: 1806.01246 [cs.CR].

[48] S. Rezaei and X. Liu, *On the difficulty of membership inference attacks*, 2021. arXiv: 2005.13702 [cs.LG].

[49] M. Rahman, T. Rahman, R. Laganière, and N. Mohammed, "Membership inference attack against differentially private deep learning model", *Trans. Data Priv.*, vol. 11, pp. 61–79, 2018.

[50] C. Song and V. Shmatikov, *Auditing data provenance in text-generation models*, 2019. arXiv: 1811.00513 [cs.CR].

[51] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis", in *Theory of Cryptography*, S. Halevi and T. Rabin, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 265–284, ISBN: 978-3-540-32732-5. DOI: 10.1007/11681878_14.

[52] M. Nasr, R. Shokri, and A. Houmansadr, "Machine learning with membership privacy using adversarial regularization", 2018. arXiv: 1807.05852 [stat.ML].

[53] M. Abadi, A. Chu, I. Goodfellow, *et al.*, "Deep learning with differential privacy", in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, ACM, Oct. 2016. DOI: 10.1145/2976749.2978318. [Online]. Available: https://doi.org/10.1145/2976749.2978318.

[54] M. Fredrikson, S. Jha, and T. Ristenpart, "Model inversion attacks that exploit confidence information and basic countermeasures", in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '15, Denver, Colorado, USA: Association for Computing Machinery, 2015, pp. 1322–1333, ISBN: 9781450338325. DOI: 10.1145/2810103.2813677. [Online]. Available: https://doi.org/10.1145/2810103.2813677.

[55] K. Ganju, Q. Wang, W. Yang, C. A. Gunter, and N. Borisov, "Property inference attacks on fully connected neural networks using permutation invariant representations", in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '18, Toronto, Canada: Association for Computing Machinery, 2018, pp. 619–633, ISBN: 9781450356930. DOI: 10.1145/3243734.3243834. [Online]. Available: https://doi.org/10.1145/3243734.3243834.

[56] Y. Kaya and T. Dumitras, "When does data augmentation help with membership inference attacks?", in *Proceedings of the 38th International Conference on Machine Learning*, M. Meila and T. Zhang, Eds., ser. Proceedings of Machine Learning Research, vol. 139, PMLR, 18–24 Jul 2021, pp. 5345–5355. [Online]. Available: https://proceedings.mlr.press/v139/kaya21a.html.

[57] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting", *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, Jan. 2014, ISSN: 1532-4435.

[58] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, *Rethinking the inception architecture for computer vision*, 2015. arXiv: 1512.00567 [cs.CV].

[59] D. Li and J. Wang, *Fedmd: Heterogenous federated learning via model distillation*, 2019. arXiv: 1910.03581 [cs.LG].

[60] M. Jere, T. Farnan, and F. Koushanfar, "A taxonomy of attacks on federated learning", *IEEE Security Privacy*, vol. PP, Dec. 2020. DOI: 10.1109/MSEC.2020.3039941.

[61] Z. He, T. Zhang, and R. B. Lee, "Model inversion attacks against collaborative inference", in *Proceedings of the 35th Annual Computer Security Applications Conference*, ser. ACSAC '19, San Juan, Puerto Rico, USA: Association for Computing Machinery, 2019, pp. 148–162, ISBN: 9781450376280. DOI: 10.1145/3359789.3359824. [Online]. Available: https://doi.org/10.1145/3359789.3359824.

[62] S. Mehnaz, N. Li, and E. Bertino, *Black-box model inversion attribute inference attacks on classification models*, 2020. arXiv: 2012.03404 [cs.CR].

[63] F. Wu, "Plfg: A privacy attack method based on gradients for federated learning", Oct. 2020, pp. 191–204, ISBN: 978-981-15-9128-0. DOI: 10.1007/978-981-15-9129-7_14.

[64] Y. Li, Y. Li, H. Xu, and S. Ren, "An adaptive communication-efficient federated learning to resist gradient-based reconstruction attacks", *Secur. Commun. Networks*, vol. 2021, 9919030:1–9919030:16, 2021.

[65] J. Sun, A. Li, B. Wang, H. Yang, H. Li, and Y. Chen, *Provable defense against privacy leakage in federated learning from representation perspective*, 2020. arXiv: 2012.06043 [cs.LG].

[66] A. K. Nair, E. D. Raj, and J. Sahoo, "A robust analysis of adversarial attacks on federated learning environments", *Computer Standards Interfaces*, vol. 86, p. 103 723, 2023, ISSN: 0920-5489. DOI: https://doi.org/10.1016/j.csi.2023.103723. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0920548923000041.

[67] A. Hatamizadeh, H. Yin, P. Molchanov, *et al.*, "Do gradient inversion attacks make federated learning unsafe?", *IEEE Transactions on Medical Imaging*, pp. 1–1, 2023. DOI: 10.1109/tmi.2023.3239391. [Online]. Available: https://doi.org/10.48550/arXiv.2202.06924.

[68] M. P. M. Parisot, B. Pejo, and D. Spagnuelo, *Property inference attacks on convolutional neural networks: Influence and implications of target model's complexity*, 2021. arXiv: 2104.13061 [cs.CR].

[69] L. Lyu and C. Chen, *A novel attribute reconstruction attack in federated learning*, 2021. arXiv: 2108.06910 [cs.CR].

[70] J. Sun, Y. Yao, W. Gao, J. Xie, and C. Wang, *Defending against reconstruction attack in vertical federated learning*, 2021. arXiv: 2107.09898 [cs.LG].

[71] Y. Wu, X. Wang, W. Susilo, *et al.*, "Mixed-protocol multi-party computation framework towards complex computation tasks with malicious security", *Computer Standards Interfaces*, vol. 80, p. 103 570, 2022, ISSN: 0920-5489. DOI: https://doi.org/10.1016/j.csi.2021.103570. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0920548921000659.

[72] H. Ku, W. Susilo, Y. Zhang, W. Liu, and M. Zhang, "Privacy-preserving federated learning in medical diagnosis with homomorphic re-encryption", *Computer Standards Interfaces*, vol. 80, p. 103 583, 2022, ISSN: 0920-5489. DOI: https://doi.org/10.1016/j.csi.2021.103583. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0920548921000787.

[73] H. Fang and Q. Qian, "Privacy preserving machine learning with homomorphic encryption and federated learning", *Future Internet*, vol. 13, no. 4, 2021, ISSN: 1999-5903. DOI: 10.3390/fi13040094. [Online]. Available: https://www.mdpi.com/1999-5903/13/4/94.

[74] J. Hayes, L. Melis, G. Danezis, and E. D. Cristofaro, *Logan: Membership inference attacks against generative models*, 2018. arXiv: 1705.07663 [cs.CR].

[75] B. Hitaj, G. Ateniese, and F. Perez-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*, ser. CCS '17, Dallas, Texas, USA: Association for Computing Machinery, 2017, pp. 603–618, ISBN: 9781450349468. DOI: 10.1145/3133956.3134012. [Online]. Available: https://doi.org/10.1145/3133956.3134012.

[76] Y. Sun, N. S. T. Chong, and H. Ochiai, "Information stealing in federated learning systems based on generative adversarial networks", in *2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 2021, pp. 2749–2754. DOI: 10.1109/SMC52423.2021.9658652.

[77] Z. Wang, M. Song, Z. Zhang, Y. Song, Q. Wang, and H. Qi, "Beyond inferring class representatives: User-level privacy leakage from federated learning", in *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*, 2019, pp. 2512–2520. DOI: 10.1109/INFOCOM.2019.8737416.

[78] L. Zhu, Z. Liu, and S. Han, *Deep leakage from gradients*, 2019. arXiv: 1906.08935 [cs.LG].

[79] B. Zhao, K. R. Mopuri, and H. Bilen, *Idlg: Improved deep leakage from gradients*, 2020. arXiv: 2001.02610 [cs.LG].

[80] Y. LeCun and C. Cortes, "MNIST handwritten digit database", 2010. [Online]. Available: http://yann.lecun.com/exdb/mnist/.

[81] G. Cohen, S. Afshar, J. Tapson, and A. van Schaik, *Emnist: An extension of mnist to handwritten letters*, 2017. [Online]. Available: http://arxiv.org/abs/1702.05373.

[82] H. Xiao, K. Rasul, and R. Vollgraf, *Fashion-mnist: A novel image dataset for benchmarking machine learning algorithms*, 2017. [Online]. Available: http://arxiv.org/abs/1708.07747.

[83] A. Krizhevsky, "Learning multiple layers of features from tiny images", *University of Toronto*, May 2012.

[84] *Cifar10-cnn using pytorch*, https://www.kaggle.com/code/shadabhussain/cifar-10-cnn-using-pytorch/notebook, Accessed: 2023-05-07.

Abbreviations

AAA	Authentication, Authorization, and Accounting
ML	Machine Learning
FL	Federated Learning
CFL	Centralized Federated Learning
DFL	Decentralized Federated Learning
MIA	Membership Inference Attacks
WB	White-Box
BB	Black-Box
CS	Central Server
DP	Differential Privacy
ASR	Attack Success Rate
AP	Attack Precision
TPR	True Positive Rate
FPR	False Positive Rate
MA	Membership Advantage
F1	F1-Score
AUC-ROC	Area under ROC curve

List of Figures

2.1 CFL-Star Topology 9

2.2 DFL: Ring 9

2.3 DFL: Fully-Connected 9

4.1 Raw MNIST Images 25

4.2 Raw EMNIST Images 25

4.3 Raw FashionMNIST Images 26

4.4 Raw CIFAR-10 Images 26

5.1 MNIST: ROC value comparison of White-Box and Black-Box attack on ML model 34

5.2 MNIST: ROC value comparison of White-Box and Black-Box attack on CFL model 35

5.3 FashionMNIST: ROC value comparison of White-Box and Black-Box attack on ML model 35

5.4 FashionMNIST: ROC value comparison of White-Box and Black-Box attack on CFL model 36

5.5 CIFAR-10: ROC value comparison of White-Box and Black-Box attack on ML model 36

5.6 CIFAR-10: ROC value comparison of White-Box and Black-Box attack on CFL model 37

5.7 CIFAR-10: Signal Histogram comparison of White-Box and Black-Box attack on CFL model 37

5.8 MNIST: ROC value comparison of Shadow Metric attack on CFL model . 38

5.9 MNIST: ROC value comparison of White-Box and Black-Box attack on DFL model, Topology Fully Connected 38

5.10 MNIST: ROC and Signal-Histogram Graphs of Shadow Metric attack on DFL model, Topology Fully Connected 39

5.11 FashionMNIST: ROC and Signal-Histogram Graphs of Shadow Metric attack on CFL model 39

5.12 FashionMNIST: ROC value comparison of White-Box and Black-Box attack on DFL model, Topology Fully Connected 40

5.13 FashionMNIST: ROC and Signal-Histogram Graphs of Shadow Metric attack on DFL model, Topology Fully Connected 40

5.14 CIFAR-10: ROC and Signal-Histogram Graphs of Shadow Metric attack on CFL model 41

5.15 CIFAR-10: ROC value comparison of White-Box and Black-Box attack on DFL model, Topology Fully Connected 41

5.16 CFIAR-10: ROC and Signal-Histogram Graphs of Shadow Metric attack on DFL model, Topology Fully Connected 42

5.17 MNIST: ROC value comparison of White-Box and Black-Box attack on DFL model, Topology Ring 42

5.18 MNIST: ROC and Signal-Histogram Graphs of Shadow Metric attack on DFL model, Topology: Ring 43

5.19 FashionMNIST: ROC value comparison of White-Box and Black-Box attack on DFL model, Topology: Ring 43

5.20 FashionMNIST: ROC and Signal-Histogram Graphs of Shadow Metric attack on DFL model, Topology: Ring 44

5.21 CIFAR-10: ROC value comparison of White-Box and Black-Box attack on DFL model, Topology: Ring 44

5.22 CFIAR-10: ROC and Signal-Histogram Graphs of Shadow Metric attack on DFL model, Topology: Ring 45

List of Tables

3.1 Comparison of different Inference Attack Families 22

5.1 Training and MIA Performance on MNIST Dataset 33

5.2 Training and MIA Performance on FashionMNIST Dataset 34

5.3 Training and MIA Performance on CIFAR-10 Dataset 34

Appendix A

Installation Guidelines

For successful installation of the open-source framework Privacy Meter, follow the guidelines and installation manual as elaborated on the Github repository under `https://github.com/privacytrustlab/ml_privacy_meter`

For successful installation of the Fedstellar framework, the installation guideline provided by the owner has to be followed, as this is only available to users with admin rights.

Appendix B

Contents of the zip File

This section lists all files and components of the zip file.

- **BA_Filip_Trendafilov.pdf**
- **BA_Filip_Trendafilov.zip**
- **MidtermPresentationFilipTrendafilov.pptx**
- **fedstellar.zip**
- **privacy_meter.zip**
- **MIA-Eval.xlsx**