

Active Inference against Federated Learning: Attacks and Solutions

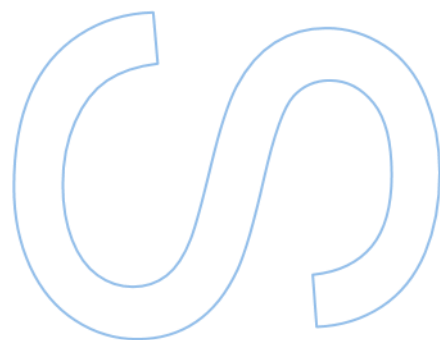
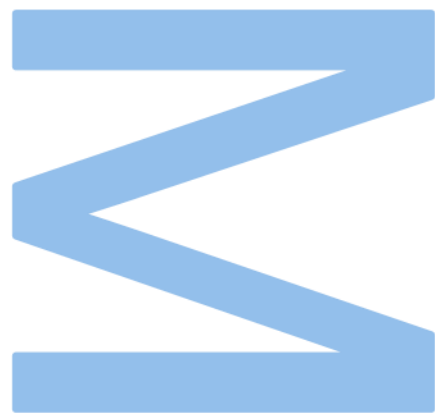
Ana Catarina Gomes

Master in Data Science

Department of Computer Science

Faculty of Sciences of the University of Porto

2024



Active Inference against Federated Learning: Attacks and Solutions

Ana Catarina Gomes

Dissertation/Internship/Project Report carried out as part of
the Master in Data Science
Department of Computer Science
2024

Supervisor

João P. Vilela, Assistant professor, Faculty of Sciences of the
University of Porto

Co-supervisor

Ricardo Mendes, Senior Privacy Technologist, Huawei
Technologies Düsseldorf GmbH, Germany

Abstract

Artificial Intelligence (AI) is revolutionizing various industries, but the reliance on vast datasets for model training raises privacy concerns, especially in distributed learning mechanisms like Federated Learning (FL), where data is decentralized across multiple devices and periodic gradient updates of model parameters are shared to collaboratively train a model. Past research showcases the potential privacy leakage that advents from memorization and overfitting, which is exploited in passive attacks over FL. However, limited attention has been given into understanding and defending against active threats that are conducted during training rounds of FL systems.

This thesis addresses this gap by introducing and analyzing novel active attacks that encode membership and sensitive attribute information by making any targeted training sample leave a distinguishable footprint on the gradient of maliciously modified neurons (model parameters). These attacks are assessed using a real-world use-case with the COP-MODE dataset [1], which consists of mobile device data. These novel active attacks, Active Membership Inference (AMI) and Active Attribute Inference (AAI), demonstrate that even well-generalized FL models are vulnerable to these threats. Results show that it is possible to successfully encode sensitive information incurring in a small error in terms of neuron activation. Inference on a practical scenario reveals that AMI is able to identify a training sample 92.6% of the times, by expecting the neighbourhood of training samples to be better represented than that of non training samples. In its turn, AAI^{*} is able to determine that the user is not at home in 85.8% of the times. This work further explores the connection between MIA and AIA by developing an Attribute Inference based on Membership Inference (MIA2AIA) that explores the correlation between membership and attribute risk both in theory and practice, showing that membership risk is an upper bound to the attribute inference risk.

The study of the impact of active attacks on model utility shows that model performance is partially recoverable after a few training rounds, demonstrating that active attacks may occur in FL systems without major long-term model utility loss, which enhances the real threat of these attacks, since their negative impact soon becomes undetectable. To address these active threats, this work introduces, for the first time, several detection strategies to mitigate privacy attacks, including a novel White-box Attack Detection and Mitigation (WADM) and two black-box strategies: Black-box Attack Detection based on Accuracy (BADAcc) and Black-box Attack Detection based on ROC AUC (BADAUC). These strategies detect abnormal changes in weights distribution and model performance, respectively. Results show that these strategies are robust

to attempts to evade detection, effectively mitigating AMI and AAI*. Remarkably, WADM* mitigates novel active attacks while preserving model utility, offering significant improvements over existing defenses that do not mitigate these attacks.

By proposing novel active attacks and developing effective countermeasures, this research contributes for an enhancement of privacy of FL systems, ensuring that users do not need to trade privacy for model utility, as they can apply WADM* to protect their private information from an adversary conducting AMI or AAI* while still being able to use the system with minimal impact on the training process.

Keywords— Federated Learning, Data Privacy, Membership Inference Attack, Attribute Inference Attack, Attack Detection

Resumo

A Inteligência Artificial (IA) está a revolucionar várias indústrias, mas a dependência de vastos conjuntos de dados para o treino de modelos levanta preocupações de privacidade, especialmente em mecanismos de aprendizagem distribuída, como Federated Learning (FL), nos quais os dados estão distribuídos por vários dispositivos que periodicamente enviam updates dos parametros dos modelos para treinar modelos de forma colaborativa. A investigação anterior demonstra o potencial de privacy leakage que advém da memorização e overfitting, o que é explorado através de ataques passivos sobre modelos de FL. No entanto, pouca atenção tem sido dada à compreensão e defesa contra ameaças ativas que são feitas durante as rondas de treino em sistemas de FL.

Esta tese aborda essa lacuna, introduzindo e analisando novos ataques ativos que codificam informações de membership e de atributos sensíveis, fazendo com que qualquer amostra de treino que seja alvo destes ataques deixe uma marca distinta no gradiente de neurónios (parametros do modelo) maliciosamente modificados. Estes ataques são avaliados usando um caso de uso real com o conjunto de dados COP-MODE [1], que consiste em dados de dispositivos móveis. Estes novos ataques ativos, Active Membership Inference (AMI) e Active Attribute Inference (AAI), demonstram que mesmo modelos de FL com boa capacidade de generalização são vulneráveis a estas ameaças. Os resultados mostram que é possível codificar com sucesso informações sensíveis incorrendo num pequeno erro em termos de ativação do neurónio. A inferência num cenário prático revela que o AMI é capaz de identificar uma amostra de treino 92.6% das vezes, quando se espera que a vizinhança das amostras de treino esteja melhor representada do que a das amostras que não foram usadas para treino. Por sua vez, o AAI* é capaz de determinar que o utilizador não está em casa em 85.8% das vezes. Este trabalho explora ainda a relação entre Membership Inference Attack (MIA) e Attribute Inference Attack (AIA), desenvolvendo um Attribute Inference based on Membership Inference (MIA2AIA) que explora a relação entre o risco de membership e o de attribute inference, tanto na teoria como na prática, mostrando que o risco de membership é um limite superior ao risco de attribute inference.

O estudo do impacto de ataques ativos na utilidade do modelo mostra que o desempenho do modelo é parcialmente recuperável após algumas rondas de treino, demonstrando que os ataques ativos podem ocorrer em sistemas de FL sem uma perda significativa de utilidade do modelo a longo prazo, o que realça a ameaça real destes ataques, já que o seu impacto negativo torna-se rapidamente indetetável. Para enfrentar estas ameaças ativas, este trabalho introduz, pela

primeira vez, várias estratégias de detecção para mitigar ataques de privacidade, incluindo um novo mecanismo White-box Attack Detection and Mitigation (**WADM**) e duas estratégias black-box: Black-box Attack Detection based on Accuracy (**BADAcc**) e Black-box Attack Detection based on ROC AUC (**BADAUC**). Estas estratégias detetam mudanças anormais na distribuição de pesos e no desempenho do modelo, respetivamente. Os resultados mostram que estas estratégias são robustas a tentativas de evasão da detecção, mitigando de forma eficaz o AMI e o AAI*. Notavelmente, o WADM mitiga estes novos ataques ativos enquanto preserva a utilidade do modelo, oferecendo melhorias significativas em relação às defesas existentes que não mitigam estes ataques.

Ao propor novos ataques ativos e desenvolver defesas eficazes, esta investigação contribui para a melhoria da privacidade dos sistemas de FL, garantindo que os utilizadores não precisam de trocar privacidade por utilidade do modelo, uma vez que podem aplicar o WADM* para proteger a sua informação privada de um adversário que aplique o AMI ou o AAI*, enquanto continuam a utilizar o sistema com impacto reduzido no processo de treino.

Palavras-chave— Federated Learning, Privacidade de Dados, Membership Inference Attack, Attribute Inference Attack, Detecção de Ataques

Acknowledgements

Words cannot express my gratitude to my supervisors, Professor João Vilela and Doctor Ricardo Mendes, for their constant guidance and support. Their honest and constructive feedback has provided me knowledge that I could not have learned on my own nor from anyone else. I would also like to extend my sincere thanks to my partner, Santiago Azevedo, for his editing help, technical support, and feedback. Lastly, I would be remiss in not mentioning my family, partner, and friends, especially for their moral support. Their belief in me has been my source of strength on the hardest times.

Contents

Abstract	i
Resumo	iii
Acknowledgements	v
Contents	x
List of Tables	xi
List of Figures	xvii
Acronyms	xix
1 Introduction	1
1.1 Main contributions	2
1.2 Structure	3
2 State of the Art	5
2.1 Federated Learning	5
2.2 Privacy Attacks on Machine Learning Models	7
2.2.1 Overview	7
2.2.2 Goal of Attacks on ML	8
2.2.3 Target of Attacks on ML	17
2.2.4 Adversary of ML models	18

2.2.5	Taxonomy of Privacy Attacks on ML	24
2.3	Defense Mechanisms	24
2.3.1	Confidence Score Masking	26
2.3.2	Regularization	26
2.3.3	Adversarial Training	27
2.3.4	Perturbation Techniques	27
2.3.5	Gradient Pruning	28
2.3.6	Differential Privacy	28
2.3.7	Knowledge Distillation	29
2.3.8	Secure Multi-party Computation	30
2.3.9	Attack Detection	31
2.3.10	Towards Local Defense Mechanisms	32
2.4	Attack Performance Metrics	33
2.4.1	Measuring MIA performance	33
2.4.2	Measuring AIA performance	34
2.4.3	Measuring PIA performance	35
2.4.4	Measuring RA performance	36
2.5	Summary	36
3	Membership and Attribute Inference	41
3.1	Novel Active Inference Attacks	41
3.1.1	Overview	42
3.1.2	Passive Membership Inference	42
3.1.3	Active Membership Inference (AMI)	46
3.1.4	Adapting AMI as an Active Attribute Inference (AAI [∗])	48
3.1.5	Attribute Inference based on Membership Inference (MIA2AIA)	51
3.2	Evaluation and Analysis	53
3.2.1	Experimental Setup	53

3.2.2	Passive Membership Inference Analysis	57
3.2.3	Active Membership Inference Analysis	60
3.2.4	Active Attribute Inference Analysis	65
3.2.5	Model Recovery from Active Attacks Analysis	73
3.2.6	Making inference attacks more robust under gradient uncertainty	76
3.2.7	Connection between membership and attribute inference	81
3.3	Summary	83
4	Defense Mechanisms	85
4.1	Overview	86
4.2	Black-box Attack Detection	87
4.2.1	Black-box Attack Detection based on Accuracy (BADAcc)	88
4.2.2	Black-box Attack Detection based on ROC AUC (BADAUC)	88
4.3	White-box Attack Detection and Mitigation (WADM*)	90
4.4	Analysis	93
4.4.1	Experimental Setup	93
4.4.2	Black-box Attack Detection based on Accuracy	96
4.4.3	Black-box Attack Detection based on ROC AUC	102
4.4.4	White-box Attack Detection and Mitigation Analysis	109
4.5	Discussion and Summary	119
5	Conclusion	123
5.1	Active Attacks	124
5.2	Defenses	125
5.3	Limitations and future research	126
A	Subjective evaluation of SOTA	131
B	Theory behind novel active attacks	133

C	Target model recovery after AMI	137
D	Experiments with other datasets	139
D.1	CelebA	139
D.2	AMI	141
D.3	AAI* and MIA2AIA	141
D.4	Conclusion	143
	Bibliography	145

List of Tables

2.1	Summary of privacy attacks on ML	37
3.1	Model performance (represented by 90% confidence intervals of ROC AUC) of federated models with ReLU and ELU as activation function ($\alpha = -2$), used for AMI plus MIA2AIA and AAI, respectively.	56
3.2	Model performance (represented by 90% confidence intervals of ROC AUC) of centralized models with ReLU and ELU as activation function ($\alpha = -2$), used for AMI plus MIA2AIA and AAI, respectively.	56
A.1	Evaluation of SOTA	132

List of Figures

2.1	FedAVG algorithm [37], where C is the fraction of clients per training round, P_k is the private dataset of the k -th client and n_k its size. Successfully training a federated model requires tuning hyperparameters, such as C , η , and E	6
2.2	Overview of entities involved in privacy attacks and their interaction	8
2.3	Taxonomy of privacy attacks - diagram.	25
2.4	Application of secure aggregation on FL systems	30
2.5	State of the art attack and defense evaluation per privacy attack. Scores in the range $[0, 1]$ express the average grade per privacy attack, whereas the number of papers contributing to each score is inside parenthesis in the figure caption. . . .	38
3.1	Overview of novel active attacks performed by the central server on a FL system.	43
3.2	ROC AUC of passive metric and model-based MIAs on centralized and federated models of several complexities.	58
3.3	Gradient norm distribution of the shadow dataset after the first and last training rounds, based on a standard complexity shadow model	59
3.4	Gradient norm distribution of the target dataset after the first and last training rounds, based on a standard complexity target model	59
3.5	Multi-target AMI TPR and FPR on federated models (standard complexity). . .	61
3.6	Multi-target AMI TPR and FPR on federated models of multiple complexities. .	62
3.7	Multi-target AMI TPR and FPR on centralized and federated models (standard complexity).	63
3.8	Single-target AMI TPR and FPR under adversarial limited knowledge (federated models of standard complexity).	64
3.9	Single-target AAI* and MIA2AIA TPR and FPR on federated models (of all complexities) for several sensitive attributes.	66

3.10	Multi-target AAI* and MIA2AIA TPR and FPR on federated models (standard complexity). Sensitive attribute is <i>seletedSemanticLoc_Home</i>	68
3.11	Multi-target AAI* and MIA2AIA performance on federated models of multiple complexities. Sensitive attribute is <i>seletedSemanticLoc_Home</i>	69
3.12	MULTi-target AAI* and MIA2AIA TPR and FPR on centralized and federated models (standard complexity). Sensitive attribute is <i>seletedSemanticLoc_Home</i>	71
3.13	Single-target AAI* and MIA2AIA performance under adversarial limited knowledge (standard complexity). Sensitive attribute is <i>seletedSemanticLoc_Home</i>	72
3.14	Recovery (ROC AUC evolution) from multi-target AMI on train and test set of models with distinct complexities over 5 training rounds.	74
3.15	Recovery (ROC AUC evolution) from multi-target AAI* on train and test set of models with distinct complexities over 5 training rounds	75
3.16	Gradient uncertainty illustration.	77
4.1	Overview of defense mechanisms applied by each client on a FL system.	87
4.2	Evolution of the weights distribution of a standard neuron, computed from the fifth and sixth training round, once the model (of standard complexity) has converged. Each plot represents a different type of model (trained with ReLU and ELU respectively).	91
4.3	Change in weights distribution of a malicious neuron (after single-target attacks), computed from the fifth and sixth training round, once the model (of standard complexity) has converged. Each plot represents a different type of model (trained with ReLU and ELU respectively).	91
4.4	Multi-target AMI performance on standard (learning rate 0.05, 100 epochs) and restricted setup (learning rate 0.005, 50 epochs), on federated models of multiple complexity and for an increasing percentage of neurons controlled by the attacker	95
4.5	Multi-target AAI* performance on standard (learning rate 0.05, 1000 epochs) and restricted setup (learning rate 0.025, 1000 epochs), on federated models of multiple complexity and for an increasing percentage of neurons controlled by the attacker	96
4.6	BADAcc performance on detecting AMI over federated models of multiple complexities, for distinct adversarial setups, and for an increasing percentage of neurons controlled by the attacker.	97
4.7	BADAcc performance on detecting AAI* conduted on federated models of multiple complexities, for distinct adversarial setups and for an increasing percentage of neurons controlled by the attacker.	98

4.8	Impact of AMI and AAI* on training accuracy on federated models of multiple complexities, for distinct adversarial setups and for an increasing percentage of neurons controlled by the attacker	99
4.9	BADAcc FPR while detecting AMI and AAI* on federated models of multiple complexities.	100
4.10	Correlation matrix of BADAcc results (of AMI and AAI* detection) with local features	101
4.11	BADAUC performance on detecting AMI over federated models of multiple complexities, for distinct adversarial setups, and for an increasing percentage of neurons controlled by the attacker.	103
4.12	BADAUC performance on detecting AAI* over federated models of multiple complexities, for distinct adversarial setups, and for an increasing percentage of neurons controlled by the attacker.	103
4.13	Impact of AMI and AAI* on training ROC AUC on federated models of multiple complexities, for distinct adversarial setups and for an increasing percentage of neurons controlled by the attacker.	104
4.14	BADAUC performance for varying error costs. Detection of standard setup AMI that controls 3% of neurons over models of standard complexity.	106
4.15	BADAUC performance for varying error costs. Detection of standard setup AAI* that controls 3% of neurons over models of standard complexity.	106
4.16	BADAUC performance for varying thresholds. Detection of standard and restricted setup AMI and AAI* that control 3% of neurons over models of standard complexity.	107
4.17	Correlation matrix of BADAUC results (of AMI and AAI* detection) with local features	108
4.18	WADM* performance on detecting AMI conducted on federated models of multiple complexities, for distinct adversarial setups and for an increasing percentage of neurons controlled by the attacker.	110
4.19	WADM* performance on detecting AAI* conducted on federated models of multiple complexities, for distinct adversarial setups and for an increasing percentage of neurons controlled by the attacker.	111
4.20	AMI performance, after applying WADM*, on federated models of multiple complexities, for distinct adversarial setups and for an increasing percentage of neurons controlled by the attacker.	113

4.21	AAI* performance, after applying WADM*, on federated models of multiple complexities, for distinct adversarial setups and for an increasing percentage of neurons controlled by the attacker.	113
4.22	WADM* false alarms (FPR) evolution along the first benign training rounds of models, used for AMI and AAI*, of multiple complexities.	115
4.23	WADM* performance on detecting AMI over federated models of multiple complexities, for distinct adversarial setups and for an increasing percentage of neurons controlled by the attacker. The threshold was chosen taking into account FPR minimization (0.4722).	116
4.24	WADM* performance on detecting AAI* over federated models of multiple complexities, for distinct adversarial setups and for an increasing percentage of neurons controlled by the attacker. The threshold was chosen taking into account FPR minimization (0.2611).	116
4.25	WADM* performance on mitigating AMI over federated models of multiple complexities, for distinct adversarial setups and for an increasing percentage of neurons controlled by the attacker. The threshold was chosen taking into account FPR minimization (0.4722)	117
4.26	WADM* performance on mitigating AAI* over federated models of multiple complexities, for distinct adversarial setups and for an increasing percentage of neurons controlled by the attacker. The threshold was chosen taking into account FPR minimization (0.2611)	117
4.27	Recovery (ROC AUC evolution) from multi-target AMI (restricted setup) on train and test set of models with distinct complexities over 5 training rounds.	119
4.28	Recovery (ROC AUC evolution) from multi-target AAI* (restricted setup) on train and test set of models with distinct complexities over 5 training rounds. . .	120
B.1	Illustration of the target model (of standard complexity) with annotations. . . .	134
C.1	Recovery (ROC AUC evolution) from multi-target AMI on train and test set of models of standard complexities over 5 training rounds	137
D.1	AMI performance on several well-known datasets from the vision domain (Figure 9, Appendix D [42]).	141
D.2	Single-target AAI* and MIA2AIA TPR and FPR on federated models (on CelebA dataset) for several sensitive attributes. The baseline for each attribute is inside parenthesis.	142

D.3	Single-target AAI* and MIA2AIA ROC AUC % (restricted to a maximum FPR of 25%) under gradient uncertainty, over federated models on celebA dataset. Only two sensitive attribute values are presented, one with a balanced and another one with an imbalanced distribution.	143
-----	--	-----

Acronyms

AAI	Active Attribute Inference	GAN	Generative Adversarial Network
AI	Artificial Intellegence	GDPR	General Data Protection Regulation
AIA	Attribute Inference Attack	KD	Knowledge Distillation
AMI	Active Membership Inference	KL-divergence	Kullback-Leibler Divergence
API	Applicaiton Programming Interface	LDP	Local Differential Privacy
BADAcc	Black-box Attack Detection based on Accuracy	LRD	Local Reachability Density
BADAUC	Black-box Attack Detection based on ROC AUC	LSTM	Long Short-Term Memory
BBL	Black-box Loss-based	MI	Model Inversion
CNN	Convolutional Neural Network	MIA	Membership Inference Attack
DP	Differential Privacy	MIA2AIA	Attribute Inference based on Membership Inference
DPSGD	Differentially Private Stochastic Gradient Descent	ML	Machine Learning
ELU	Exponential Linear Units	NN	Neural Network
FC	Fully Connected	PIA	Property Inference Attack
FL	Federated Learning	RA	Reconstruction Attack
FNR	False Negative Rate	ReLU	Rectified Linear Units
FP	False Positive	REPR	Class Representative Inference Attacks
FPR	False Positive Rate	RNN	Recurrent Neural Network

ROC AUC	Area Under the Receiver Operating Characteristic Curve	TPR	True Positive Rate
		WADM	White-box Attack Detection and Mitigation
SGD	Stochastic Gradient Descent	WBG	White-box Gradient-norm-based
SMC	Secure Multi-party Computation	WBIG	White-box Improved Gradient-norm-based
TP	True Positive		

Chapter 1

Introduction

Artificial Intelligence (**AI**) is leading a technological paradigm shift that impacts several application fields. Powered by sophisticated machine learning models, AI methods find applications in diverse fields, ranging from healthcare and finance to transportation and entertainment. These models, driven by data, hold immense potential for innovation and advancement.

However, as AI continues to advance, concerns about data privacy have come to the forefront. The reliance of these models on vast datasets for training poses significant challenges to data privacy. This concern is especially pertinent in centralized learning, whereby all data is held by a single entity. However, privacy concerns also exist even in distributed machine learning models like Federated Learning (**FL**), where data is decentralized across multiple devices and periodic updates are shared to collaboratively train a model (global model). Clients participating in a federated learning scheme are exposed to privacy attacks not only from the central server, but also from any other client.

Privacy attacks, such as membership and attribute inference attacks, try to extract information that was not intended to be shared. Such knowledge can be about particular training samples or about properties of the training data such as unintentionally encoded biases (Rigaki et al. [46]). In a Membership Inference Attack (**MIA**), an attacker with access to the trained model attempts to infer whether a given sample was used in the training process. Despite its simple goal, MIA can raise *severe privacy risks* to individuals. In addition to the potential violation of data regulations (e.g. General Data Protection Regulation (**GDPR**) [2]), highly accurate MIAs are also useful to demonstrate privacy leakage [55]. In an Attribute Inference Attack (**AIA**), an attacker with access to the trained model attempts to infer a training sample's sensitive attribute value (Rigaki et al. [46]). Even though the sensitiveness of an attribute can be subjective, the threat in terms of data privacy becomes clear with an example: the inference of gender or race. Although many individuals do not consider this information as sensitive, it may be highly sensitive depending on the context, as these attributes are often associated with discrimination. The level of threat of AIAs ultimately depends on the destination of such inferred information, which can be leveraged, for example, to deliver personalized advertisements to users (Jia et al. [29]).

Past research primarily focused on passive attacks or unrealistic scenarios, leaving a *gap in understanding and defending against active attacks*. These active threats, which manipulate models to extract sensitive information, pose a significant threat to FL systems, which do not require a model to be more or less accurate or overfitted (unlike passive attacks). Instead, it is only required that the model is sufficiently complex. Therefore, state-of-the-art defenses, such as differential privacy and regularization techniques, do not mitigate these attacks. In fact, given that active privacy attacks are underexplored, so are defense mechanisms to mitigate them, namely defenses that balance the trade-off between privacy and model utility, such as attack detection.

This work aims to address this gap by focusing on active and realistic membership and attribute inference attacks over FL models. These novel attacks do not rely on overfitting, but increase model leakage by encoding membership and attribute information on model weights. With these attacks, the ultimate goal of this work is to identify vulnerabilities of well-generalized distributed learning models, that often carry a false sense of security. Furthermore, by exploring the connection between membership and attribute inference attacks, the goal is to show that membership risk is an upper bound for attribute risk. Additionally, since these attacks modify the global model, a novel defense mechanism is developed to mitigate them while minimizing model utility loss, which resorts to distance-based techniques to detect an ongoing malicious interference on the global model, therefore addressing the gap of privacy attack detection strategies.

In essence, this research highlights the importance of privacy of federated models and the urgent need for robust defense strategies. By understanding the risks posed by active attacks and developing effective countermeasures, this work aims to contribute for an *improvement of privacy-preserving FL systems*, ensuring that users do not need to trade privacy for model utility.

1.1 Main contributions

Aiming to improve privacy-preserving FL systems by proposing and defending against novel active attacks, the contributions of this thesis are the following:

- A systematic literature review and a taxonomy of several privacy attacks over machine learning models and defense mechanisms, an overview of metrics to measure the performance of these attacks, and an assessment of attacks and defenses through a graded evaluation according to various parameters to identify open challenges.
- A qualitative and quantitative analysis of an Active Membership Inference (**AMI**) that outperforms passive attacks over federated learning models to show that well-generalized models are vulnerable to membership inference attacks.
- A qualitative and quantitative analysis of three passive MIAs, Black-box Loss-based (**BBL**), White-box Gradient-norm-based (**WBG**), White-box Improved Gradient-norm-

based (**WBIG**), to serve as baseline for AMI and to show that passive attacks are not effective against well-generalized models.

- The development and evaluation of a novel active attribute inference attack, Active Attribute Inference (**AAI**) that is an adaptation of AMI, to increase model leakage with respect to a sensitive attribute.
- A qualitative and quantitative analysis of the most common state-of-the-art approach for attribute inference, Attribute Inference based on Membership Inference (**MIA2AIA**), to serve as baseline for **AAI**^{*}.
- The development and evaluation of two novel black-box attack detection algorithms, Black-box Attack Detection based on Accuracy (**BADAcc**) and Black-box Attack Detection based on ROC AUC (**BADAUC**), to detect ongoing active attacks based on an abnormal drop in local accuracy or Area Under the Receiver Operating Characteristic Curve (**ROC AUC**). These are inspired in drift and anomaly detection algorithms, yet they are, for the first time, used in this context.
- The development and evaluation of a White-box Attack Detection and Mitigation (**WADM**), that detects maliciously modified neurons based on an abnormal change in the weights distribution of neurons, followed by a mitigation phase to, not only reduce adversarial advantage, but also model utility loss. This approach is developed to mitigate novel active attacks proposed in this work, unlike state-of-the-art defenses which are ineffective.

Note that novel approaches, for the first time proposed and developed in this work, appear with a small star on the top to help distinguish them from state-of-the-art approaches, as well as new attacks and defenses based on state-of-the-art approaches.

1.2 Structure

This thesis is structured as follows:

Chapter 1: Introduction: Brief description of the problem, its motivation, previous efforts to tackle it, as well as the key innovation factor of this work.

Chapter 2 - State-of-the-Art: Description and critical view of the state-of-the-art approaches to the same problem.

Chapter 3 - Membership and Attribute Inference: Development, qualitative and quantitative analysis of novel membership and attribute inference attacks, and a comparison to state-of-the-art baselines.

Chapter 4 - Defense Mechanisms: Development, qualitative and quantitative analysis of novel attack detection mechanisms.

Chapter 5 - Conclusion: Summary of the main findings and suggestions of future work.

Chapter 2

State of the Art

In this chapter, the reader can find some theoretical background to understand the following content and a review of the state of the art about Machine Learning (ML) privacy, namely privacy attacks and defense mechanisms. It is assumed that the reader knows some basic concepts about machine learning, particularly about deep learning, although a few footnotes briefly explain some concepts. Background concepts that are not assumed to be fully known and understood are mainly related to machine learning privacy. A systematic review of prior work about privacy attacks aims to convince the reader about how dangerous and, sometimes, underestimated these threats are, particularly for the federated learning setting. A review of most common defense mechanisms should highlight the need for approaches with a better trade-off between privacy and model utility. Following this introduction, federated learning is explained in Section 2.1, privacy attacks on machine learning models are explained in Section 2.2, defense mechanisms in Section 2.3, and metrics to measure attack performance in Section 2.4. A critical analysis of the state of the art of privacy attacks and defense mechanisms, according to various parameters, is presented in Section 2.5.

2.1 Federated Learning

Modern mobile devices have access to a wealth of data suitable for learning models, which in turn can greatly improve user experience. However, this rich data is often privacy sensitive, large in quantity, or both, which may preclude logging to the data center and training there using conventional approaches (McMahan et al. [37]). This was the motivation that led Google to propose Federated Learning (FL) in 2016 - an alternative to centralized training that leaves private training data distributed on mobile devices and trains a shared model by aggregating locally computed updates.

Federated learning involves a central server (service provider) which aggregates each participant's (also referred as client) contribution. This framework can be categorized based on data partitioning (Yin et al. [63]) and aggregation strategy. Regarding data partitioning, Horizontal

Algorithm 1 *FederatedAveraging*. The K clients are indexed by k ; B is the local minibatch size, E is the number of local epochs, and η is the learning rate.

Server executes:

```

initialize  $w_0$ 
for each round  $t = 1, 2, \dots$  do
   $m \leftarrow \max(C \cdot K, 1)$ 
   $S_t \leftarrow$  (random set of  $m$  clients)
  for each client  $k \in S_t$  in parallel do
     $w_{t+1}^k \leftarrow \text{ClientUpdate}(k, w_t)$ 
   $m_t \leftarrow \sum_{k \in S_t} n_k$ 
   $w_{t+1} \leftarrow \sum_{k \in S_t} \frac{n_k}{m_t} w_{t+1}^k$  // Erratum4

```

```

ClientUpdate( $k, w$ ): // Run on client  $k$ 
 $\mathcal{B} \leftarrow$  (split  $\mathcal{P}_k$  into batches of size  $B$ )
for each local epoch  $i$  from 1 to  $E$  do
  for batch  $b \in \mathcal{B}$  do
     $w \leftarrow w - \eta \nabla \ell(w; b)$ 
  return  $w$  to server

```

Figure 2.1: FedAVG algorithm [37], where C is the fraction of clients per training round, P_k is the private dataset of the k -th client and n_k its size. Successfully training a federated model requires tuning hyperparameters, such as C , η , and E .

FL requires private datasets to have an identical feature space but different identity spaces, meaning distinct instances described by the same set of features. Vertical FL requires private datasets to have an identical identity space but different feature space, meaning the same identity is described by a distinct set of attributes across participants. Federated Transfer Learning is employed in a scenario in which private datasets have distinct identity and feature space. Throughout the rest of this document, only Horizontal FL will be considered, referred simply as FL.

Regarding the aggregation strategy, *FedSGD* and *FedAVG* (McMahan et al. [37]) are of special interest. Stochastic gradient descent (SGD) can be applied naively to a federated optimization problem, where a single batch gradient calculation on a randomly selected client is done per round of communication. Although being computationally efficient, it would require a large number of training rounds. Thus, instead of a single batch gradient, *FedSGD* requires a client to send its gradient updates with respect to its global dataset. *FedAVG* is a generalization of *FedSGD* to account for several local training iterations. Each client trains the global model locally and sends the difference between its updated copy and the original model to the server. Figure 2.1 formalizes this algorithm.

Among its wide range of applications, FL can be used to train a recurrent neural network language model for next-word prediction in a virtual keyboard for smartphones (Hard et al. [23]). Server-based training using stochastic gradient descent is compared with training on client devices using the *FederatedAveraging* algorithm (*FedAVG*). The federated algorithm, which enables training on a higher-quality dataset, is shown to achieve better prediction recall compared to other server-trained approaches. This demonstrates the feasibility and benefit of training

models on client devices without exporting sensitive user data to servers.

2.2 Privacy Attacks on Machine Learning Models

Attacks on machine learning models encompass a range of strategies to exploit vulnerabilities and compromise data privacy. These may be categorized according to its goal (Rigaki et al [46]), target, and adversary (Yin et al. [63], Hu et al. [25]). Regarding the goal of privacy attacks, one can have membership, attribute, property, and class representatives inference, as well as model inversion. Regarding the target of attacks, one can describe it by the type of target model, and in which target model's phase the attack occurs. Regarding the adversary, one can describe it according to its identity/role in the target model, its access level, its effect/impact (if any), and the overall approach that must be tailored according, not only to its goal, but to its knowledge. First, an overview of how these attacks may occur is presented on this section (2.2.1), and the remainder details privacy attacks on ML models under each of these aspects.

2.2.1 Overview

Before diving into more details, this overview of privacy attacks aims to introduce some concepts used throughout this document. Figure 2.2 illustrates entities involved in privacy attacks on ML models, as well as their interaction (referred as attack mode). First, for an attack to occur a victim must be defined, in this case, the victim is a predictive system (as a whole, made available to users, or just part of it, i.e. a client participating on a FL system). This victim owns private data (Train and Test dataset in the figure) that uses to train and test the target model. With the goal of inferring sensitive information about the victim, the adversary trains a shadow model (to mimic the target model behaviour) with the shadow dataset to then make inference about the target dataset. This target dataset most likely has train and test samples (from the victim's private data) about which the adversary is interested in inferring sensitive information, for example which of those instances was used during training. Note that not all adversaries train a shadow model, yet it is a sufficiently common practice to include on this overview.

To conduct the attack, the adversary may only have query access to the Target Model, that is only the output vector is accessible. This is the most limited type of access that an adversary can have and it is common when, for example, the pre-trained target model is released as an Application Programming Interface (API) for predictive purposes only. In these cases, the attack mode is Passive, as the adversary behaves as an apparently normal user, most known as *honest but curious*, since it complies with the protocol but tries to infer as much information as it can. On the other hand, the adversary may interfere in the target model's training process, when it is a participant in a FL system, therefore having privileged access to the Target Model. In these cases, the attack mode is Active, meaning that the adversary tries to make the Target Model leak the desired information, by sending malicious weights for example.

Throughout the rest of this section, further details regarding both entities (victim and adversary), as well as different attack modes, will be explained in order to categorize each type of privacy attack.

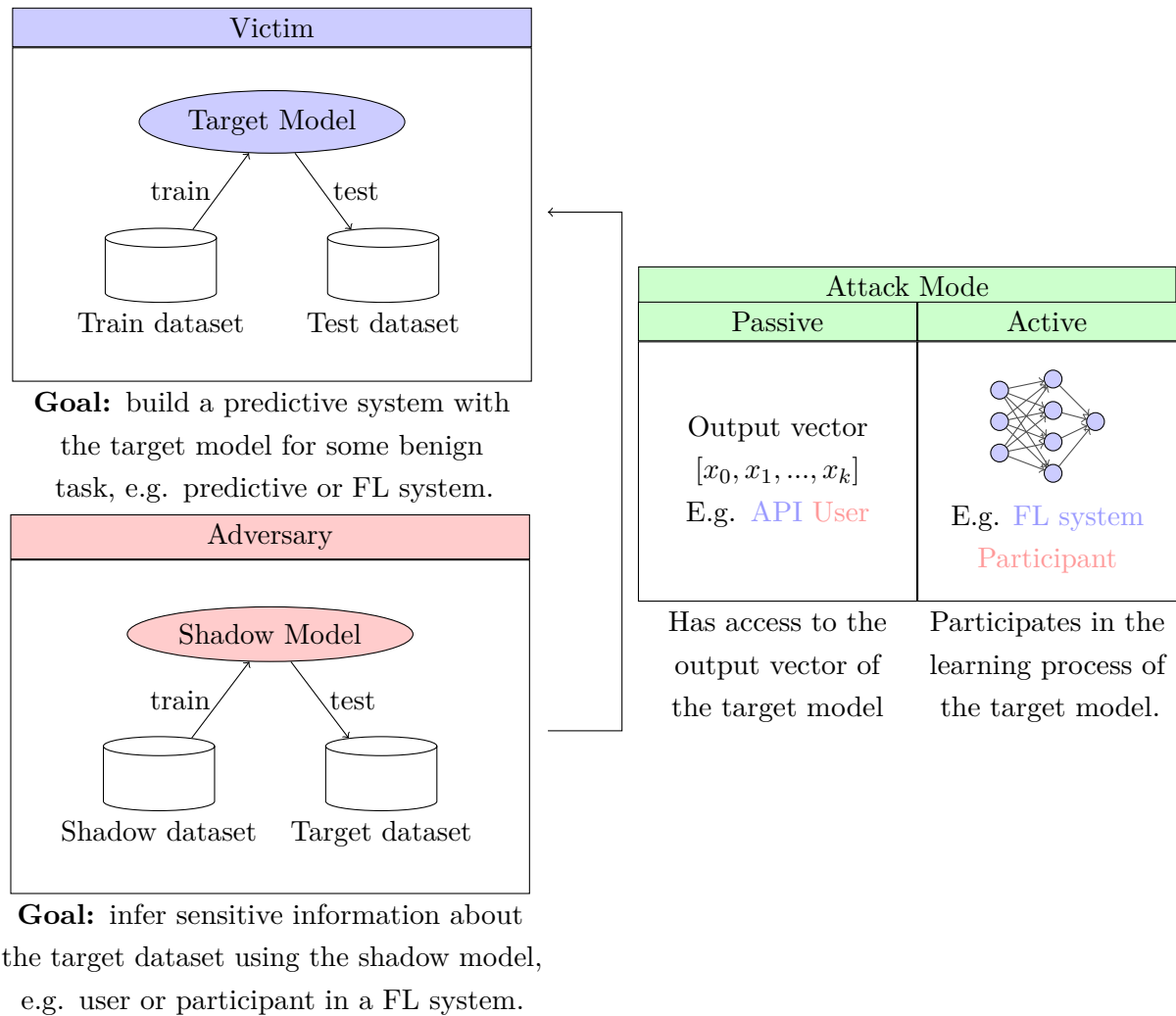


Figure 2.2: Overview of entities involved in privacy attacks and their interaction

2.2.2 Goal of Attacks on ML

An adversary may have distinct goals, all of which constitute a violation to the privacy of data owners, leveraged upon a released model or the adversary's participation in a model training system. Throughout the rest of this section, the following adversarial goals will be explored: membership, attribute and property inference, as well as data reconstruction.

2.2.2.1 Membership Inference

In a Membership Inference Attack (**MIA**), an attacker with access to the trained model attempts to infer whether a given sample was used in the training process (Rigaki et al [46]), those samples are called members. Despite its simple goal, MIA can raise *severe privacy risks* to individuals. For example, by identifying that a clinical record has been used to train a model for assisting with diagnoses, the attacker can infer that the individual has some disease with high probability. In fact, just knowing about the presence of an individual in a medical dataset can be a violation of both health and personal data regulations (e.g General Data Protection Regulation (**GDPR**) [2]). Highly accurate MIAs would be useful not only to demonstrate privacy leakage, but for other applications like detecting train/test contamination and verifying if copyrighted data was used to train a given model (Wang et al. [55]). The intuition of these attacks is that some records, when used for training, have unique influences on the target model, even when the model does not overfit ¹ the data.

A major distinction between different MIAs is that some aim to infer a sample’s membership status based on the fact that members’ predictions are directly optimized (up to a point) and non-members’ predictions hopefully follow the same trend, such as several reviewed papers [7, 10, 12, 20, 26, 40, 44, 45, 48, 49, 52, 58, 61, 64]. These attacks exploit the *difference between evidence (direct optimization) and hope (performance on unseen data)*. Hence, overfitting is pointed out as the main reason for membership vulnerability (Yeom et al. [62]). In this case, a high-risk sample is one that has a significantly different output if the model has previously seen it during training. Although some MIAs look for these samples in the lower tail of the target dataset’s loss distribution, this is not always true because small values of loss can either be achieved by feeding that point to the model during training or by naturally easier to classify data points. Analogously, the higher tail of loss’s distribution can have non-members or members that are naturally harder to classify. This insight was previously explored by Carlini et al. [7].

Contrarily to what has been claimed in most of the literature, models do not need to completely overfit training data to be vulnerable to these MIAs, at least considering standard overfitting definition given by a sufficiently large difference between train and test performance. Instead, a model only needs to be locally overfitted to leak membership information about high-risk samples, as suggested by Choquette-Choo et al. [12] and Carlini et al. [7]. Meaning, poorly represented areas from the whole input space that incur in larger generalization error ², due to the target model’s incapacity to learn how to classify that part without memorizing. Thus, membership risk is greater in those areas. In that case, MIA’s threat depends on how much of the meaningful input space is not mapped confidently. On the other hand, an adversary could exploit the meaningful input space for which the target model is sufficiently confident. For example, Choquette-Choo et al. [12] built an attack that works by querying the model

¹Overfitting is a common, but undesirable, phenomenon in ML in which a model fits the training data too closely, adapting to its random variation, to the point of not being able to generalize the learned relation (between input features and main task label) to unseen data.

²Error of model prediction on unseen data.

on augmentations of a data point ³ and it is based on the 0-1 loss ⁴ value of each perturbed sample. The intuition behind this approach is that a model trained with augmentation will have been trained to strongly recognize a member and its augmentations, particularly in vision domains. The success of this attack shows that data augmentation techniques can actually increase membership risk, an insight that was also highlighted by Carlini et al. [7].

Even though well-generalized ⁵ models may be exposed to MIAs, membership risk is still strongly related to *overfitting*. Yeom et al. [62] claims that although overfitting is a sufficient condition, it is not necessary. Attacks that prove that overfitting is a sufficient condition are based on the target model's error when predicting the target sample. However, assuming a colluding model trainer, a sufficiently complex model would be able to leak precise membership status by memorizing specific samples without hurting target model's performance. This certainly raises awareness about how excessively complex models can compromise data privacy - the weights that are supposed to contribute to the main-task are actually used for privacy violating purposes. The latter approach motivates the need to investigate other type of MIAs that aims to discover membership status that is somewhat encoded in model's parameters. For a model to be vulnerable to this type of MIAs, overfitting is not a request. Instead, these attacks increase model leakage. For example, Nguyen et al. [42] developed an attack that forced the model to leak precise membership status of a sample in a specific neuron's gradient. The impressive performance of this approach certainly raises awareness about how dangerous gradient sharing can be, as well as adversary's privileged access to the target model - conditions that are met in federating learning. More approaches are detailed in the following sections regarding the target and adversary's categorization.

To conclude, when looking for a defense strategy, other than obvious and, in some cases, useless defenses like increasing training dataset size (to improve model's generalization capability) or removing naturally high-risk samples (like outliers), one must look for strategies that reduce each particular point's impact on the target model. In other words, one must try to smooth the target model's approximation to the real function that takes an input and correctly returns its label. However, these defenses do not mitigate the latter, and strongest, type of MIAs, meaning active MIAs that increase model leakage (Nguyen et al. [42]). In fact, there is a noticeable lack of defense strategies for these.

2.2.2.2 Attribute Inference

In an Attribute Inference Attack (**AIA**), an attacker with access to the trained model attempts to infer a training sample's sensitive attribute value (Rigaki et al [46]). Note that it may be considered partial reconstruction, although the sensitive attribute does not need to be a predictive

³Augmentations of a data point are samples artificially created to improve model generalization and that are similar to that data point.

⁴Loss function that counts the number missclassifications.

⁵The opposite of overfitted models, meaning models that can generalized the learned relation between input features and main task label to unseen data.

feature. These usually *exploit the correlation between model performance, at the record-level, and sensitive attribute value*. Although the sensitivity of an attribute can vary depending on individual perspectives and on the context, AIA's level of threat ultimately depends on the destination of the inferred information, particularly of attributes that are often associated with discrimination, such as gender or race. For instance, in social media, a user's public data could be the list of pages that the user liked on Facebook. Given these page likes, an attacker can conduct an AIA to accurately infer user's private attributes including, gender, sexual orientation, and political view (Jia et al. [29]). Such inferred attributes can be, for instance, further leveraged to deliver personalized advertisements to users. It is important to note that, contrarily to MIAs, these typically require strong assumptions about adversarial knowledge of data distribution with respect to each attribute value, as well as non-sensitive information regarding each target data point. Therefore, one must improve on prior knowledge capability to succeed.

Attribute inference attacks can be further divided by the type of attribute being inferred. The target attribute may be a predictive feature, whose value can be somewhat encoded in model's output or parameters, or a non-predictive feature, whose information may be intrinsically learned by the target model. Although some attacks might work regardless of the type of attribute (Song et al. [50]), most require it to be a predictive feature [28, 36, 51, 62]. Note that AIAs are closely related to MIAs (Yeom et al. [62]), this connection is explored latter in this chapter.

Machine learning models implicitly learn to recognize sensitive attributes and properties that are not part of the learning task. Sometimes it may not be possible to prevent this leakage, since it comes from the simplest utility to be retrieved from a model - its predictions. Song et al. [51] shows that models unintentionally learn statistical properties of sensitive attributes, and even decreasing output's information about them, via censoring techniques to reduce the amount of mutual information that output vectors contain about the sensitive attribute value, can be defeated by a sufficiently powerful adversary. This is even more concerning for non-predictive features, as they cannot be censored using any known technique. The fact that overlearning may be unavoidable raises urgent need of a balanced trade-off between model utility and data privacy. Even when overlearning of intrinsic sensitive attributes is not enough for the desired attack performance, an active adversary with privileged control over the target model in a collaborative training system (e.g. central server) can encode their values in the output, namely by forcing the model to return an output vector with higher entropy for a sample with the positive sensitive attribute value (Jayaraman et al. [36]).

Both target models and publicly available knowledge can leak information regarding sensitive attributes. In fact, Jayaraman et al. [28] critical view of AIA and respective success enables the author to better judge additional privacy violation of releasing models. Is it reasonable to blame model's release anytime an adversary accurately infers a sensitive attribute value? Authors argue that previous AIA, specially black-box (query access), do not reveal more about training data than can be inferred by an adversary without access to the trained model, but with the same knowledge of the underlying distribution needed to conduct the attack. Black-box attacks developed to prove this insight holds are based both on model's confidence and sensitive

attribute’s distribution. Authors compare these attacks with data imputation⁶ and conclude that *model release does not create additional privacy risk*. Note that no additional privacy risk is different from saying that models do not leak information about training data. It only means that such leakage is a greater threat when adversaries have poor prior knowledge about training data. In such cases, access to a model can make a huge difference as the ability of the model to express statistical information (main task related or not) can be further exploited to infer sensitive information about training data. However, a critical reader must be wondering if sensitive attribute information leakage in a black-box setting is always negligible. The answer depends on the adversarial knowledge about data distribution and ultimately on the data imputation technique, which in this case is a neural network based one. Nevertheless, this critical view suggests the need to compare AIA performance to powerful data imputation techniques to better judge additional model leakage.

Building upon this critical view and considering a metric that captures additional privacy risk of model release, attack performance ultimately depends on the sensitive attribute, particularly its association with the main task label, as well as the attack design and data imputation technique. Intuitively, the co-occurrence relationship between the sensitive attribute and the main task label plays an important role. If the attribute is a predictive feature, then it probably has some type of association with the main task label, otherwise it would probably be removed from the set of predictive features. In the case of an extremely strong association, confidence scores will most likely contain considerable mutual information with the sensitive attribute value. But such a strong association is already probably known. Take for example the case of medical records and a model that predicts the probability of dying in the hospital, it could be considered sensitive information if the patient had cardio-respiratory arrest (condition most likely fatal) or if it was intubated (usually means critical clinical situation, most likely leading to death). In this situation, it is quite challenging to protect these samples from AIAs targeting these sensitive features because prior knowledge already leaks this information. Therefore, an adversary would not need model access to succeed.

The most interesting case is when the attribute is moderately related to the main task label. Joining prior knowledge about the sensitive attribute, non-sensitive attributes of records and confidence scores may improve on data imputation. Note that this insight does not contradict the empirically validated insight about black-box AIAs in Jayaraman et al. [28]. Neither does it devalue the need of comprehensively evaluate additional privacy risk caused by model release. It only questions if those conclusions are independent of the data imputation technique and its ability to capture mutual information about the sensitive attribute present in confidence scores. Possible defense strategies include adversarial training to reduce mutual information between confidence scores and sensitive attribute value, which is latter explained in Section 2.3.3, even if the attribute is not a predictive feature (Song et al. [51], Song et al. [50]).

A different type of AIAs that exploit the relationship between neuron’s activations and

⁶Process of replacing missing features on a sample by leveraging prior knowledge about the missing feature and its relation with the remaining known features.

attribute values are a major threat (Jayaraman et al. [28]), especially on FL settings having access to other users' gradient updates. These are in fact quite challenging to prevent, even if there is only a moderate relationship between the sensitive attribute and the main task label. As the whole point of neural networks is precisely to account for different contributions (activations) regarding distinct attribute values into the final output, at least for attributes with some correlation with the main task label. Therefore, investigating new defense strategies for this type of threat is an interesting future direction.

2.2.2.3 Property Inference

Closely related to AIA is Property Inference Attack (PIA), in which an attacker with access to the trained model attempts to infer a property of the whole training dataset (Rigaki et al [46]). Although both attacks aim to infer sensitive attribute information, one at the record-level (AIA) and the other at the populational-level (PIA), some papers use both terms interchangeably. For instance, an AIA may try to infer the gender of a person (based on a particular image) participating in a face recognition model, and a PIA may try to infer whether there was an imbalanced training dataset with respect to gender, meaning if the model was trained with more men than women. Here, there is a clear difference between AIA and PIA. Furthermore, PIAs vary on their specific goal, which can be to infer property existence or its frequency. PIAs usually *exploit the correlation between model performance and sensitive attribute value at the populational-level*. Aside from different goals, PIAs can be further divided according to how they exploit the relationship between property and main task. Some might try to capture the influence of property presence (or size) through model parameters since the goal is to infer global information about the training dataset, such as several reviewed papers [17, 38, 43, 53, 56]. Other types of PIAs may try to capture such influence through model outputs (Chase et al. [35], Chaudhari et al. [8], Suri et al. [53]). These aim to capture *different trends of model's behaviour* on a collection of data points when the target model is trained under a specific property proportion (or under the property's existence/absence). Although similar to sample-specific inference attacks, such as membership or attribute, to infer about training data as a whole, the attacker needs to decide based on a set of sample-specific inferences (via hypothesis testing for example).

For a model to be more vulnerable to PIAs, other than overlearning training data distribution, one could think that high model complexity may be a risk factor. For that reason, Parisot et al. [43] focuses on the behaviour of PIAs performed on models with different complexities. The distinction factor of this study is that prior work usually focuses on developing better attacks, not on their success related to the type of target model. For this purpose, a classifier is trained on auxiliary, also called shadow, models, particularly model parameters, which were trained on datasets with and without the property. This classifier is then used to predict whether the target model was trained on a dataset with and without the property. However, by varying model complexity measured by the number of model parameters, *results do not reveal any relationship between model complexity and vulnerability to PIAs* (measured by attack accuracy).

This conclusion is not expected having in mind that more complex models are usually able to better learn statistical properties of training data.

Naturally, a property that is highly correlated with the main task, would be easily inferred from analysing the target model’s behaviour. However, less correlated properties may also be easily inferred, particularly if an active attacker artificially increases that correlation. To increase model leakage, Chase et al. [35] and Chaudhari et al. [8] developed PIAs based on data poisoning attacks, in which the adversary intentionally sends corrupted data with some malicious purpose. Federated learning is not exposed to these threats as it does not require data exchanging (Yin et al. [63]), but other collaborative learning techniques might. Nevertheless, an adversary sends poisoning data aiming to introduce correlation between the target property and the main-task label in the training dataset, which in its turn will change the target model’s decision boundary⁷. To maximize the difference between poisoned and clean data, hopefully decreasing required poisoning rate without hurting attack performance, authors point to the least frequent type of property and main-task label co-occurrence as the best way to artificially increase correlation. To infer whether the target model was trained on a dataset with the property, an attack model must be trained to distinguish a classifier’s behaviour on both situations. Results show that as poisoning rate increases so does attack accuracy, showcasing that properties that are correlated with the main-task, even if artificially, are easier to infer than uncorrelated properties.

Sensitive properties in the context can enhance the threat of PIAs on a subjective level, such as the presence of a person in another ones’ image folder. However objectively, property inference is successful, comparing to what an adversary could retrieve from prior knowledge about the training distribution, only if *property presence has a significant impact on the final decision boundary or model parameters*. Since the target property is a characteristic of the training data distribution, different proportions of its presence yield different distributions. Evaluating whether its presence (or size) makes a difference in model’s prediction is *equivalent to test if a model is remarkably biased towards its training data distribution*. This intuitively presents similarities with membership inference, although with a slightly different formulation, as the goal is not to infer membership status of a particular sample but of a particular set of samples, describing a distribution, for which the property is present in a fraction of the set. In a face recognition system, for example, when trying to infer if a person (identity) is on the training set an attacker may not hold the same image that was used for training but be able to generate a set of images from the same identity, therefore determining if the identity (property) is present. In the latter case, one would classify this attack as a PIA, yet it is closely related to a MIA, particularly a relaxed version of it (Parisot et al. [43]).

Defense strategies against PIAs are poorly explored in the literature. However, user-level Differential Privacy (detailed in Section 2.3.6), in a FL setting, can be an interesting direction if the property being inferred is user-specific. Although, it is usually associated with high model utility loss. Moreover, the intimate connection between property and alternative membership inference indicates that perhaps traditional defenses against MIAs can also reduce PIAs success,

⁷Hyperplane, on the feature space, that separates different classes.

mainly defenses that aim to improve the target model’s generalization capability. However, they do not seem promising as the goal of a PIA is to learn about statistical properties of the underlying training distribution which may still be present even if individual training records’ contribution is obfuscated. In fact, depending on the target property, these attacks may be unavoidable (Song et al. [51]).

2.2.2.4 Reconstruction Attacks

In a Reconstruction Attack (RA) an attacker with access to the trained model attempts to reconstruct training samples, either the whole data record or just part of it, given output labels and partial knowledge of some features (Rigaki et al [46]). Also known as Model Inversions (MIs), these attacks generically take advantage of *data records’ correlation with their respective model output or gradients*, as RAs often require access to gradient updates. With a powerful RA it is possible to recover exact training samples from gradient updates (Zhao et al. [67]), which makes them specially concerning in a FL scenario. Take for example the case of a collaborative trained speech recognition system, an adversary able to reconstruct user’s data will be able to listen to private conversations.

Most common model inversion attacks aim to reconstruct training data from target model’s behaviour (in the form of confidence scores vectors or gradient updates). These are called optimization-based, since an optimization algorithm is developed to start from a particular response and a dummy, or a blurred/corrupted, sample and get the original training sample that when passed through the model gets the target response, such as several reviewed papers [15, 18, 21, 41, 50, 54, 65–68]. The intuition behind this approach is that similar gradients/model responses come from similar samples (Zhu et al. [68]), whereas analytic MI attacks customize model parameters/architecture to directly retrieve training data from gradients of a Fully Connected (FC) layer (Gupta et al. [21], Zhao et al. [67]). The latter type of attacks seem naturally stronger, as they require an active adversary, meaning one that interferes in the training process. RAs are remarkably distinct from inference attacks (previously explained), in terms of requirements, as they usually require unrealistic adversarial setups, such as access to individual gradient updates in a FL system and access to gradients of a single sample (meaning batch size equal to 1). However, they require fewer prior knowledge about training samples, as the whole point of these attacks is to precisely reconstruct training samples. As for risk factors, *the target model’s predictive power helps model inversion attackers achieving better results*, since those models are able to explain a sample’s response with its features successfully and MI attacks are known to exploit that relationship, as argued by Zhang et al. [65].

A broader concept of reconstruction attacks includes reconstruction of the training distribution, not exactly training inputs, but rather synthetic samples. These are called Class Representative Inference Attacks (REPR). Even though their associated definition of privacy violation may not be considered actual violation by some, as recovered images are not real training samples. Despite incurring in a poorly defined privacy leakage, these still pose a threat to ML models,

as a user should not trust a system that lets casual users recover images that are effectively indistinguishable from the ones in his picture folder (Hitaj et al. [24]). Take for example the case of a face recognition system in which Alice's photos were used during training, such adversary will be able to generate faces that resemble Alice. Even though not being able to actually reconstruct real photos, the adversary may give a malicious destination for these real looking Alice's images, for example marketing campaigns or adult content creation. Certainly frightening when attacking sensitive information, like human faces, its effectiveness is directly related to model's utility. In fact, poorly accurate models might mitigate this attack but to retrieve some utility from the learning task, the model must be considerably accurate which will lead to effective class representative inference attacks.

As a privacy-preserving framework, federated learning avoids data centralization, thus each participant has its own private data which may be uniquely biased towards the global distribution. Dishonest participants may be tempted to infer sensitive information about other participants data distribution, which is precisely the goal of REPRs. Few papers are published about these attacks, however Hitaj et al. [24] and Wang et al. [57] developed REPRs on FL that worked by training a Generative Adversarial Network (GAN)⁸ to generate samples from the target distribution. As in other type of attacks, the discriminator is the global model which must be able to discriminate real from fake samples of a particular class and, in some cases, of a particular client. However, these attacks usually require a privileged access to client's information, for example individual gradient updates, which is unrealistic for a privacy-preserving FL system.

Reviewed papers express concerns about the fact that well-known defense mechanisms, like differential privacy, do not mitigate this type of attacks by construction. By definition, Differential Privacy (DP) protects training samples from being recovered, but does not stop an attacker from generating synthetic samples.

2.2.2.5 Connection between different goals

At this stage, the reader must be wondering if there is a connection between reconstruction, membership and attribute inference attacks. Can a model be significantly more exposed to a specific type of adversary, and can the same defense strategy be applied against different attacks? Concerned about this connection, Song et al. [50] aims to show that embedding models leak sensitive information and are vulnerable to model inversion, attribute and membership inference attacks. This study shows that models can have *different levels of vulnerability regarding distinct adversarial goals* and their respective mitigation strategies can also affect model's utility differently. In this case, removing input word sensitive information from embeddings (which determine model inversion success) is a harder task than removing authorship (sensitive attribute) and thus will have a larger impact on model's utility. Since authorship is not a predictive feature,

⁸Popular class of models that can generate realistic but fake examples, these are trained orchestrating 2 networks, a generator and a discriminator. The generator is trained to fool the discriminator, while this one is trained to distinguish real from fake samples.

information regarding authorship is intrinsically learned by the model but can be easily removed through adversarial training. This may not hold for different couples of target models and sensitive attributes (Song et al. [51]), which poses a problem to defense strategies for AIAs research - it is highly dependent on the practical setting.

Even though the same target model (and its training data) can have different levels of vulnerability regarding distinct privacy attacks, there is a relationship between attribute, and membership vulnerability, which is theoretically supported by Salem et al. [47]. Particularly, *MIA and AIA are mutually reducible*, meaning one can find an AIA that is as good as any MIA and vice-versa. However, this does not seem to hold for the relation between MIA and RA. Indeed, their relation seems to be dependent on the type of MIA - average case MIAs that exploit the difference between the target model behaviour on members and non-members are not reducible to a RA. Salem et al. [47] suggests that, by considering a *membership adversary matching the capabilities of a reconstruction adversary*, one can reduce a MIA to a RA. Meaning *resilience against MIA would imply, in that case, resilience against RA*. Analogously, resilience against MIA implies resilience against AIA.

In fact, a *high-risk sample for membership inference is also a high-risk sample for attribute inference* even for the weakest type of MIA. Accurate membership inference of a sample requires the model to behave sufficiently different from other non-training samples. Therefore, one could perform MIA on every possible attribute value, given the rest of the non-sensitive attributes, and accurately predict the sensitive attribute value based on how differently the model behaved for every case (or how confident was the MIA). This conclusion still holds when a stronger type of MIA is applied, for example when model parameters encode membership status. From these insights, one must see that a model's vulnerability to membership inference seems to be a boundary for the same model's vulnerability to other privacy attacks, such as attribute inference. Therefore, investigating and mitigating this type of threat seems to be a promising future direction, particularly if such research can validate the connection between MIA and other privacy attacks.

2.2.3 Target of Attacks on ML

One of the main factors that determine privacy leakage of a ML model is the influence that a specific training data point has on its weights. Thus, a natural defense mechanism would be to decrease any individual point's influence by increasing training dataset size. However, joining and processing large volume databases has its disadvantages, such as the associated computational overhead, not to mention the challenge to find labeled data. Other than computational overhead, private data sharing may be a concern, which is one of the motivations for Federated Learning. As a distributed learning mechanism, FL works by aggregating client's local trained models into a final global model. The aggregating process aims to weight each client's contribution to the final model. FL is a *privacy-preserving framework* since it avoids data centralization. However, in a realistic scenario, the type of target model may limit the adversary's knowledge, as an

adversary that participates in a FL system and targets the model being trained has full access to it, whereas a privacy-preserving predictive system based on centralized learning only needs to share the model through an API. Thus, an adversary can target centralized models, such as several reviewed papers [7, 12, 26, 28, 44, 45, 48, 49, 51, 52, 61, 62], or federated learning models, such as a few reviewed papers [10, 20, 40, 42, 44, 58, 60, 64]. Despite its privacy-preserving concern, FL models may not always provide sufficient *privacy guarantees*, as they are also exposed to several privacy attacks. FL models are particularly exposed to attacks that affect the learning process (active) which aim to increase private information leakage. Plus, as both the central server and several clients participate in the learning process, any of them could attempt to violate other users' privacy. Moreover, regarding which phase to attack, adversaries targeting FL models may also choose, depending on their identity, when to attack, meaning if the attack occurs while the model is training or after the training is completed, as it would do on any privacy-preserving predictive system based on centralized learning.

Furthermore, FL models are still vulnerable to attacks conducted by outsiders, meaning adversaries that do not participate in the learning scheme, as any other centralized learning model. However, Pustozero et al. [44] showed that MIAs on centralized models reach greater performance than outsider attacks on FL models, due to the reduced influence of each training batch into the final model in the latter case. Insider attackers, however, apparently break this advantage by performing attacks at each step, since the recently updated global model has weights determined by just a few clients, frequently with less local training data when compared to a centralized dataset. Therefore, FL models can be more robust or more vulnerable, depending on who might be the adversary. This study contributes for an understanding of the effect of the learning protocol in MIAs performance. Thus, the key to effectively attack a federated model lies on *when* and *how* to attack, as well as *who* is attacking. However, regardless of the adversary's identity and knowledge about the target model, a monitorization of the behaviour of a federated model with respect to a set of target samples may increase an attacker's chance to correctly predict membership status, as it is expected to better capture the target model's dependency on the training set. This means that there is continuous leakage of membership information, which can be leveraged to increase attack performance (Shokri et al. [40], Gu et al. [20]).

2.2.4 Adversary of ML models

As previously discussed in Section 2.2.3, one of the keys to effectively attack a federated model lies on *how* to attack (approach), as well as *who* is attacking (adversary's identity). Ultimately, the approach will be determined by the adversary's access and desired effect on the target model. The adversary identity denotes whether it is an outsider or an insider, meaning its role on the target model's training process before the attack occurs. In its turn, the effect relates to the adversary's interference on the training process by performing the attack, meaning if, by conducting an attack, the adversary impacts the target model's training process differently than it used to, in case the adversary already participated in the training process. These adversarial components will be explored throughout the rest of this section.

2.2.4.1 Identity

Regarding the identity of an adversary, a model can be attacked by an insider [10, 20, 20, 40, 40, 42, 44, 58, 64] or an outsider [20, 30, 40, 42], meaning an attacker that is part of the learning process and one that simply uses the final model, often published as an API, respectively. In a realistic scenario, a privacy-preserving predictive system will only be attacked by outsiders, since the model is only accessible to users. However, a privacy-preserving FL system can be attacked by both, even though outsiders are expected to have a restricted access. Thus, each type of adversary will then have its possible type of access, further explored in 2.2.4.2, and its possible effect, further explored in 2.2.4.3. Note that the term *possible* accounts for the most probable combination of adversary identity, access, and effect, that is outsiders have restricted access and no impact on the training process, whereas insiders may have unrestricted access and may have an impact on the training process. The identity of an adversary, as well as access and effect, often determine the type of attack design (approach) and, consequently, attack performance, which is explained throughout the rest of this section.

Further detailing different types of insider adversaries in a FL system, an insider adversary can be a participant [10, 20, 40, 44, 58, 64], meaning a data owner that is part of the learning process, or the central server [20, 40, 42], that does not own data to train the model but orchestrates several participants, therefore having a different role in the learning process. A different role in the learning process most likely leads to a different type of attack. An attack conducted by the central server (also called global attack) is often viewed as a potentially stronger attack than one performed by a participant (also called local attack), since they can have different types of access and their effect impacts the target model differently. Shokri et al. [40] and Gu et al. [20] developed global and local attacks (conducted by the central server or by the participant, respectively), using the same strategy but assuming a different type of access and effect. Both of them concluded that global attacks outperformed local ones. However, that is due to the assumption that individual gradient updates are accessible to the central server, which is an unrealistic assumption for a privacy-preserving FL system. Therefore, under a realistic assumption, global and local attacks that do not have an impact on the target model have the same performance. However, if the attack depends on the adversary's ability to leverage its effect for a malicious purpose, global and local attacks may perform differently, since a participant's effect can fade away in the aggregation process. Different effects on the target model are further detailed in 2.2.4.3.

2.2.4.2 Access

Access to the target model is one of the most important adversarial characteristics that ultimately determines attack success. It refers to the adversary's knowledge about the target model's architecture and learned parameters. One can have white and black-box (Hu et al. [25]). Other definitions may include grey-box, which is intended to describe attacks that are neither black nor white, naturally. In a white-box setting, the adversary has access to the target

model’s architecture and its learned parameters, as it is the case of several reviewed papers [10, 20, 40, 42, 44, 58, 60, 64]. In a black-box setting, the attacker has access to queries (model’s output/response) returned by the target model, also known as query access, as it is the case of several reviewed papers [12, 26, 30, 45, 52, 61]. In a grey-box setting, it is assumed that the attacker has query access and knows the target model’s architecture but does not have access to model parameters, as it is the case of several reviewed papers [12, 26, 30, 45, 52, 61]. The type of accessible target model’s response plays an important role in attack performance, which can be the full confidence scores vector, only the top-k confidence scores or the prediction label. The latter case is the most challenging, yet the most dangerous because when a model is moderately vulnerable to this type of attack, then it must be considerably more vulnerable to stronger attacks.

However, privileged access is not always a guarantee of greater attack performance. Shokri et al. [40] aims to show that a straightforward extension from black to white-box is ineffective, especially for well-generalized models and that even *well-generalized models are significantly susceptible to MIAs* when these are carefully designed. The proposed attack can be described by being a white-box attack that exploits the privacy vulnerability of Stochastic Gradient Descent (SGD). But why should an adversary explore SGD? Each training point will leave a distinguishable footprint on gradients, which can be detected by a powerful attack model. Adding activation functions’ outputs from all hidden layers to the output of the last layer, typical of a black-box scenario, composes the straightforward extension. This is inefficient because if the model generalizes well, so do the activation functions. But having access to gradients and since its dimension is often an order of magnitude greater than the training size, the whole gradient may not generalize well over training data. So, for Shokri et al. [40], the key to effectively exploit white-box access relies on using both gradients and activation function’s output to leverage a MIA. Experiments with features from different layers, either the activation function’s output or the gradient, reveal that *last layers leak more membership information*. No significant gain in attack accuracy comes with the addition of features from the first layer. Between gradients and activation function’s output, the former leaks significantly more information about the training set. Moreover, results reveal that membership leakage depends on how distinct the last-layer gradient norms’ distribution from members and non-members are, which might be different between classes. This dependency seems to be even stronger than generalization error.

In its turn, grey-box access enables adversaries to train shadow models, meaning auxiliary models to mimic the target model’s behaviour, which behave even closely to the target model if they share the same architecture. This approach, further detailed on 2.2.4.4, aims to build a labelled dataset with respect to membership status so that a classifier is able to determine whether the target model’s response to a given sample is similar to the shadow model’s response when that sample is used during training or not. Note that this scenario is often labelled as black-box in the literature, since the attack will only use target model’s response on the inference phase, as black-box attacks do.

In some settings, namely when the target model is released as an API, adversarial access

may be restricted to the output layer, either the full vector, top-k confidence scores or only the predicted label (called label only attacks). Any attack conducted on a black-box setting is not supposed to improve upon white-box access, and label only attacks can only hopefully be as successful as unrestricted black-box access attacks are. Choquette-Choo et al. [12] proposed a label-only attack that estimates the target sample’s distance to the decision boundary based on the predicted labels of neighbouring points (equivalent to finding the smallest adversarial perturbation). The intuition behind this approach is that points far away from the decision boundary are more likely to be a member because to misclassify them one would need to apply strong perturbations, as the model is supposedly more confident with these points. Particularly on vision domains, if neighbouring points are obtained via data augmentation, the adversary can exploit the fact that a model trained with augmentation, common on this domain, will have been trained to strongly recognize a member and its augmentations. Furthermore, Rahimian et al. [45] proposed an approach, based on label-only access, to exploit the target model’s *sensibility to a small perturbation* of a sample to approximate the vector of confidence-scores. Similarly based on perturbed samples, an histogram over the predicted labels of these points is considered an approximation to the output vector. This approach can be used as an extra step to any kind of MIA.

Also to address the possibility of a limited target model access, specifically when only the maximum confidence score is accessible, Salem et al. [48] shows that an adversary can leverage the fact that a model tends to be more confident when predicting a training sample to infer target samples’ membership status. Intuitively, the most dangerous type of model leakage occurs when label only, or more generally black-box, attacks are successful.

2.2.4.3 Effect

Regarding adversarial approach, one can also distinguish attackers based on their interference with the target model’s training process. An adversary that simply observes the training process or the final model to infer membership, for example, is said to be honest-but-curious (passive attacks, as in several reviewed papers [7, 12, 20, 26, 30, 40, 44, 45, 48, 49, 52, 58, 61, 62, 64]). On the contrary, an adversary that intentionally modifies the learning process, namely by injecting poisoned data (Chase et al. [35]) or model parameters (Nguyen et al. [42]), is said to be malicious, as it participates in the learning process with the goal of increasing information leakage (active attacks, as in several reviewed papers [10, 20, 40, 42]). Note that an outsider adversary cannot interfere in the learning process, therefore this categorization is only meaningful for insiders.

Active and passive attacks are frequently compared in the literature (Shokri et al. [40], Gu et al. [20]) and almost always active attacks outperform passive ones. For example, Shokri et al. [40] proposed an active MIA that aims to increase membership information leakage by updating the target model with the gradient ascent of target samples, which goes exactly in the opposite direction, therefore increasing target sample’s loss. The intuition behind this attack is that if the target sample is used for training then its loss will be significantly smaller in the following

round. The equivalent passive version simply analyses target samples' loss evolution along the training process, which is expected to lead to a less powerful attack, as some training samples' loss may have a stable evolution. Isolating attack, another approach proposed by Shokri et al. [40], consists of skipping the gradient updates' aggregation process, meaning each client trains its own model while thinking it is training collaboratively with other participants as it is part of a FL system.

Among active adversaries, one can have malicious participants manipulating the global model in order to increase information leakage [20, 40] or a malicious server [20, 36, 40, 42]. Differences between both cases rely on the fact that a participant's malicious impact will be averaged out in the aggregation process, most likely leading to a decrease of attack performance, comparing to active attacks conducted by the central server, because its impact will not be averaged out. Moreover, different levels of access and control also increase the gap between global and local active attacks, as, in some papers, it is assumed that the central server has access to individual gradients. For example, some approaches would not be possible without the privileged access and control that only a central server can have (Zhao et al. [67]).

2.2.4.4 Approach

At the approach level, there are two main categories: model-based and metric-based attacks (Hu et al. [25]). New approaches may emerge as research evolves. Focusing on MIAs to better understand different approaches, in a model-based MIA, the attacker trains a binary classifier to distinguish target models' behaviour on members from its behaviour on non-members, meaning this classifier predicts the probability of a sample being a member. The way this classifier, known as attack model, is trained depends on the adversary's knowledge about the target model. Several papers proposing MIAs use this approach along with shadow training [10, 12, 20, 40, 44, 48, 49, 58, 64]. Shadow training consists of training several shadow models to mimic the behaviour of the target model and about which the adversary knows the ground-truth label (real membership status of shadow samples). This approach may be coupled with an adversarial network that takes as input a dataset with tuples labelled as members and non-members (of the shadow models) to predict membership status, as in model based attacks. Note that the shadow training technique transforms an unsupervised learning task into a supervised one (Shokri et al. [40]), which is the primary reason for one to develop a model-based using this technique. Note that training shadow models may increase the adversary's computational resources needed to attack. Model-based MIAs can find a non-linear decision boundary taking as input several types of features from the target model, such as true label and confidence score vectors [12, 20, 44, 48, 49, 58, 64]. The intuition behind this type of MIA is that a model is usually more confident when predicting a sample with which it was trained. For these examples, the adversary only needs to have black-box access, whereas to fit as input target sample's gradient update, for instance, white-box access is needed (Shokri et al. [40]). Note that most black-box model-based attacks are actually grey-box as they train shadow models with the same architecture as the target model.

In its turn, metric-based MIAs are more simple and less computationally expensive, as they do not involve training an attack model. These attacks are based on pre-calculated metrics, or metrics obtained from shadow training, that are compared with a preset threshold to decide the membership status (Hu et al. [25]). Among several types of metrics, loss [49, 61], confidence [12, 20, 26, 45], and entropy [20, 52] are the most used (Hu et al. [25]). The intuition behind these types of attacks is the same, meaning a model is usually more confident when predicting a sample with which it was trained. Thus, an attacker can infer that a record is a member if it is correctly predicted, if its loss is sufficiently smaller, if its maximum confidence is sufficiently large, or if its entropy is sufficiently small. Note that although the intuition is the same, model-based attacks are expected to outperform metric-based ones, as they are able to find non-linear decision boundaries and take more complex inputs. Furthermore, metric-based MIAs may also leverage white-box access (Nguyen et al. [42]) or use shadow training technique to compute the threshold (Shokri et al. [49] and Carlini et al. [7]).

Many powerful MIAs require enough labelled (with respect to membership) data to learn the decision boundary, e.g., the model-based attack proposed by Shokri et al. [49]. Non-members datasets are usually easier to obtain but that is still not enough to train a binary attack model. One possible solution is to perform metric-based attacks, yet they usually achieve slightly worst performance (Rahimian et al. [45]). Moreover, when the target model is released as an API, it may come with additional security triggers to stop the attacker from querying the model on a huge/specific dataset. So, even the non-members set generation can be a problem. Unavailability of ground-truth is also an issue often disregarded when designing a MIA. To address all these issues, Hui et al. [26] proposed a new approach - *differential comparison*. Its emergence was motivated by the need to tackle the lack of assumptions about adversarial knowledge about data distribution and unrestrained access to the target model. As the name suggests, it uses the Differential Comparison technique. The intuition behind this approach is that mixing two sets of different items gradually approximates both sets, as they tend to be a mixture of both items. Starting with two sets, a non-members and a mixture set, the attacker iteratively moves samples from the latter set to the former. If the distance between these sets increases then the sample is a non-member, otherwise it is a member, where the maximum mean discrepancy between centroids (of confidence scores vectors mapped to the Reproducing Kernel Hilbert Space) is taken as the distance metric.

Novel approaches to overcome the unsupervised nature of a membership inference task have emerged, such as a clustering-based approach proposed by Shokri et al. [40]. Taking advantage of white-box access, the goal is to obtain a representation (embedding) of the data point's gradient update such that a spectral clustering method can separate members from non-members. One must define as the set of non-members the set of data points for which the gradient norm is larger.

2.2.5 Taxonomy of Privacy Attacks on ML

To summarize the review, a taxonomy of privacy attacks on ML models is created and illustrated in Figure 2.3. Inspired by Yin et al. [63] and Hu et al. [25] respective taxonomies, this taxonomy classifies privacy attacks based on goal, target and adversary specific characteristics, in accordance with the discussion from previous sections. Linkage between reviewed papers and their classification based on this taxonomy is presented in the same figure. Note that approaches of RAs are not detailed as they would require a more extensive study, besides being remarkably different from inference attacks, which are the focus of this work. Nevertheless, recall that RAs are mainly divided into optimization based and analytic approaches.

2.3 Defense Mechanisms

Vulnerabilities of machine learning models should already be clear to the reader, but privacy defenses are not. Throughout the literature review of privacy attacks, some defense mechanisms have been tested against those attacks. Under-explored, defense strategies are frequently not effective or incur in high model utility loss. Naive approaches aim to *improve target model's generalization*, such as increasing dataset size, yet that is often not possible or can be defeated by an adversary with extra capabilities. Moreover, even *well-generalized models leak information* of hard to fit examples (e.g. outliers), or of under-represented input regions. In these cases, one may consider that removing these samples/regions from the training dataset is a sufficiently good strategy. In fact, it would protect these samples' privacy, but not all data mining problems aim to model the expected, some aim to model unexpected samples (e.g. fraud detection, and automobile risk estimation). Therefore, this strategy must not be intended as a satisfactory defense mechanism that is main task agnostic, which motivates the search of other solutions. Most common defense mechanisms in the literature can be divided into 9 categories: confidence score masking, regularization, adversarial training, perturbation techniques, gradient pruning, differential privacy, knowledge distillation, secure multi-party computation, and attack detection.

In this section, the reader can find an overview of defense mechanisms for several privacy attacks, in some cases there is a brief introduction as it is not assumed that the reader has prior knowledge on those defenses. Confidence score masking strategies are reviewed in 2.3.1, regularization in 2.3.2, adversarial training in 2.3.3, perturbation techniques in 2.3.4, gradient pruning in 2.3.5, differential privacy in 2.3.6, knowledge distillation in 2.3.7, secure multi-party computation in 2.3.8, and attack detection in 2.3.9. Additionally, customizable local defense mechanisms are explored in 2.3.10 which should motivate local defense strategies to be developed accounting for users preferences.

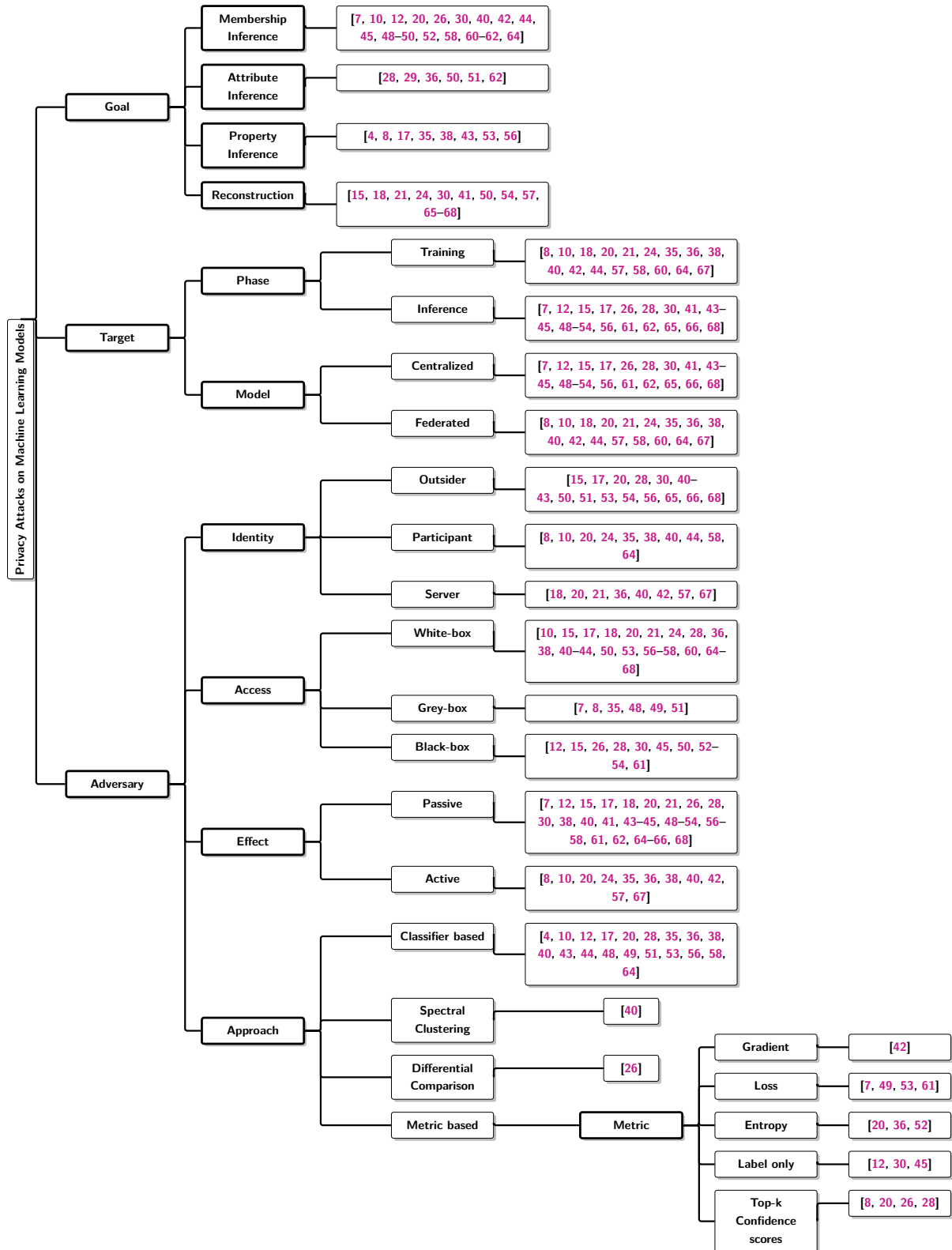


Figure 2.3: Taxonomy of privacy attacks - diagram.

2.3.1 Confidence Score Masking

Confidence score masking aims to *hide the true confidence scores returned by the target classifier* (Hu et al. [25]). By definition, it does not mitigate white-box attacks, only black and grey-box eventually. Confidence score masking techniques include restrain the output to the top-k confidence scores or to the predicted (hard) label. It may also denote output perturbation techniques, such as adding noise to the output, and outputs with reduced precision. The intuition behind applying confidence score masking techniques to prevent privacy attacks is to hopefully prevent the adversary from exploiting model's behaviour on its behalf. For example, restrain the output to the predicted label may prevent a MIA by hiding the potential memorization of a training sample, as the target model would have a greater confidence score for memorized samples, yet only the correctly predicted label would be visible for the attacker. Song et al. [52] showed that confidence score masking technique reduced entropy-based MIA accuracy, yet the attack was still effective. Zhu et al. [68] showed that half-precision fails to mitigate DLG (deep leakage from gradients, a reconstruction attack) or when it does it decreases model accuracy significantly.

2.3.2 Regularization

Regularization techniques, frequently applied outside the scope of privacy protection, aim to *decrease overfitting by improving a model's generalization ability*. Most common approaches include dropout, L1 and L2-regularization, early stopping, and data augmentation. Dropout is a technique where random neurons are temporarily removed during training to prevent the model from becoming overly reliant on specific pathways, thereby reducing overfitting. In its turn, L1 and L2-regularization are techniques that add a penalty equivalent, or proportional, to the model coefficients, promoting sparsity by driving some coefficients to zero, or smaller and more evenly distributed weights, to improve generalization [3]. Additionally, a model trainer applying early stopping must monitor model's performance on a validation set and stop training once performance ceases to improve, preventing overfitting by avoiding excessive training. While these techniques are applied at the training phase, the quality of training data, particularly fair representativeness of the feature space, may also improve the model's generalization ability. For that purpose, data augmentation artificially expands the training dataset by applying transformations, such as rotations or flips (common on data visualization problems), to increase model robustness and reduce overfitting. These techniques are essentially applied against MIAs, since most of these exploit memorization of training samples. Song et al. [52] showed that early stopping reduced entropy-based MIA accuracy, yet the attack was still effective, similarly to other attempted defenses, such as confidence score masking and adversarial training (explained in Section 2.3.3). Gu et al. [20] showed that dropout decreased active and passive white-box attacks and main task accuracy, therefore not effectively mitigating attacks. Melis et al. [38] showed that dropout is also not effective at mitigating white-box PIAs on FL.

2.3.3 Adversarial Training

Adversarial training methods use adversarial techniques to train machine learning models to be *more robust against adversarial attacks*. As a defense mechanism, the training process incorporates an *adversarial loss function that penalizes the model for making predictions that inadvertently leak information*, which combined with the main task loss function aims to *balance main task accuracy and privacy*. The intuition behind this defense strategy is to penalize main task loss with attack performance. One must assume that the adversary has full knowledge of the classifier parameters, and the goal is to ensure that the model cannot be attacked even if the adversary has full knowledge (white-box access) [31]. A generic adversarial loss function, where $loss$ stands for the standard main task loss and $objective_{adv}$ stands for the adversarial objective which the attacker aims to maximize, can be described as follows:

$$\min loss (\max objective_{adv})$$

Song et al. [52] showed that adversarial training reduced entropy-based MIA accuracy, yet the attack was still effective. Song et al. [51] proposed adversarial training, particularly a censoring technique to encode the input into a representation that does not reveal unwanted properties about the input, yet is expressive enough to predict the main task label. However, the same authors also designed an adversary capable of overcoming that defense. Furthermore, stronger censoring does not reduce attack accuracy but reduces main task accuracy significantly for a slight reduction of adversarial accuracy, thus being a non effective defense strategy.

Adversarial training can also be applied against AIAs that aim to infer information about non-predictive sensitive attributes. Song et al. [50] showed that adversary’s performance on inferring non-predictive sensitive attribute (authorship) drops significantly (with an adequately tuned hyperparameter) while model utility remains stable, therefore being a *successful defense strategy*. However, adversarial training might be successful at protecting against a specific objective, i.e. a particular sensitive attribute, but the adversary might have multiple objectives. Using adversarial training to deceive the latter type of adversary most likely leads to a significant main task performance drop.

2.3.4 Perturbation Techniques

Perturbation techniques aim to add noise, either to the output or input of the model, or the gradient, to fool the attacker into inferring incorrect information. These can be applied against any type of attack. For example, Zhu et al. [68] added Gaussian and Laplacian noise on gradients as an attempt to mitigate deep leakage from gradients (DLG, a model inversion attack). The greater the variance the stronger is the defense, but it also degrades model accuracy significantly. Furthermore, Zhao et al. [67] also studied how could a local gradient noise addition mechanism mitigate LOKI (active reconstruction attack on FL systems) without hurting model utility. Assuming secure aggregation is applied to hide individual gradient updates from the central

server, if the noise is divided by all participants such that each one adds a small amount (not affecting local model utility), then one would need 10000 clients to mitigate LOKI.

2.3.5 Gradient Pruning

Gradient pruning aims to *balance the trade-off between model capacity and privacy*, pruning to zero gradients of small magnitudes. This defense strategy is most suitable for white-box attacks that exploit model complexity, as zeroed out gradients may hide a part of the model's behaviour with respect to the target sample. Zhu et al. [68] tested gradient compression and sparsification against DLG (reconstruction attack) and showed that pruning more than 20% effectively mitigates the attack. Malekzadeh et al. [36] hypothesized that a pruning technique could decrease mitigate HBC (an active AIA) since these classifiers could be exploiting overparameterization of models. Moreover, Gupta et al. [21] showed that gradient pruning is not effective at mitigating private text recovery in FL because attacks recover tokens with non-zero gradients. However, attack recall decreases as gradient pruning ratio increases since word embeddings get completely zeroed out.

2.3.6 Differential Privacy

Differential Privacy is a privacy-preserving technique that intuitively aims to reduce a sample's contribution to the trained model. Formally, let $K(\cdot)$ be a randomized function, ϵ and δ thresholds defined by the model trainer, D and D' be two datasets differing on at most one sample, then for all possible outputs $S \in \text{Range}(K)$:

$$P(K(D) \in S) \leq \exp(\epsilon)P(K(D') \in S) + \delta$$

In other words, and letting K be a randomized/stochastic machine learning model, the *presence or absence of any single data record should not significantly change model's behaviour*. Naturally, smaller ϵ yields stronger privacy guarantees, δ is typically very small and its purpose is to relax a strongest definition (with $\delta = 0$). Therefore, an algorithm trained under (ϵ, δ) -DP is such that each sample's contribution to its behaviour is as small as desired (by the choice of ϵ and δ).

Differentially private algorithms *carefully add noise while preserving statistical properties*. Some principal methods are the addition of noise, for continuous data, and choosing according to an exponential distribution, for discrete data [14]. The amount of noise added is carefully calibrated based on the sensitivity of the algorithm and the chosen ϵ . Despite powerful theoretical guarantees, DP is frequently associated with *high model utility loss*, as empirically proved by Gu et al. [20], Melis et al. [38], and Gupta et al. [21].

In federated learning, where the central server is potentially curious, private data owners must ensure that the central server cannot gain knowledge of individual data records. In this

setting, instead of a centralized approach of adding noise to ensure standard differential privacy guarantees, a decentralized approach is designed - Local Differential Privacy (**LDP**). LDP requires each participant to add noise to its local private data before any centralized aggregation takes place, thus not depending on a trusted third-party [59].

DP can be applied against several types of attacks. As an example of DP to mitigate MIAs, Gu et al. [20] showed that differential privacy with smaller budgets effectively mitigates the attack but at a high cost in terms of model performance. To mitigate reconstruction attacks, particularly class representative inference attacks, Hitaj et al. [24] and Wang et al. [57] stated that DP does not mitigate these attacks by construction – it protects training samples from being recovered, but does not stop an attacker from generating synthetic samples. To mitigate model inversion attacks, Gupta et al. [21] showed that Differentially Private Stochastic Gradient Descent (**DPSGD**) effectively mitigates the proposed attack but at a high cost in terms of model performance. To mitigate property inference attacks, Melis et al. [38] showed that participant-level DP could not be applied in that setting, as the target model did not converge, arguing that this technique is only feasible for a very large number of participants.

2.3.7 Knowledge Distillation

Knowledge Distillation (**KD**) is an effective technique for compressing and accelerating deep neural networks ([19]). It involves training a smaller *student model to mimic the behaviour of a larger and more complex teacher model*. This process helps the student model to learn more efficiently and effectively, reducing the number of parameters and computational complexity required to achieve similar performance as the teacher model.

Knowledge distillation can be described by the training scheme and type of knowledge being transferred. Regarding training scheme, one can have online, offline or self-distillation ([19]). Offline distillation requires a pre-trained teacher model to then transfer knowledge to students, whereas online distillation consists of training a teacher model simultaneously with the student. The goal is to dynamically adjust teacher-student interaction during the training process. Self-distillation involves distilling knowledge from a pre-trained model to itself, that is the same, potentially complex, architecture is trained to mimic itself but with a different dataset.

Regarding different types of knowledge being transferred, one can have response-based, feature-based and relation-based [19]. Response-based is perhaps the most natural when KD comes to mind, in which the student is trained to minimize the distance from its output (soft label) to the teacher’s output. Whereas feature-based knowledge transfer involves mimicking the intermediate process, typically by transferring information from hidden layers. Relation-based knowledge transfer focuses on capturing relationships across different samples or feature maps, namely by transferring information from input layers. Note that feature-based KD is most known as Transfer Learning. In natural language processing applications, relation-based KD is known as simply sharing the embedding matrix.

The intuition behind using KD as a defense strategy is to be able to reduce the complexity of publicly available target models (and consequently susceptible to privacy attacks) and access/dependency on private training data, which may reduce attack success in some cases. Gu et al. [20] used Pseudo Complementary Knowledge Distillation (PCKD) which reduces the training set of each teacher model and uses model averaging to generate soft targets of transfer data. Experiments prove that this approach could resist proposed attacks effectively, but at a cost of data utility and complexity, as it involves a non-negligible performance overhead.

2.3.8 Secure Multi-party Computation

Privacy-preserving mechanisms, such as *cryptographic techniques*, are designed to realize data utility while ensuring that original information will not be disclosed to other individuals or groups (Yin et al. [63]). Among several cryptographic techniques, Secure Multi-party Computation (SMC) is of major interest when designing a privacy-preserving FL protocol as its parties are mutually distrustful, so they do not allow their private data to be shared. SMC is a cryptographic scheme that enables distributed participants to *collaboratively calculate an objective function without revealing private data*. Secure Aggregation and Secure Shuffling are two examples of SMC techniques frequently used in FL. Secure Aggregation aims to calculate an aggregate value, such as the sum, through individual contributions of each party without compromising any private data (Bonawitz et al. [5]). The steps in Figure 2.4 must be taken when applying secure aggregation in the gradient updates' aggregation process. In particular, each pair of users u and v agree on a matched pair of input perturbations $s_{u,v}$ and $s_{v,u}$, which they exchange over their secure channel to then compute perturbations $p_{u,v}$ and $p_{v,u}$. These perturbations are added to the value they aim to hide from the central server (x_u and x_v respectively). Finally, the server simply sums the perturbed values to get the sum of x_u over any every user u .

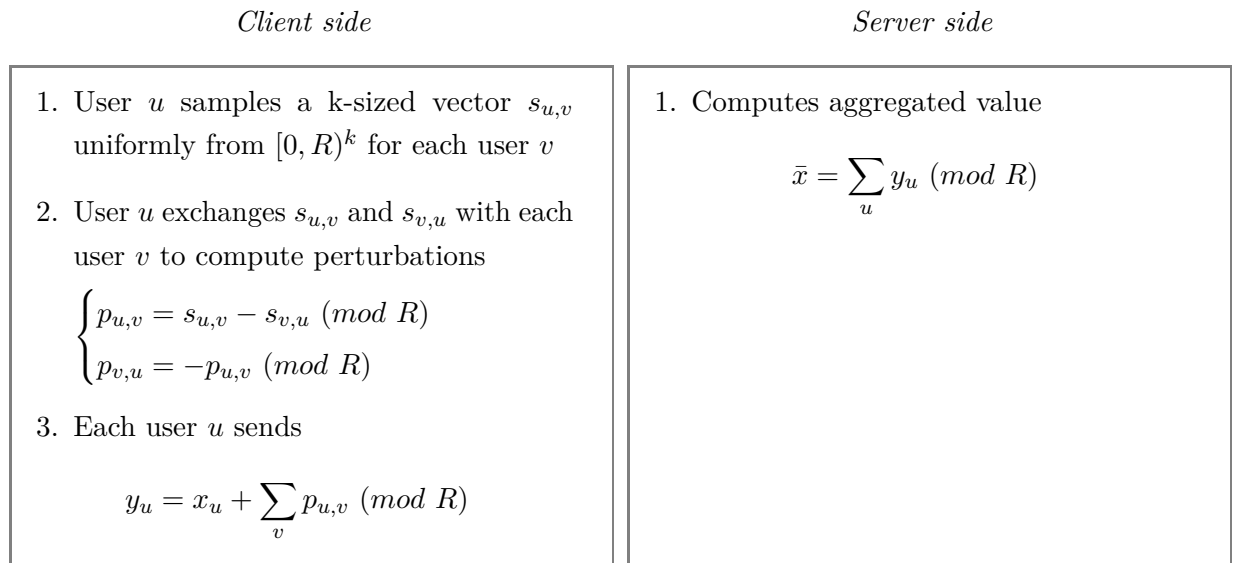


Figure 2.4: Application of secure aggregation on FL systems

Correctness of the Secure Aggregation procedure is guaranteed because paired perturbations cancel out as follows:

$$\bar{x} = \sum_u x_u + \sum_u \sum_v p_{u,v} = \sum_u x_u + \sum_u \sum_v s_{u,v} - \sum_u \sum_v s_{v,u} = \sum_u x_u \pmod{R}$$

More complex versions are needed if the server is not only curious but also malicious, for example when he lies to users about how many clients dropped out of training in each round. This method provides important security against possible unauthorized access or malware that might try to access everyone’s data. As a defense mechanism, secure aggregation prevents stronger attacks to federated models by reducing the central server’s access to only aggregated gradient updates. In other words, it ensures that an honest-but-curious server does not learn any private information, aside from what is learnable from the aggregate value itself. Secure aggregation on FL systems would mitigate some previously proposed attacks, for example a model inversion attack proposed Geiping et al. [18] and the Isolating attack (MIA) proposed by Shokri et al. [40]. Although, Zhao et al. [67] proposed an active model inversion attack that broke secure aggregation, therefore enabling the central server to have access to individual gradients. In this case, gradient updates were separable per client, and by definition, after the adversary’s modification on the target model’s architecture, sending a global model with different parameters to each client. Thus, on top of secure aggregation, a privacy-preserving FL system must restrict server’s access and possible actions.

2.3.9 Attack Detection

Detecting privacy attacks is an important feature of a privacy-preserving training or predictive system. Any attack detection scheme must continuously monitor the target model behaviour or interactions from users, participants or servers. Early detection, meaning identifying suspicious behaviour on an early stage (if applicable), is a major concern of this type of defenses, as well as an effective mitigation phase, since the detection itself does not protect anyone’s privacy. Minimization of false positives is also a concern, as in any other detection system.

Regarding privacy attacks detection, a very underexplored defense mechanism, a major distinction among these approaches is that they may target malicious users, by detecting malicious queries, or target malicious participants/servers, by detecting malicious modifications to the target model. Note that this distinction is motivated by the need to have each approach tailored for a specific type of attack regarding its impact on the target model. That is, passive attacks performed by outsiders (users) may be detectable through malicious query detection, whereas active attacks performed by insiders may be detectable through malicious modifications to the target model. An example of passive attack detection was developed by Ko et al. [30] to detect label-only membership inference and model inversion attacks. PrivMon, the detection system, operates as a stream-based system for real-time detection that continuously monitors a stream of queries, which does not require prior knowledge about the target model. For each

incoming query, the dissimilarity between current and past queries is calculated, where the average distance to the K nearest neighbours is taken as a dissimilarity measure. A query is classified as malicious if its dissimilarity measure, with past queries, is below a pre-set threshold. The intuition behind this attack is that malicious queries tend to be semantically similar as attackers usually generate a sequence of queries that are slight variations of each other, while benign queries are usually dissimilar as they typically represent a broader range of natural variations. Similarly, Blacklight (Li et al. [33]) and Stateful Detection (Chen et al. [11]) were proposed to defend against black-box evasion attacks, where their core idea is to detect evasion attacks via similarity among queries.

Currently, this defense mechanism is mostly used to detect cybersecurity attacks, such as backdoor attacks (Chen et al. [9]) and Denial of Service (Janardhana et al. [27]). Backdoor attacks are techniques that implant secret behaviours into trained models, which work as usual until the backdoor is triggered by specially crafted input provided by the adversary. Backdoor attacks can be extremely damaging by identifying anyone as a specific person in the security guard systems (Chen et al. [9]). In its turn, Denial of Service attacks (DoSs) are attempts to initiate an enterprise or IoT device to a crawl, keeping the system unavailable to its legitimate users (Janardhana et al. [27]). However, to the best of the author’s knowledge, there are no approaches to detect active privacy attacks.

2.3.10 Towards Local Defense Mechanisms

In a FL system, clients may use a defender application to protect their privacy. Ideally, such application should work regardless of its knowledge about the target model, as well as private data to reduce compromised defenders’ likelihood of violating user’s privacy (Jia et al. [29]). Additionally, it must also use algorithms that do not require a lot of computational resources given that clients’ device on a FL system may be as simple as a smartphone.

Aiming to design a computationally tractable defense mechanism with small utility loss, Jia et al. [29] uses evasion attacks for the first time as a defense technique to mitigate AIAs. Particularly, the defender is a software on the user’s side. After permission is granted, the defender can modify user’s public data. This is an especially smart setting because, by being on the user’s side, it can be a *customizable* defense mechanism. Truly an improvement on prior work which simply aims to defend a model as a whole. Despite being adequate for membership inference (in which the definition of privacy violation is homogeneous), privacy violation of attribute inference depends on the attribute’s real sensitivity. Because neither privacy nor attribute sensitivity definitions are the same for everyone, customizable solutions seem to be a relevant future direction. Particularly in FL scenarios, where users have total control over their private training data and can thus apply different defense mechanisms according to their preferences, accounting for malicious, or simply curious, servers and participants.

In the setting of Jia et al. [29], the defender’s goal is to *find a randomized noise addition mechanism that minimizes attack success*, that is to reduce an attack to the baseline (for example

uniform distribution over the sensitive attribute’s possible values). A challenging goal since neither the attack model nor the true attribute value are known. Therefore, a tractable goal is to find a minimum noise vector for each possible attribute value such that by adding it to a data point an attack model mispredicts the sensitive attribute value (evasion attack). Moreover, several *customizable privacy policies*, available through the defender’s application, can be selected by the user, according to its preferences. These consist of different forms of adding noise: add a new entry, modify an existing one or both, with the latter one being the most effective in terms of attack mitigation. Empirical results show that *Attriguard*, the proposed approach, is highly effective at defending against AIAs - it significantly reduces attack accuracy while incurring in minimal utility loss to users’ public data. *Attriguard* paves the way for local defense mechanisms that must focus on being effective and efficient while respecting user’s privacy and preferences.

2.4 Attack Performance Metrics

An adversary may have distinct goals, all of which constitute a violation to the privacy of data owners. However, measuring these threats is not a trivial task, particularly if one is interested in measuring to what extent model release helped attackers inferring private information. For that reason, the following section focuses on metrics to measure each privacy attack performance.

2.4.1 Measuring MIA performance

Membership inference succeeds if the attacker can correctly guess a sample’s membership status. As a binary task, one could think that by classifying all target dataset samples as members an adversary would catch all true members in that set. But what type of adversary wants to commit so much error (false positives)? Moreover, such an attack would not prove that the target model is vulnerable to MIAs, as the adversary would not need a trained model to make that prediction. As any other hypothesis test, there are two types of error to control. Here, controlling the amount of non-members that are classified as members is far more important than controlling the amount of members that are classified as non-members, as reducing this type of error is trivial.

Accuracy is the most commonly used metric, but it does not measure attack success fairly, as it depends on the proportion of members and non-members and on a threshold, for both metric and model-based attacks (in which the predicted probability is also compared to a threshold). To account for False Positive (FP) minimization and True Positive (TP) maximization simultaneously, precision seems to be a fair evaluation metric. However, since it also depends on absolute values of TP and FP, it can be misleading due to members and non-members imbalance. With the purpose of considering several thresholds and removing the dependency on members and non-members proportion, Area Under the Receiver Operating Characteristic Curve (ROC AUC) is another commonly used metric to measure attack success. ROC AUC is the probability that a classifier will be more confident predicting the positive label for a member than for a non-member. Closely

related to this metric, is the ROC curve offering a visualization of True Positive Rate (TPR) as a function of False Positive Rate (FPR). For systematic evaluation purposes, ROC AUC is often preferred as it is easier to compare. And if accounting for larger FPRs is a problem, one can restrict the calculation of the area under the curve for a curve that does not allow for FPR larger than a threshold (restricted ROC AUC).

Considering other metrics based on TPR and FPR, Yeom et al. [62] proposes Membership Advantage, $TPR - FPR$, which measures how well an adversary can distinguish samples of the training dataset from samples coming from the general population after being given the model. The distinction factor of this definition regarding others that account for TPR and FPR is in the fact that the attacker does not classify a data point as a non-member, it simply assumes that there is not enough evidence to conclude that it is a member. Therefore, the adversary must assume that the sample comes from the general population from which training and non-training samples are drawn. Nguyen et al. [42] also proposed to use the difference between TPR and FPR to measure MIA success. Closely related to this definition, at least in the sense that both use TPR and FPR, is the evaluation metric proposed by Carlini et al. [7]. Since both FPR and False Negative Rate (FNR) are relevant, TPR at low (fixed) FPR is used to evaluate attacks. Note that TPR is a function of FNR, particularly $FNR = 1 - TPR$. Authors define as a successful MIA one that can reliably violate privacy, even for a few users, and as a failed MIA one that only unreliably achieves high success rate. Meaning, if it only reaches high TPR by letting FPR be at least as high.

2.4.2 Measuring AIA performance

Attribute inference succeeds if the attacker can correctly determine the sensitive value of a single, or multiple, training data records. Intuitively, accuracy seems to be a good metric. Despite being frequently used, it has its limitations, namely it does not measure training data risk but rather what can an adversary infer correctly regardless of the model's additional leakage.

Yeom et al. [62] is mostly concern in determining an adversary's advantage in inferring sensitive attribute values of training samples. For that purpose, an *attribute advantage* metric is defined as the *difference between the probability of accurately guessing the sensitive attribute value of members and non-members*. Attribute advantage is 0 if an AIA is equally powerful on train and test data and is maximized when the attack only works on training data. However, this definition has a disadvantage: an adversary that can perform membership inference can fool this score by performing poorly on non-members. An alternative advantage defines attribute advantage as the difference between probability of accurately guessing sensitive values of members based on the target model and the probability of doing so for non-members using only prior knowledge. This definition seems to be motivated by a common practice in ML: compare model accuracy with what a dummy, or random, classifier would achieve. However, it still has a disadvantage: does not consider what does the target model reveal about non-members, that is general information about training data distribution.

Correctly capturing training data vulnerability to this type of attack is of major interest as it bounds additional model leakage. Meaning, if training samples are especially vulnerable to AIAs after model release. If non-members are as vulnerable as members, then the model does not leak any additional information that was not already present in public available knowledge. This insight is further explored in Jayaraman et al. [28].

2.4.3 Measuring PIA performance

Property inference succeeds if the attacker can correctly guess whether a model was trained with $t_0\%$ or $t_1\%$ of positive (with respect to the property) samples. In this case, accuracy is the most commonly used metric, and it is a suitable metric if the adversary is equally interested in correctly predicting both labels. On the contrary, if the adversary is most interested in detecting when a model is trained with one of the proportions (either $t_0\%$ or $t_1\%$), then other metrics maybe considered, such as precision. ROC AUC is also a suitable metric, with the advantage of being threshold independent. Moreover, property inference also succeeds if the attacker can estimate the real proportion with small error and for which mean squared error is the most common metric. However, the lack of an established goal leads to the lack of an established metric to measure PIAs threat.

To tackle this limitation, Suri et al. [53] studies *how hard it is for an attacker to reach a given accuracy* by relating inferred information from the model to what it would learn by sampling from the training distribution. An alternative metric is also proposed for attacks that aim to estimate the proportion, rather than distinguishing between $t_0\%$ and $t_1\%$. The most powerful adversary would be one that can sample (or reconstruct perfectly) the whole, or a part, of the training data distribution. In such cases, property inference is reduced to a direct estimation of the property proportion of reconstructed samples. For that reason, n_{leaked} measures the number of samples an adversary would need to estimate property proportion with the same accuracy. Intuitively, the harder the task the more samples one would need to estimate property proportion confidently.

Correctly capturing PIAs threat is of major interest since it allows for one to see the extent to which the attack formulation itself contributes to its accuracy. Naturally, for a fixed target model, simpler tasks will lead to greater accuracy without it necessarily mean that the model actually leaks a significant amount of information. In fact, observed small values of n_{leaked} for every distinguishing/estimation task shows that there is *little leakage*, i.e. the attack accuracy is mostly determined by how easy is the task at hand and not so much by the target model leakage expressed in terms of n_{leaked} . This means that the threat of previous PIAs may be exacerbated, thus paving the way for future directions to improve PIAs - improve their threat not their accuracy.

2.4.4 Measuring RA performance

Unlike MIAs for which TPR and FPR, or a function of both, better express attack performance, reconstruction attacks are harder to evaluate. When there is a one-to-one correspondence between reconstructed and private training data (MI), one may use Peak Signal-to-Noise Ratio (PSNR) to measure the quality of reconstructed data. On the other hand, when the goal is to generate data from the same distribution as private training data (REPR), not necessarily obtain specific training samples, there is a lack a quantitative evaluation metric. So far, such attacks have been evaluated by a qualitative analysis (Zhu et al. [68]), disabling fair and objective comparison between different attacks.

Tanaka et al. [54] aims to fill that gap by proposing an evaluation metric based on the distance between two probability distributions (real and inferred training data distribution) using Kullback-Leibler Divergence (KL-divergence). A simple interpretation of KL-divergence between two distributions is the expected excess surprise from using one distribution as a model when the actual distribution is the other. This metric is most suitable for attacks that lack one-to-one correspondence between reconstructed and training data, as the conditional-GAN based reconstruction attack that aims to reconstruct training data distribution of a particular class. According to the categorization of reconstructed attacks, explained in Section 2.2.2.4, this would be a class representative inference attack. Thus, the goal of this metric is to measure how well the generator approximates private training data distribution. Experiences with real data, from EMNIST dataset, show that there is a direct relationship between the value of KL-divergence and the visual quality of images - for higher values of KL-divergence one cannot identify the respective digit.

2.5 Summary

This chapter covers what has already been explored, what needs to be improved and what is yet to be discover in the context of privacy-preserving machine learning. To summarize each type of privacy attack (and respective solutions) on ML models, Table 2.1 presents several parameters that describe each privacy attack, namely goal of the attack, example application, approach, requirements and potential defenses. In this table, the relationship between MIA and other attacks, such as AIA and PIA, is clear in the column of defenses, given that defenses to mitigate MIA also work for AIA and PIA. Moreover, the difference between inference and reconstruction attacks is highlighted by the requirement column, in which is clear that inference attacks require more prior knowledge about the training data distribution.

Additionally, to summarize each reviewed paper's contribution to this broader picture of privacy attacks and defense mechanisms, Figure 2.5 contains an assessment based on attack parameters (starting with A . in the figure) and defense-wise parameters (starting with D .), according to the following parameters for a systematic assessment of research works in this area:

Goal of Attack	Example Application	Approach	Requirements	Defenses
MIA (infer whether a given sample was used in the training process)	Determine if a patient's data was used to train a model for assisting with diagnoses.	Exploit distinct model behaviour depending on the membership status.	At least black-box access, overfitted (globally or locally) target model, access to some training data.	Reduce overfitting and influence of each sample, input/gradient/response perturbation.
		Encode membership status in model parameters.	White-box access, access to some training data.	
AIA (infer a training sample's sensitive attribute value)	Discover the political view of a user to deliver personalized advertisements.	Exploit the correlation between model performance/-parameters and sensitive predictive attribute value.	At least black-box access, knowledge about the sensitive attribute distribution, access to some training data.	Mitigate MIA, reduce unintended overlearning.
		Exploit unintended overlearning of non predictive features.		
PIA (infer a sensitive property of the whole training dataset)	Discover if a person is present in another ones' image folder, where the property being inferred is the presence of the person	Exploit the target model's bias towards its training data for different proportions of the property.	At least black-box access, access to the training data distribution.	Reduce overfitting (mitigating population-level MIA), reduce unintended overlearning.
		Exploit the correlation between model parameters and proportion of the property.	White-box access, access to the training data distribution.	
MI (reconstruct training samples, part of RA category)	Recover real images from embeddings from a DB or in a distributed learning scenario.	Exploit the correlation of data records with output/gradients, via optimization or direct leakage.	Often require white-box access and unrealistic adversarial setups.	Reduce/perturb response/gradients, ensure minimum number of local epochs/clients in FL systems.
REPR (reconstruct training distribution, part of RA category)	Generate images that are indistinguishable from real ones.	Train GAN to generate samples from the target distribution.	Often require white-box access and unrealistic adversarial setups.	Reduce target model accuracy.

Table 2.1: Summary of privacy attacks on ML

- *A. Effectiveness*: ability of the proposed attack to infer information (e.g. membership status and sensitive attribute value);
- *A. Efficiency*: stands for how efficient that attack is in terms of computational power;

- *A. Realistic* stands for how realistic adversarial conditions/assumptions are, for example attacks on FL systems that required access to individual updates were marked as not realistic;
- *A. Accurately Measured* stands for the suitability of the evaluation metric used, for example MIAs that used accuracy to measure attack performance were penalized;
- *A. Connection Between Attacks* denotes how well the connection between different privacy attacks is explored, most papers focus on a single type of attack;
- *D. Effectiveness* stands for the ability of the proposed defense mechanism to mitigate attacks;
- *D. Efficiency* stands for how efficient that defense strategy is terms of computation and communication complexity;
- *D. Low Model Utility Loss* stands for balanced trade-offs between privacy and model utility, defense mechanisms that incur in high model utility loss, such as differential privacy, were penalized;

State Of The Art - Evaluation

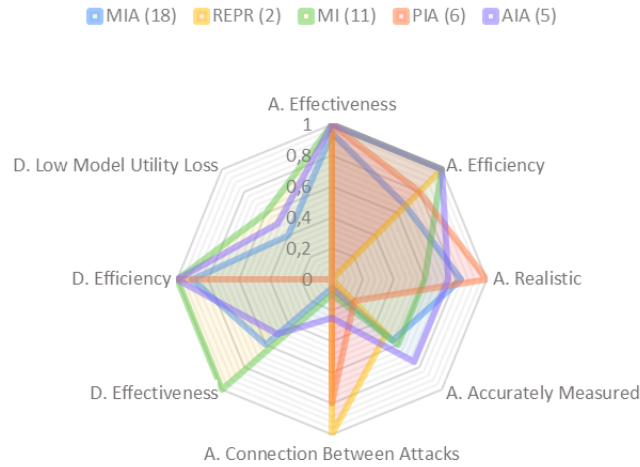


Figure 2.5: State of the art attack and defense evaluation per privacy attack. Scores in the range $[0, 1]$ express the average grade per privacy attack, whereas the number of papers contributing to each score is inside parenthesis in the figure caption.

From this figure, one can see that, for every privacy attack, parameters such as *accurately measured* and *connection between attacks* are the most disregarded, especially among MI and MIAs, as illustrated by the low number of papers positively evaluated in these parameters (MIA with 9 and MI with 6 accurately measured papers and 1 exploring connection between attacks for both MIA and MI). *The realistic nature of attacks was also disregarded*, especially among MIs

and REPR with 6 and 0 positively evaluated papers. Note that attack effectiveness is harder to evaluate since if the proposed approaches were not effective then probably no paper would be published. Moreover, results depend on practical settings that are certainly tuned conveniently. The lack of a uniform evaluation methodology also hinders critical evaluation and comparison of results.

Effective and with low model utility loss defenses are also missing from almost every privacy attack. Regarding effective defenses, only 6 out of 10 MIA papers that evaluated defense strategies were positively evaluated in this parameter, 0 out of 2 PIA papers, and 2 out of 4 AIA papers. Regarding low model utility loss defenses, only 3 out of 9 MIA papers that evaluated defense strategies were positively evaluated in this parameter, 0 out of 2 PIA papers, and 2 out of 4 AIA papers.

Focusing on the content of papers about MIA, passive MIAs, that exploit differences in model response depending on the membership status, are usually applied over extremely overfitted models (for example in Shokri et al [40]), which are far away from the best models trained on those well-known public datasets. This observation suggests that only overfitted models are vulnerable to those type of attacks (or at least are considerably more vulnerable). In other words, from this observation, it should be expected that well-generalized models provide some level of security. However, this may be a false sense of security as attacks that do not rely on overfitting (e.g. Nguyen et al. [42]) should be able to infer sensitive information regardless of the target model’s generalization ability. In fact, there has been little research on active attacks, as there are only 4 MIAs compared to 15 passive ones (Figure 2.2.5). Active attacks may be under-explored due to their potentially strong impact on the target model, therefore being easily detectable. But there are still no approaches to detect active privacy attacks (as far as the author’s knowledge goes). Thus, these threats are real and are not being addressed, since defenses to improve generalization do not mitigate attacks that do not rely on it and no proper attack detection strategy has been proposed.

This critical evaluation (Figure 2.5) also highlights a trend regarding the literature on this topic - *there is a tendency to focus on developing attacks, leaving defenses under-explored*, judging by the amount of papers evaluated in terms of defense parameters (Table A.1). Even though identifying new vulnerabilities is a relevant research topic, developing defense mechanisms to address those vulnerabilities is even more important, as adversaries evolve faster than countermeasures. Moreover, most defenses were preventive, meaning that the goal of reducing adversarial advantage is included on the training process by default (e.g. adversarial training, which involves using a combined loss function to simultaneously reduce main task error and adversarial advantage), regardless of the suspicion of any dishonest behaviour. Although relevant to deal with passive attacks (in which the adversary follows the protocol), these are expected to affect model utility significantly as, in some cases, the task of reducing the advantage of a potential adversary leads the whole training process, leaving the main task to the background. Thus, the key to reduce model utility loss from the application of a defense strategy may be to only activate the mitigation phase when necessary. Albeit relevant, the focus on developing preventive defenses leaves a gap

on research about reactive ones, that only mitigate attacks when an alarm for suspicious activity is triggered (e.g. attack detection).

Finally, almost all papers focus on a single type of attack, leaving the connection to other types under-explored. This is particularly relevant on the context of MIAs, which are the most common privacy attack among the literature and whose connection to other attacks, such as reconstruction, is not well understood. Moreover, the connection between PIAs and MIAs also lacks an empirical validation, as it was only briefly mentioned by Parisot et al. [43] (as far as the author’s knowledge goes). These limitations are not addressed by this thesis, although the connection between membership and attribute risk is studied with the goal of showing that membership risk is an upper bound for other privacy risks.

Given all these limitations, research on active and realistic attacks, that do not rely on overfitting, while being accurately measured is of special interest. Even more so if their connection to other privacy attacks is also studied. Furthermore, effective defense strategies, namely reactive ones to address the underestimated threat posed by active attacks, and that incur in low model utility loss are urgently needed to cover the gap of defenses in general.

Chapter 3

Membership and Attribute Inference

Privacy attacks pose a significant threat to models. While federated learning decoys from the need of centralized data for training, it also introduces additional vulnerabilities compared to centralized learning. Clients in Federated Learning (FL) are exposed to powerful *active* attacks that exploit model complexity rather than relying on overfitting (Nguyen et al. [42]). This is an even more serious threat than commonly designed passive Membership Inference Attacks (MIAs), for example, and for which no defense strategy has been proposed. This chapter introduces and evaluates active membership and attribute inference attacks aiming to demonstrate that well generalized models carry a false sense of security, as they are exposed to these threats.

In this chapter, novel active attacks are proposed such that membership and attribute information are directly encoded into maliciously modified neurons. Several passive MIAs, as well as a MIA-based Attribute Inference Attack (AIA), from the state of the art, are implemented as baselines to assess the effectiveness of active inference attacks. Based on Section 2.5, active inference attacks address previously identified limitations. In particular, the *non-realistic attack* limitation is covered by the fact that the central server only has access to aggregated updates, and through a challenging evaluation at the end of this chapter. The quality of the shadow dataset, either in size or in distance from the training dataset, should also be realistically tuned, since a valuable approach is one that meets expected privacy risks. Moreover, the *lack of a suitable attack performance measure* is also addressed by considering True Positive Rate (TPR) and False Positive Rate (FPR), or a function of both, as well as with a practical assessment of the real threat of active attacks. Finally, prior work’s lack of an understanding about the relationship between different privacy attacks is tackled by a theoretical and empirical comparison of membership and attribute inference attacks performance.

3.1 Novel Active Inference Attacks

To accomplish these main contributions: passive membership inference attacks, used as baseline, are developed and explained in Section 3.1.2; an active membership inference attack is explored,

and for which its conditions and methodology are explained in Section 3.1.3; a new active attribute inference attack is developed, and for which its conditions and methodology are explained in Section 3.1.4; finally an attribute inference attack based on the proposed MIA is developed and explained in Section 3.1.5. To help the reader understand these attacks before diving into details, an overview of how these novel active attacks work is presented in Section 3.1.1.

3.1.1 Overview

This section outlines the methodology behind both active membership and attribute inference attacks, as well as MIA-based AIAs (as long as they are based on the same active MIA). Figure 3.1 illustrates the general methodology of active attacks performed by the central server in a FL setting. On the top of this figure, a common FL system is represented in which several participants locally train a global model which was previously sent by the central server, presumably a benign model. After local training, each client sends its local model to the central server. However, Secure Aggregation (Section 2.3.8) stops the central server from receiving individual local models (also called gradient updates). These local models are all aggregated safely without the intervention of any entity from this system, as illustrated in the figure by the Aggregation step node.

At some point of this training process, the central server performs an active attack in which it locally changes the global model so as to leak information. The bottom of this figure explains those malicious modifications. Firstly, a neuron from the second fully connected layer is selected (outlined as a red circle node) whose weights/edges will be the initialization of the next step. Note that by selecting a neuron from that layer, weights/edges connecting the input layer to the first fully connected layer are also collected. Secondly, starting from that initialization, an adversarial network is trained, with the shadow dataset, so as to leak membership or attribute information about a target sample. However, the adversary must wait for another training round (of the FL system) to complete this attack. So, it replaces benign weights/edges (from the first and second fully connected layer) by these maliciously trained weights/edges (in red). Then, this global model is sent to clients which in their turn sent it back to the central server (after local training). Information leakage occurs in this step, when the central server observes the difference (also called gradient update) between the previous (malicious) and the current global model, particularly in the selected neuron (node filled in red). The inference methodology includes analysing the selected neuron's nullity or sign, depending on the type of attack, which is further explained latter in the following sections.

3.1.2 Passive Membership Inference

In order to prove that active attacks are much more powerful than passive attacks, three passive MIAs are implemented as baseline based on Shokri et al. [40, 61]: Black-box Loss-based (BBL), White-box Gradient-norm-based (WBG), and White-box Improved Gradient-norm-based (WBIG). This section explains the attack methodology of all of these approaches,

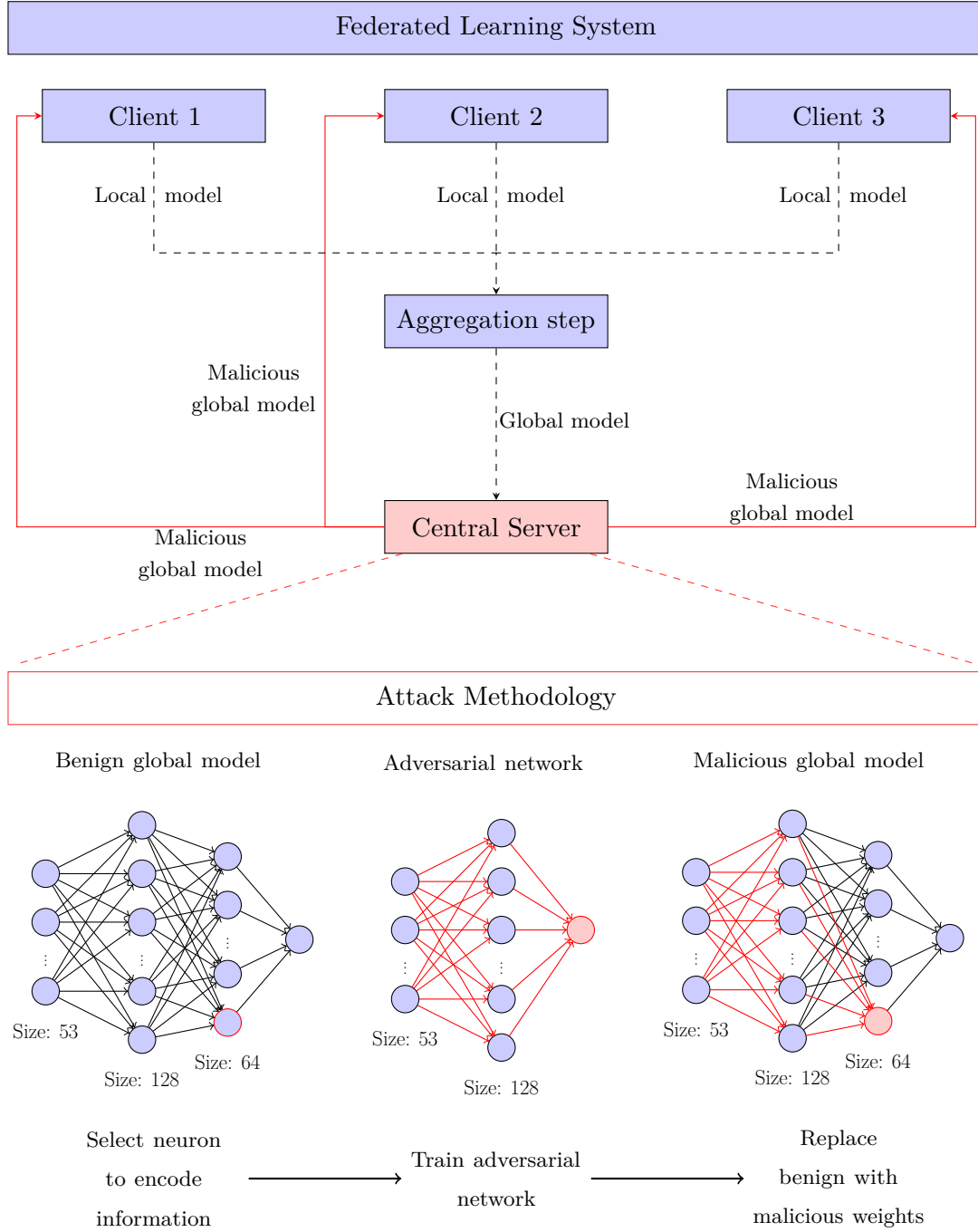


Figure 3.1: Overview of novel active attacks performed by the central server on a FL system.

respectively in sections 3.1.2.1, 3.1.2.2, 3.1.2.3, while Section 3.2 presents the results.

3.1.2.1 Black-box Loss-based (BBL)

Based on Shokri et al. [61], particularly its model-dependent MIA, Black-box Loss-based (BBL) is a metric-based on loss attack that aims to distinguish target model's behaviour on members

from non-members based on the loss distribution. Generally, smaller values of loss are more frequently associated with training data. Analogously, higher values of loss are more frequently associated with non-training data. However, easy to fit non-training samples may have small values of loss and hard to fit training samples may have greater values of loss. An adversary of this kind must be willing to incur in some error and must find a balance between specificity and sensitivity.

The methodology is quite simple. After collecting the loss of target samples, the adversary finds the loss distribution by ordering those values. Then, the inference phase consists of classifying those target samples whose loss is lower than a given threshold as members. Finding an optimal threshold is key for an attacker to succeed at this type of attack. In a practical scenario, the adversary would have to estimate this threshold, for which the most common technique to use is shadow training. However, from an audit point of view, one must evaluate to what extent is the adversary able to infer membership regardless of the threshold. Thus, no shadow models are trained to estimate the optimal threshold and a threshold-independent metric is used to assess attack performance.

Note that this attack can be performed by the central server, by a participant or by an external agent with access to the model's confidence. Additionally, it is performed during the inference phase of the target model and on both centralized and federated models. Thus, using the taxonomy (Figure 2.3), one would classify BBL as:

- **Goal:** Membership
- **Target:**
 - **Phase:** Inference
 - **Model:** Any
- **Adversary:**
 - **Identity:** Any
 - **Access:** Black-box
 - **Effect:** Passive
 - **Approach:** Metric based (loss)

3.1.2.2 White-box Gradient-norm-based (WBG)

Most white-box attacks improve upon black-box due to the ability of collecting more information. Particularly, gradient updates are especially interesting from an adversary's point of view as they reveal the magnitude of the change as well as the direction. Thus, gradient-based attacks are expected to perform better than loss-based, even though gradient's magnitude is correlated with loss's magnitude. However, gradient's dimension may hinder its analysis, so one may

take gradient norm as a summary of the information provided by gradients. Other forms of dimensionality reduction are also possible, such as using a Convolutional Neural Network (CNN) to retrieve the most relevant information, as in Shokri et al. [40]. Inspired by Shokri et al. [40], White-box Gradient-norm-based (WBG) is a metric-based attack that aims to distinguish the target model's behaviour on members from non-members based on gradient norm distribution. Similarly to the loss, smaller gradient norm is frequently associated with members and greater gradient norm with non-members. As in BBL, an adversary of this kind must be willing to incur in some error and must find a balance between specificity and sensitivity.

The methodology is similar to the previous case, but instead of collecting loss values, the adversary collects the gradient norm for each target sample and estimates the gradient norm distribution. The inference phase consists of classifying those target samples whose gradient norm is lower than a given threshold as members. Just as in the BBL attack, finding an optimal threshold is key to succeed at this type of attack. For the same reasons, shadow models are not trained but would be needed to estimate the threshold.

Realistically, WBG can only be performed by insiders of a FL system, since a privacy-preserving centralized model would only make the output layer public (query/black-box access). Thus, using the taxonomy (Figure 2.3), one would classify WBG as:

- **Goal:** Membership
- **Target:**
 - **Phase:** Inference
 - **Model:** Federated
- **Adversary:**
 - **Identity:** Insiders (participant and server)
 - **Access:** White-box
 - **Effect:** Passive
 - **Approach:** Metric based (gradient)

3.1.2.3 White-box Improved Gradient-norm-based (WBIG)

As highlighted in Chapter 2, most model-based attacks improve upon metric-based as they enable high-dimensional input to be processed, as well as a non-linear decision boundary to be drawn. Additionally, in FL systems, an insider adversary usually improves what an outsider can do due to the ability of collecting information along the training phase. White-box Improved Gradient-norm-based (WBIG) is a model-based attack that aims to distinguish the target model's behaviour on members and non-members through gradient norm along the training rounds. The ultimate goal is to explore continuous leakage in FL systems. This attack is inspired by Shokri

et al. [40] and Gu et al. [20] which also proposed MIAs that leverage continuous leakage in FL systems although processing the sequence of gradient norms per sample in a different way.

Contrary to the previous passive attacks, WBIG requires training an adversarial network to perform inference taking as input the sequence of gradient norms of each target sample, collected during the training process, to then output membership probability. To take advantage of the temporal dependency between gradient norm values at each training round, the adversary must start by feeding each input sequence to a Recurrent Neural Network (RNN). RNNs are designed to process sequential data by maintaining a *memory* of previous inputs through internal states. Particularly, Long Short-Term Memory (LSTM) networks, an improved version of traditional RNNs, are used to overcome the long-term dependency problem. The LSTM yields a feature representation vector of the input sequence, which is then processed by a traditional Neural Network (NN) that outputs the membership probability. Note that WBIG may improve upon prior work by using LSTM to process the sequence of information collected along the training process.

At this point, the adversary aims to succeed at an unsupervised learning task. Similarly to several attacks from prior work, the central server uses its shadow dataset to train the attack model in a supervised environment. For that purpose, the adversary trains a FL model (shadow model) using the membership status of each shadow sample, with respect to the shadow model, as the target variable for the attack model (LSTM based one). Finding an optimal adversarial network's architecture and hyperparameters is key to succeed at this type of attack.

Using the taxonomy (Figure 2.3), one would classify WBIG as:

- **Goal:** Membership
- **Target:**
 - **Phase:** Training
 - **Model:** Federated
- **Adversary:**
 - **Identity:** Insiders (participant and server)
 - **Access:** White-box
 - **Effect:** Passive
 - **Approach:** Classifier based

3.1.3 Active Membership Inference (AMI)

In order to show that well-generalized models carry a false sense of security, even against a realistic adversary, a powerful active white-box attack is designed to leak membership information. Based on Nguyen et al. [42], AMI is an *active white-box* attack, performed by the central server,

which can be considered *metric-based*, since its inference phase is based on a specific neuron's gradient. The intuition behind this attack is to *encode membership status* of each target sample in a specific neuron, which should be decoded in the gradient update of that neuron. Neuron's weights are trained such that the neuron is only activated by the target sample, therefore any non-zero update of the malicious neuron reveals that the target sample was used during training. The intuition behind this approach is to overfit the malicious neuron towards the target sample. That is, the malicious neuron will draw a linear boundary that separates the target sample, in a higher-dimensional representation of it (from passing through previous layers of the network), from the rest of the training samples.

Using the taxonomy (Figure 2.3), one would classify AMI as:

- **Goal:** Membership
- **Target:**
 - **Phase:** Training
 - **Model:** Federated
- **Adversary:**
 - **Identity:** Central server
 - **Access:** White-box
 - **Effect:** Active
 - **Approach:** Metric based (gradient)

In this section, the reader can find conditions that enable AMI execution (3.1.3.1) and explained attack methodology (3.1.3.2). Empirical results are presented in Section 3.2.

3.1.3.1 Requirements

For AMI to work, other than being a neural network, the target model must apply Rectified Linear Units (ReLU) to the second fully connected layer from which a neuron will be selected to leak membership status. Therefore, it is also required that the target model has at least 2 fully connected layers. Actually, the target model can apply any activation function $f(x)$ that satisfies the following condition:

$$\begin{cases} \frac{\partial f}{\partial x} = 0 & x < 0 \\ \frac{\partial f}{\partial x} > 0 & x \geq 0 \end{cases}$$

Where $x = \sum_j W_{ij} * z_j + b_i$ is the logit/linear combination of inputs of the selected neuron i for sample z .

To perform the attack, the adversary requires access to the target sample, although not its label. Moreover, the central server must have a shadow dataset with which neuron's weights are trained to be deactivated. Adversarial knowledge is limited to this set, therefore any statistical information, such as distribution of an attribute, is calculated based on this dataset, hopefully not significantly diverging from clients' data. However, the adversary *does not require access to individual updates*, nor it imposes local training restrictions regarding batch size and number of local epochs. It is worth mentioning that, since AMI is based on gradients, a perfect prediction, i.e., the model output 100% matches the label, would result in a gradient of 0, resulting in the wrong inference. However, in practice this is unlikely to happen, given that the target model is reduced to a random guessing level (right after the attack) as one shall see latter on this document. Due to these fair and attainable assumptions, AMI is a realistic attack, and, in contrast with some from the literature (c.f. Chapter 2), it works regardless of the use of secure aggregation.

3.1.3.2 Methodology

Methodology of novel active attacks, in which AMI fits, was briefly outlined in Section 3.1.1, thus specific details are explained in this section. In a FL system, a malicious server executing AMI selects a neuron to be modified and collects its weights after the aggregation clients updates, that is from the new global model. When the adversary aims to target more than one sample, it can do so by selecting as much neurons as the number of target samples. An adversarial network is created with the goal of training those weights to leak precise membership status. This network must share the same input size and first hidden layer size with the global model architecture, as well as their respective activation functions. Its second, which is also the last, layer is the number of selected neurons plus an additional neuron that non-members must activate. In this way, what could be a regression problem is turned into a classification problem, that reveals the sample presence or absence. In addition, this classification setup enables the adversary to use weighted loss, which is determinant for AMI success, replacing the need of duplicating target samples. Sigmoid is used as the output function as it maps logits to $[0, 1]$.

After the adversarial network is optimized, the server replaces benign weights for maliciously modified ones in the global model, which is then sent back to clients. The resulting global model is then updated with aggregated gradient updates which leak membership status. Therefore, the *theoretical inference phase* consists of determining if aggregated updates for the selected neuron are *non zero*, if so then the target sample is predicted as a member.

3.1.4 Adapting AMI as an Active Attribute Inference (AAI*)

Building upon AMI from Section 3.1.3, this section presents a novel AAI* attack which is an adaptation of AMI to encode attribute information, improving upon MIA based AIA. At the best of the author's knowledge, this is a new approach. AAI* is an *active white-box* attack, performed

by the central server, which can be considered *metric-based*, like AMI. The intuition behind this attack is to *encode attribute information* of each target sample in a specific neuron, which should be decoded in its gradient update. The selected neuron’s weights are trained such that it takes a different value according to the sensitive attribute value and such distinction must hold in the gradient update, enabling attribute inference. In other words, the malicious neuron’s weights must draw 3 linear boundaries to separate: (1) non target samples from the target sample regardless of its attribute value; (2) the target sample with a positive attribute value from the same sample with a negative attribute value and from non target samples; (3) the target sample with a negative attribute value from the same sample with a positive attribute value and from non target samples. Note that this approach is restricted to binary sensitive attributes. Similarly to AMI, these linear boundaries are not based on the original input representation, but on a higher-dimensional representation from the following layer.

Using the taxonomy (Figure 2.3), one would classify AMI as:

- **Goal:** Attribute
- **Target:**
 - **Phase:** Training
 - **Model:** Federated
- **Adversary:**
 - **Identity:** Central server
 - **Access:** White-box
 - **Effect:** Active
 - **Approach:** Metric based (gradient)

In this section, the reader can find assumptions that enable the AAI* execution (3.1.4.1) and explained attack methodology (3.1.4.2). Empirical results are presented in Section 3.2.

3.1.4.1 Requirements

For AAI* to work, other than being a neural network, the target model must apply the Exponential Linear Units (ELU) activation function, to the second fully connected layer from which a neuron will be selected to leak sensitive attribute information. Therefore, similarly to AMI conditions, it is also required that the target model has at least 2 fully connected layers. Particularly, ELU as an activation function only allows AAI* to work for a *negative* α , which is further explained in the following Section 3.1.4.2.

Similarly to AMI conditions, *no non-realistic FL protocol conditions are imposed*, although the adversary needs the target sample input and label (as explained in Appendix B). Note that

AMI only required the input, however requiring the label, in that case, must not be a problem, as a sufficiently accurate target model would most likely predict the correct label. Moreover, the central server must have a *shadow dataset* with which neuron's weights are trained to have a specific value. Similarly to AMI conditions, adversarial knowledge is limited to this set, therefore the sensitive attribute distribution from the shadow dataset is assumed to be the same as in clients data. Considering these conditions, AAI* is a *realistic* attack, as it is an adaptation of an equally realistic one. It is important to note that even though ELU with negative α is required, that does not mean that AAI* is less realistic, as it is still a reasonable and expected adversary from which FL solutions must be protected. In fact, given that the adversary is the central server, it might have permission to choose the initial model architecture, thus requiring 2 FC layers and ELU as the activation function should not be a problem. However, the fact that this attack is limited to binary attributes may reduce its applicability.

3.1.4.2 Methodology

Methodology of novel active attacks, in which AAI* fits, was briefly outlined in Section 3.1.1, thus specific details are explained in this section. In a FL system, a malicious server executing AAI* selects a neuron for each target sample and collects its weights after the aggregation of clients' updates. An adversarial network is created with the goal of training those weights to leak the target sample's sensitive attribute value. Regarding adversarial network hyperparameters, they must match the target model's hyperparameters, as in AMI.

The major difference between AMI and AAI* methodology is in how this adversarial network is trained and, consequently, its inference phase. A modified neuron only leaks the target sample's attribute value if for different logits correspond different gradients, particularly one needs to have *three clearly distinctive gradient values*. AMI only required two: zero and non-zero gradients. But AAI* requires three to distinguish between non-targets, targets with positive and targets with negative attribute value. Therefore, one must infer the target sample's attribute value based on gradient updates *sign* - positive, negative, and null.

A target sample with a positive attribute value must have a positive logit, a target sample with a negative attribute value must have a logit close to zero (from the negative side) and a non-target sample must have a negative logit as far from zero as possible. The next set of equations explain the reason behind these choices.

$$ELU(x) = \begin{cases} x & x > 0 \\ \alpha(e^x - 1) & x \leq 0 \end{cases} \quad \frac{\partial ELU}{\partial x} = \begin{cases} 1 & x > 0 \\ \alpha e^x & x \leq 0 \end{cases} \quad (3.1)$$

where $x = \sum_j W_{ij} * s_j + b_i$ (logit of selected neuron i for sample s)

Therefore, if $\alpha < 0$

$$\frac{\partial ELU}{\partial x} = \begin{cases} > 0 & \sum_j W_{ij} * s_j + b_i > 0 \\ < 0 & \sum_j W_{ij} * s_j + b_i \leq 0 \\ \approx 0 & \sum_j W_{ij} * s_j + b_i \ll 0 \end{cases}$$

Please note that the choice of training non target samples to have a logit as far from zero as possible (to the negative side) is not arbitrary. Most samples will be non target, so one must ensure that their influence in the selected neuron’s gradients does not harm attribute inference. Moreover, it is important to note that non-target samples will only have zero influence if their gradient is null, which only happens in practice due to finite precision number representation. Furthermore, it is important to guarantee that gradient’s computation from subsequent layers (which affect the malicious neurons gradient) does not change the sign. Based on Appendix B, the adversary has a condition on how to select the malicious neuron (or how to change the selected neuron’s weight that connects it to the output layer accordingly), which is explained in the same section.

As one can imagine, encoding negative values close to zero is harder than the rest, as a small error can cause incorrect attribute identification. Thus, an adversary can reduce the probability of those cases by encoding the most frequent attribute value to positive logits, whether that is the negative or positive value. In this case, the inference phase must be adjusted accordingly. On the other hand, if the AAI* attribute distribution is extremely imbalanced, the adversary may be more interested in inferring correctly samples that diverge from the expected (most common value). In such cases the adversary may encode the least frequent value to positive logits, therefore reducing the probability of misclassifying a sample from the least common attribute value.

Unlike AMI, this network is trained as in a regression problem, despite the fact that the target variable only has three different values. A kind of weighted loss strategy is also applied to ensure AAI* success and its role is to give greater weight for target samples’ gradients, replacing the need of duplicating target samples. After the adversarial network is optimized, the server replaces benign weights for maliciously modified ones in the global model, which is then sent back to clients. During the inference phase, the adversary decides the target’s attribute value based on *aggregated gradients sign*.

3.1.5 Attribute Inference based on Membership Inference (MIA2AIA)

In order to show that AAI* improves upon the most common type of AIA in terms of attribute information leakage, a MIA-based AIA, particularly based on AMI, is proposed. MIA2AIA is an *active white-box* attack, performed by the central server, which can be considered a metric-based, like AMI and AAI. The intuition behind this attack is to perform MIA for every possible attribute value of each target sample. For a binary sensitive attribute, the adversary applies AMI to two samples resembling the target one, each of which with a distinct attribute value. The target sample’s attribute value is the one for which its neuron was activated, presumably the only one.

Given that MIA baselines were all passive attacks, one could be tempted to use an AIA based on a passive MIA as baseline. However, the target model's behaviour, which is exploited by passive attacks (as they cannot force an increase of model leakage), may not be sufficiently different for samples only differing on a single predictive feature, unless that feature is highly correlated with the main task. Therefore, comparing AAI* to an active AIA baseline enables a fair comparison regardless of the sensitive attribute's correlation with the main task.

Using the taxonomy (Figure 2.3), one would classify MIA2AIA as:

- **Goal:** Attribute
- **Target:**
 - **Phase:** Training
 - **Model:** Federated
- **Adversary:**
 - **Identity:** Central server
 - **Access:** White-box
 - **Effect:** Active
 - **Approach:** Metric based (gradient)

Note that this approach may not be restricted to binary sensitive attributes, although only binary attributes are tested for comparison purposes. As it is a version of AMI for attribute inference purposes, its conditions are the same. In this section, the reader can find an explained attack methodology (3.1.5.1) and empirical results in Section 3.2.

3.1.5.1 Methodology

Methodology of novel active attacks, in which MIA2AIA fits, was briefly outlined in Section 3.1.1, thus specific details are explained in this section. In a FL system, a malicious server executing MIA2AIA selects a couple of neurons, in the case of a binary sensitive attribute, for each target sample and collects its weights after the aggregation of client's updates. Each target sample is duplicated, assuming a binary sensitive attribute, such that each duplicate has a different attribute value. An adversarial network is trained to leak membership information of each of these samples in the same way as in AMI.

The major difference between AMI and MIA2AIA methodology is in the inference phase. More than a distinct purpose, MIA2AIA can have three different outputs: correctly or incorrectly inferred attribute value and *inconclusive attack*. The first two outputs are self-explanatory, and the last one includes the possibility of a target sample, during FL training, activating more than one neuron. When more than one neuron, in the set of neurons dedicated to a single

target sample, is activated, the adversary still does not know which attribute value to predict. Instead it only knows that the target sample is a member, in the best case scenario in which no other sample activates that neuron. In those cases, a realistic adversary must use its statistical knowledge about the training data to infer the attribute value. Therefore, when more than one neuron is activated, the predicted attribute value is the most probable one - the most frequent value in the shadow dataset. Although, it is important to note that MIA2AIA performance is now based on AMI, a powerful attack, but also in *data imputation*, unlike AAI* which is solely based on *attribute leakage*. Additionally, one must note that inconclusive attacks were not an option for AAI* inference phase.

3.2 Evaluation and Analysis

In this chapter, the reader can find a comprehensive analysis of membership and attribute inference attacks, previously explained, applied on a real-world dataset. The effectiveness of active attacks is evaluated and compared to already existing MIAs and AIA in order to prove that well-generalized models are vulnerable to these type of attacks. Several components that may have a significant impact on attack performance are studied, such as number of target samples, target model complexity, target model learning protocol, and adversarial knowledge. First, details about the practical setup are clarified in Section 3.2.1, then passive membership results in Section 3.2.2, active membership results are presented in Section 3.2.3, and attribute inference results in Section 3.2.4. In Section 3.2.5 the target model's ability to recover from active attacks is explored. Section 3.2.6 is dedicated to measure active attacks real threat under a practical scenario. Finally, the connection between membership and attribute inference is commented in Section 3.2.7.

3.2.1 Experimental Setup

In this section, the reader can find details about the experimental setup, particularly about the dataset used in Section 3.2.1.1, target models in Section 3.2.1.2, and metrics in Section 3.2.1.3. All experiments conducted in this study were repeated 30 times to ensure robustness and reliability of findings. A confidence interval of 90% confidence was calculated, providing a rigorous statistical guarantee. Note that the term *significant* is used throughout this result section to denote statistical significance of findings, meaning if the confidence intervals involved are disjoint. To account for a possibly different significance level, when intervals are disjoint but close to each other the expression *significant but for a small margin* (or similar) is used.

Regarding active attacks, some adversarial setup parameters are fixed, except for those whose impact on attack performance is carefully studied. Adversarial knowledge is determined by its shadow dataset, which is set to have 1000 samples randomly selected from the initial dataset. These samples are excluded from every benign experiment. Naturally, the greater the knowledge the easier it is to conduct the attack, as training of malicious neurons' weights is expected to

approach the training dataset behaviour on modified neurons better when the shadow dataset approaches real training data, in size and in distribution. Since active attacks have an impact on main task performance, one must not use all second fully connected layer neurons for a malicious purpose. Therefore, $\frac{2}{3}$ is defined as the maximum proportion of neurons used for that purpose. The standard model architecture is: 53 input neurons, 2 hidden layers with 128 and 64 neurons each and 1 output layer.

3.2.1.1 Dataset

The dataset used for these experiments was collected in a previous project, *COP-MODE project* [1]. The dataset is comprised of permission decisions captured from a runtime permission manager in Android (Mendes et al. [39]). An application was developed to intercept permission requests and collect necessary data. The final dataset, comprising 93 users has 2180302 permissions, from which 65261 were manually answered by the users, with an average of 701.73 answered requests per user. The following list contains some of the features collected whenever a request was intercepted.

- *wasRequestExpected* - User expectation regarding the current permission request;
- *appCategory* - Requesting application category (communication, social, tools, ...);
- *checkedPermission* - Type of requested information (location, microphone, camera, ...);
- *isForeground* - Whether the app is currently in foreground or background;
- *isTopAppRequestingApp* - Whether the requesting app is the top foreground app, meaning it is currently being used by the user;
- *selectedSemanticLoc* - User's semantic location given as input by himself;
- *hour* - Hour of the day the request is displayed;
- *networkStatus* - The network status at the time of the permission dialog;
- *isInEvent* - Whether the user is in an event at the current time according to his calendar;
- *isWeekend* - Whether the request was displayed during the weekend;
- *callState* - The current call state as returned by the TelephonyManager;
- *plugState* - The current plug (charging) state.

This dataset is used to train machine learning models to *predict user's answers* (Mendes et al. [39], Brandão et al. [6]). From those manually answered permissions, 66% were accepted and 33% denied. Among several predictive features, enumerated below, *expectation* (*wasRequestExpected*) is by far the most correlated with the target (57%).

To feed these features into a NN, categorical features are converted to its one-hot-encoding form, and numerical features are normalized. This preprocessing step yields an input size of 53, meaning each sample (user answer) is characterized by 53 features, most of which are 0's (expressing absence of that attribute).

To train a FL model with COPMODE dataset, each client represents a user whose dataset is composed by the permission requests he answered. Hence, clients division is realistic, as each user corresponds to a real user from the data collection campaigns, even though the federated learning setting is a simulation (in a single machine). However, some users have small datasets, some of which with a single label, meaning they accepted or denied all requests. Moreover, most clients' data, as well as the centralized version of the dataset, is imbalanced with respect to the target variable. To overcome such limitations, one must perform local *random oversampling*¹.

Furthermore, the question of how to measure the model's ability to predict users answers maybe address differently according to the business objective, meaning depending on whether there is a target label (grant or deny permission) that one is most interested in predicting correctly. Accuracy could be a suitable evaluation metric since it measures the model's predictive quality without prioritizing the positive nor the negative label. However, it is threshold dependent. Therefore, *ROC AUC* is used to measure the probability of classifying a real positive sample more confidently than a real negative sample. That is, it measures how confident the model is in accepting a previously accepted request than a previously denied one.

3.2.1.2 Target Models

These target models are trained to predict user response to permission requests, based on a set of 65K samples, approximately. ROC AUC is the metric used to measure the model's ability to distinguish both classes without prioritizing any of them. Early experiments showed that model performance has a significant impact on attack performance, at least for a large difference in terms of models performance. Furthermore, one the primary goals of this research is to show that well-generalized models are vulnerable to these active attacks, contrarily to what prior work suggests by mainly attacking overfitted models. Therefore, it is relevant to show performance of target models used throughout this work.

Tables 3.1 and 3.2 present training hyperparameters and 90% confidence intervals for ROC AUC of every model used for membership and attribute inference attacks. All models receive input of the same size (53) and have a single output neuron in which sigmoid output function is used. Model complexity only varies in the number of neurons in each hidden layer and models of standard complexity have 128 and 64 hidden neurons, respectively. These complexities were choosen according to the hyperparameter tuning phase, that took into account the fact that models to predict user response on this dataset do not need to be very complex (Brandão et al.

¹This is a sort of data augmentation technique to balance training data with respect to the target variable and it consists of randomly selecting samples from the minority class (with replacement) and artificially add them to the training dataset.

)[6]). The choice of two hidden layers is motivated by the conditions to perform these attacks, as the adversary only maliciously modifies second Fully Connected (FC) layer neurons. All models are trained with the same loss function (Binary Cross Entropy Loss), optimizer (Adam), and all of them use batch size 32. Neither of them applies regularization techniques, such as L2-regularization (most known as weight decay). Note that they are divided by active attack since each attack requires a different activation function.

Model complexity	Attack	ROC AUC	
		Train	Test
(64, 32)	AMI, MIA2AIA	(83.85, 83.96)	(83.78, 83.98)
	AAI*	(83.32, 83.47)	(83.28, 83.48)
(128, 64)	AMI, MIA2AIA	(82.16, 82.36)	(82.06, 82.35)
	AAI*	(83.47, 83.66)	(83.46, 83.68)
(256, 128)	AMI, MIA2AIA	(83.8, 83.9)	(83.71, 83.9)
	AAI*	(83.62, 83.73)	(83.55, 83.75)

Table 3.1: Model performance (represented by 90% confidence intervals of ROC AUC) of federated models with ReLU and ELU as activation function ($\alpha = -2$), used for AMI plus MIA2AIA and AAI, respectively.

Model complexity	Attack	ROC AUC	
		Train	Test
(64, 32)	AMI, MIA2AIA	(81.54, 81.89)	(81.51, 81.84)
	AAI*	(83.04, 83.28)	(82.99, 83.27)
(128, 64)	AMI, MIA2AIA	(82.85, 82.99)	(82.76, 82.99)
	AAI*	(83.17, 83.38)	(83.11, 83.4)
(256, 128)	AMI, MIA2AIA	(81.33, 81.58)	(81.29, 81.55)
	AAI*	(82.77, 82.98)	(82.75, 82.99)

Table 3.2: Model performance (represented by 90% confidence intervals of ROC AUC) of centralized models with ReLU and ELU as activation function ($\alpha = -2$), used for AMI plus MIA2AIA and AAI, respectively.

From Tables 3.1² and 3.2³, one can see that these models are all well-generalized, meaning there is little to no gap between train and test performance, regardless of the training protocol. Moreover, none of each type of models (defined by the attack column) seem to improve upon the other type significantly, not even an increase of model complexity causes significant changes in model performance.

Since a few of the following experiments used centralized models, it is also relevant to show performance of centralized models, which were trained to achieve the same performance (within

²On FL models, each client trains for 5 local epochs per training round and all 93 clients are used at each round (these models were trained by 5 rounds.).

³Centralized models were trained for at most 5 epochs and with smaller learning rates to approximate their performance to the corresponding FL model.

a confidence interval), for comparison purposes. Despite a slight (but statistically significant) difference between distinct type of models (here defined by the learning protocol), centralized models are also well-generalized, since there is no significant train test gap.

3.2.1.3 Metrics

Adequate attack performance measures are important to better judge the potential threat of privacy attacks and which factors improve its performance. Since this chapter is about privacy attacks, the goal is to improve attack performance. As for active attacks, one must use TPR and FPR as they are not threshold dependent, their theoretical inference phase is based on gradient updates nullity/sign (whose definition does not depend on a threshold). Therefore, ROC AUC is not appropriate. It is important to note that TPR denotes correct neuron activation, meaning each target sample activates its correspondent neuron. On the other hand, FPR denotes non-target samples that activated a neuron. Note that, for multi-target attacks, it is presented an average over selected neurons, and that passive attacks are evaluated by ROC AUC as they are threshold dependent (as previously explained in Section 3.1.2).

Regarding AAI* and MIA2AIA experiments, TPR and FPR are also used to measure attack performance. However, TPR is always compared to a baseline, which corresponds to the percentage of samples that have the majority attribute value. In other words, the baseline stands for the maximum rate, on average, that a dummy classifier would achieve by solely attributing the most frequent value to every target sample. It is important to compare TPR to this baseline because if TPR is lower than this baseline then a dummy classifier would achieve greater performance, assuming that FPR does not affect the ability of an adversary to make accurate inference.

These metrics (TPR and FPR) are used to measure attack performance on the lab, meaning in the domain of neuron activation, following the theoretical inference phase. However, even if TPR is perfect (100%), FPR is almost never perfect (0%). Thus, in practice, the adversary must account for a small margin of error. Finding an acceptable margin of error is equivalent to finding a threshold above which one must classify the target sample as a training sample (analogously a positive sample with respect to the sensitive attribute). Following on the need to measure attack performance by accounting for a small margin of error, a more challenging evaluation and respective metrics are presented in Section 3.2.6.

3.2.2 Passive Membership Inference Analysis

This section presents experimental results for passive MIAs, with the goal of showing that *passive MIAs are considerably less powerful and require overfitting*. Metric-based passive attacks, detailed in 3.1.2.1 and 3.1.2.2, are tested on both federated and centralized models, since these only require black and white-box model access. The model-based passive attack detailed in 3.1.2.3 is only evaluated on federated models, as its goal is to explore continuous leakage in FL systems.

Model-based White-box Improved Gradient-norm-based (WBIG) attack was previously trained on gradient norms of shadow samples collected over 5 training rounds of a federated learning simulation. The hyperparameter tuning phase, using the shadow model and shadow dataset, yielded a simple attack model - a LSTM layer that receives a sequence of 5 inputs and returns a feature representation vector of size 25 followed by a linear regression network (meaning from these 25 features a linear decision boundary to divide members from non-members is drawn). This attack model yields approximately 63% of ROC AUC in the validation phase.

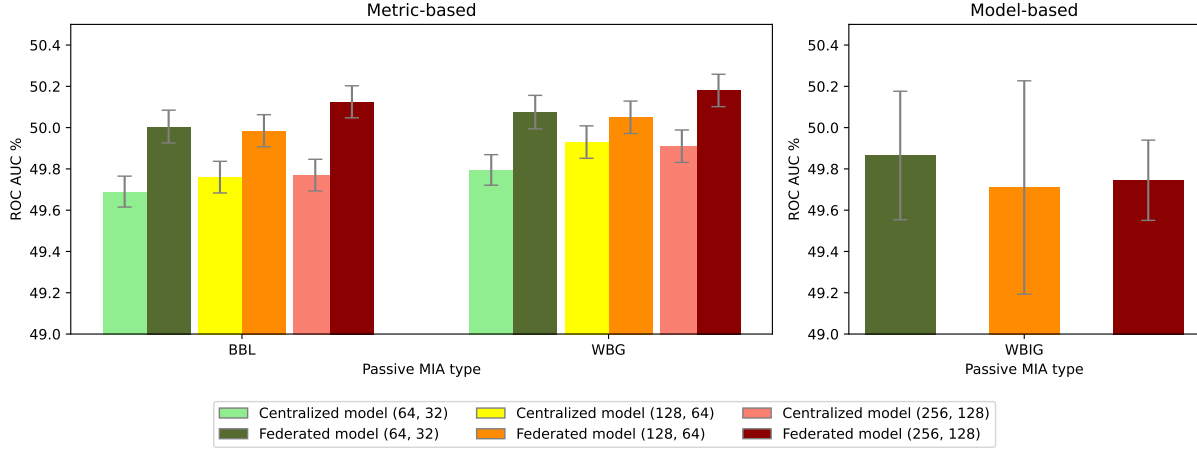


Figure 3.2: ROC AUC of passive metric and model-based MIAs on centralized and federated models of several complexities.

Figure 3.2 illustrates ROC AUC of every passive MIA on models of different complexities, both federated and centralized when applicable. From this figure, one can see that *passive MIAs are not significantly different from random guessing* (i.e. around 50%), regardless of the approach. Model complexity does not impact these attacks performance significantly, but training protocol does in some settings.

The reason why WBIG works significantly better for the shadow dataset (63%) than for the target dataset ($\approx 50\%$) is explained by Figures 3.3 and 3.4. Figure 3.3 illustrates the shadow dataset’s gradient norm distribution, obtained after 1 and 5 training rounds of the shadow model. Analogously, Figure 3.4 shows the target dataset’s gradient distribution computed from the target model. Note that colors denote membership status. From Figure 3.4, one can see that even after the first training round, members and non-members’ gradient norm distributions are identical, that is the target model generalizes quite well for both members and non-members from the start. On the other hand, according to Figure 3.3, there are significant differences between members and non-members gradient norm distribution, which are reduced over training rounds. Regarding WBG performance, the bottom row of Figure 3.4 also explains such performance, as gradient norm of members and non members is identical after 5 training rounds. Note that gradient norm is correlated with loss, therefore one may use the same observation to explain BBL performance.

All in all, these attacks are ineffective, as they are not significantly different from random guessing, due to the fact that the target model generalizes quite well, leaving no room for an

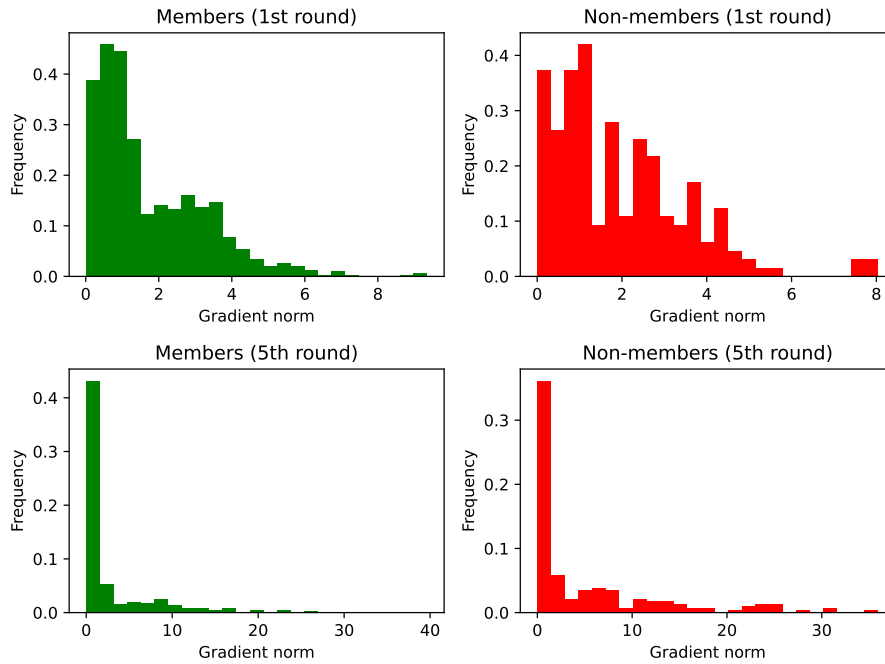


Figure 3.3: Gradient norm distribution of the **shadow dataset** after the first and last training rounds, based on a standard complexity **shadow model**.

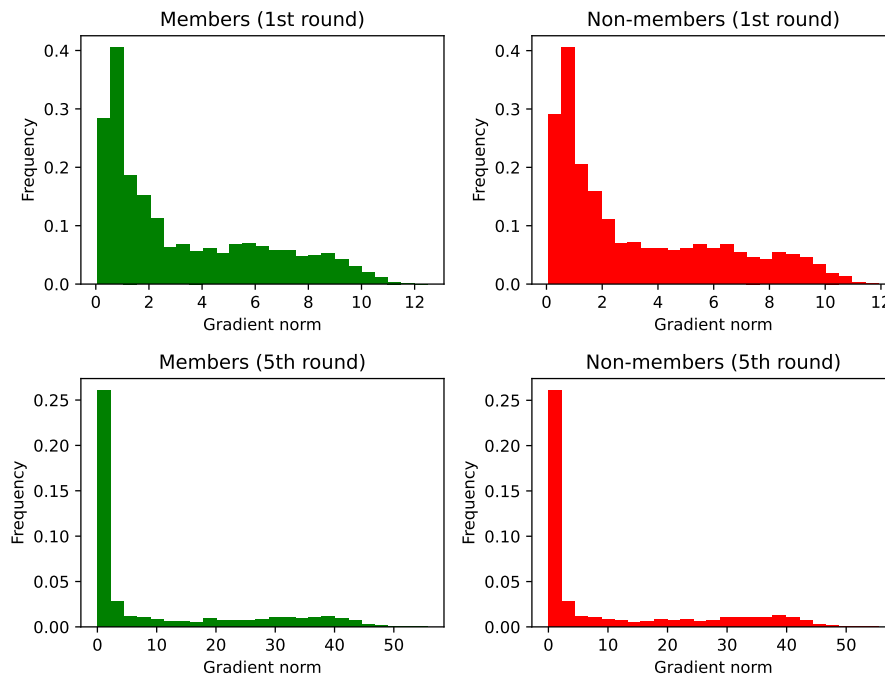


Figure 3.4: Gradient norm distribution of the **target dataset** after the first and last training rounds, based on a standard complexity **target model**.

adversary to exploit its potentially different behaviour on members and non members. These experiments show that passive MIAs rely on overfitting to succeed as these approaches, from the state of the art, are known to achieve better performance on extremely overfitted models.

3.2.3 Active Membership Inference Analysis

In this section, the reader can find experimental results of AMI in which the impact of several components on AMI performance is presented in order to test the robustness of AMI, with the goal of showing that *AMI is a powerful active white-box attack that improves significantly upon passive attacks*. Since all models are non-overfitted, impressive AMI results prove that well-generalized models are vulnerable to these types of attacks. The following experiments, regarding the impact of several attack components, aim to investigate whether the adversary can increase what should already be an impressive attack performance, judging by results presented by Nguyen et al. [42], or whether attack performance is significantly affected by something that is in, or eventually out, of the adversary’s control. That is, the number of target samples, model complexity, and the learning protocol (in some cases) are parameters that the adversary can tune conveniently, whereas adversarial limited knowledge is out of the adversary’s control.

These experiments have some fixed hyperparameters to better judge each components’ impact on attack performance. The learning rate is 0.05 given that it yielded a better attack performance, both on the validation phase (on the central server side with the shadow dataset) and on the testing phase (with the target dataset). In every experiment, the adversary trains the adversarial network for 100 epochs, then evaluates TPR and FPR on every training sample (including oversampled ones). Note that the choice of a high number of epochs is motivated by the need to overfit target samples, that is each malicious neuron should overfit the respective target sample to the point of only being activated by it.

3.2.3.1 Impact of the number of target samples

From Nguyen et al. [42] impressive results, one must be expecting great attack performance when the same attack is conducted on these models. However, this paper only evaluated TPR and FPR for single-target attacks. Thus, it is interesting to see *how does attack performance behave with an increase of target samples*, meaning if the attack model is able to overfit several target samples to the point of leaking membership information.

Figure 3.5 illustrates TPR and FPR evolution with an increase of the percentage of neurons controlled by the attacker, meaning percentage of target samples divided by the maximum, which is exactly the number of neurons in the second fully connected layer. For models of standard complexity, the central server can try to infer membership of at most 64 samples, that is exactly the number of neurons in the second FC layer from which neurons will be targeted to leak membership information. Although restricted to a maximum of $\frac{2}{3}$ of these neurons (to potentially reduce AMI’s impact on main task performance), only 42 samples can be targeted at most. From Figure 3.5, one can see that *TPR is always 100% regardless of the number of target samples*. Additionally, FPR is always below 3% and tends to slightly decrease with an increase of the number of target samples. The fact that *FPR is always under control* suggests that the shadow dataset with which these neurons are trained to overfit target samples is sufficiently

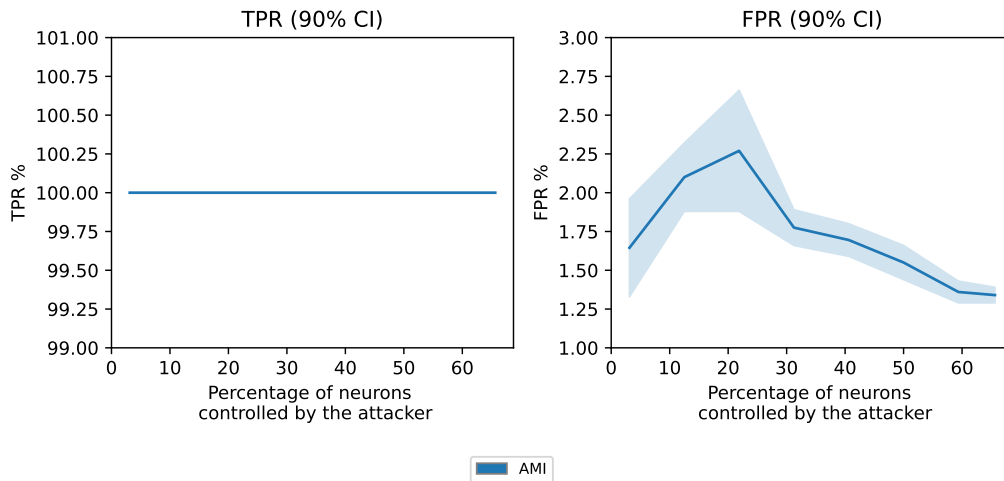


Figure 3.5: Multi-target AMI TPR and FPR on federated models (standard complexity).

representative of client’s data.

These experiments reveal that AMI can effectively overfit several target samples, leaking their membership information, proving that the impressive attack performance from Nguyen et al [42] still holds with an increase of the number of target samples. These results are particularly relevant to show how threatening this attack could be by revealing that an ambitious adversary, with multiple target samples, not only succeeds at his malicious inference, but also does it while reducing the error (FPR). Most of the following adversarial setup parameters whose impact on attack performance is also carefully studied must be evaluated for an increasing number of target samples, since, for this particular setup (federated models of standard complexity), great results were observed.

3.2.3.2 Impact of model complexity

According to Zhao et al. [67], linearly increasing second fully connected layer neurons enables the adversary to increase leakage rate while maintain the main task performance. But an adversary conducting AMI is not allowed, by design, to change the target model architecture along the training process. Nevertheless, as a service provider, the adversary may set, from the beginning, a more or less complex model according to the main task characteristics and its potential intention of attacking this system. Moreover, increasing model complexity may help to overfit the target sample, given that the original weights distribution is described by an increasing number of bins. Therefore, it is relevant to study the impact of model complexity, a parameter determined by the adversary in some cases, on AMI performance.

Figure 3.6 illustrates multi-target AMI TPR and FPR on federated models of increasing complexity. From this figure, one must be convinced that AMI impressive *TPR values are not affected by target model complexity (as all complexities reach 100% on average)*. On the other hand, AMI *FPR significantly decreases as target model complexity increases*. Such evolution is

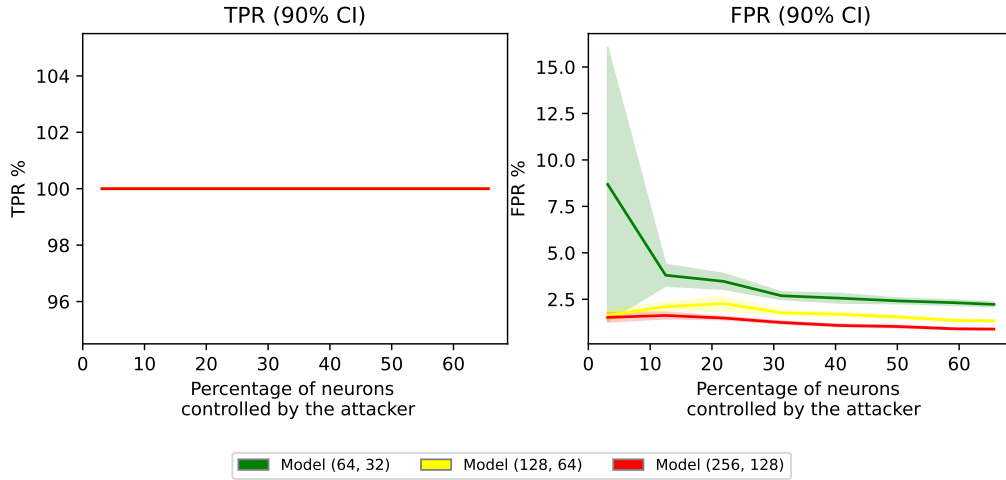


Figure 3.6: Multi-target AMI TPR and FPR on federated models of multiple complexities.

only statistically significant when the adversary controls more than 14% of neurons. Although for simpler models, which incur in a higher amount of error, FPR is out of control (being at most 16%). Note that the impact of the percentage of controlled neurons does not change according to model’s complexity, as all complexities exhibit the same behaviour - a tendency to decrease FPR with an increase of the number of target samples, while maintaining perfect TPR on average (100%).

Other than being potentially capable of passing more information maintaining the same number of neurons dedicated to the malicious task (or possibly more), more complex models give an extra advantage to the adversary in terms of FPR. This insight was already expected given that, with a more complex architecture, the adversary can represent target samples in a higher dimensional space, naturally able to encode more information, which makes distinguishing target samples from non target ones an easier task. These results are partially in accordance with Zhao et al. [67] since increasing model complexity also increases the leakage rate (the same percentage of controlled neurons allows to attack more target samples), however the effect of this adversarial parameter on main task performance after AMI is explored latter in Section 3.2.5. Moreover, as a parameter determined by the adversary in some cases, these experiments reveal that an adversary may increase the target model complexity to improve its inference, and as an attempt to evade the defense (the behaviour of defense strategies under different model complexities is explored in Chapter 4).

3.2.3.3 Impact of target model’s learning protocol

Centralized models are not realistically exposed to active white-box attacks, since their training phase does not depend on more than one device/client, but, from the adversary’s point of view, centralized or federated target models are simply a different initialization for the malicious network. From the experimental phase of this work, centralized models were able to reach

significantly higher performance (when hyperparameters were tuned to optimize the network), which suggests that models trained under this protocol can better, and faster, express the relation between input features and the main task label. Also note that if, for some reason, the FL system has a single training client from the beginning, that client would be essentially training the global model on a centralized fashion. Thus, it is relevant to see whether an initialization coming from a centralized learning model leads to a significant difference on AMI performance. Note that both centralized and federated models have the same performance (Tables 3.1 and 3.2), therefore any significant difference must come from the learning process itself.

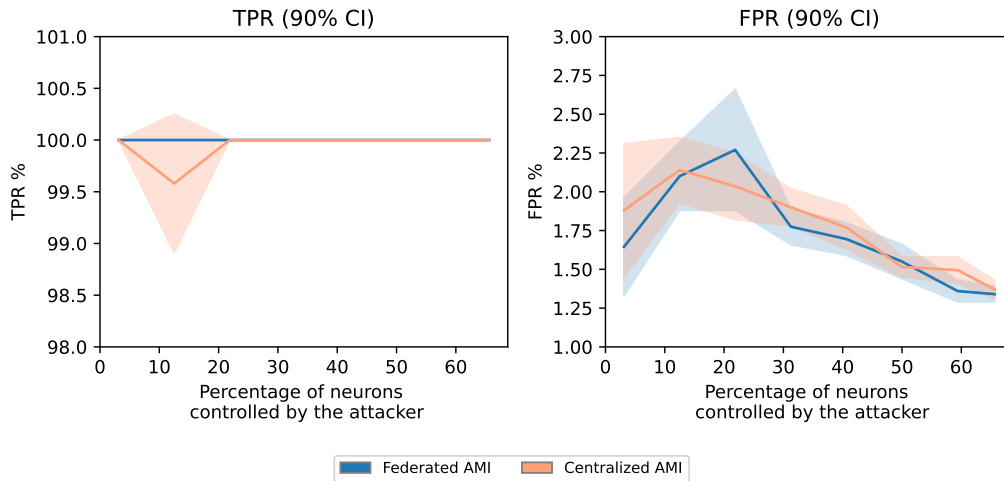


Figure 3.7: Multi-target AMI TPR and FPR on centralized and federated models (standard complexity).

Figure 3.7 shows multi-target AMI TPR and FPR evolution depending on the type of target model. This figure shows that *federated models are neither significantly harder nor easier to attack than centralized models*. Note that this conclusion does not invalidate empirical insights from Pustozero et al. [44], as the adversarial setup, for attacking centralized and federated models, may have caused those observed differences, given that the adversary had a distinct target model access according to the type of model. These experiments suggest that AMI performance does not seem to depend much on the initialization (meaning the target model), at least when the target model has the same performance (in terms of ROC AUC). Therefore, after the target model has converged, AMI could occur in any of the following training rounds.

3.2.3.4 Impact of limited adversarial knowledge

A realistic adversary is one that does not know everything and does not have unlimited access, an adversary worth to defend against, as it poses a realistic threat. As an attempt to design a slightly more realistic adversary, the central server may not know every predictive feature of target samples. Yet that should not stop him from trying to infer membership. Instead, he can estimate those features according to his knowledge about clients' data distribution. Shadow dataset's average or most frequent value, depending on the type of feature, is taken as the missing

value for every target sample. Starting from the least correlated main task features, one must see *if AMI performance is still strong as the percentage of unknown features increases*.

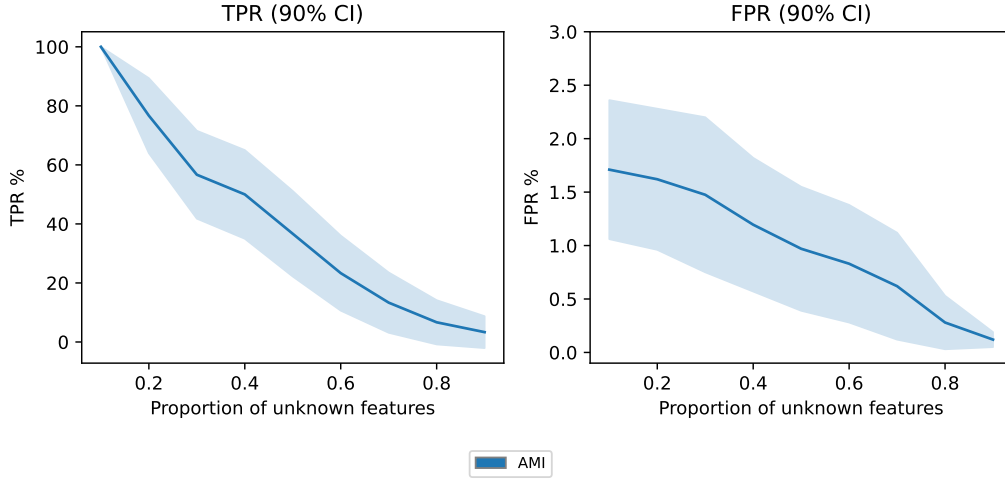


Figure 3.8: Single-target AMI TPR and FPR under adversarial limited knowledge (federated models of standard complexity).

Figure 3.8 illustrates single-target AMI TPR and FPR evolution with an increase of unknown features. It is clear that *this scenario hinders membership inference significantly*, particularly in terms of TPR. As the percentage of unknown features increases, the adversary is progressively targeting the average (shadow) sample. For a large percentage of unknown features, this neuron is trained to be simultaneously activated by the average shadow sample and deactivated by every single shadow sample, which is an incoherent goal. Note that TPR and FPR evolution present similar decreasing trends (yet with a much higher reduction for TPR), which means that neuron’s activation rate, either by the intended target sample or by unintended client’s samples, decreases.

This finding also reveals that input perturbation (potential defense strategy) is expected to impact TPR significantly. Although, judging from this figure, one would have to perturb almost the same percentage of features as the desired TPR decrease. A standard input perturbation technique (e.g. addition of noise) could require even a higher percentage of perturbed features to reach the desired drop, as adding low-variance noise does not perturb each feature as much as replacing it by its global mean (at least in some cases).

3.2.3.5 Summary

These experiments show that AMI effectively encodes membership information on maliciously modified neurons, enabling the adversary to attack well-generalized models. Furthermore, AMI outperforms passive MIAs (Section 3.2.2), from prior work, that exploit overfitting (BBL and WBG), or continuous leakage during the learning phase of FL systems (WBIG). The reason why these baselines were random guessing is that these target models are well-generalized (given the absence of any statistically significant train test gap, based on Section 3.2.1.2). Moreover,

the impact of some attack components was also studied in order to test the robustness of AMI under challenging conditions that the adversary does not control (limited adversarial knowledge). Additionally, the impact of parameters that are controlled by the attacker is studied to see whether the adversary could improve its inference by tuning those parameters conveniently (number of target samples, model complexity, and learning protocol). These experiences revealed that an ambitious adversary, with multiple target samples, succeeds at his malicious inference while reducing the error (FPR), showing that AMI can effectively overfit several target samples. The speed with which the adversary reduces the error increases with an increase of model complexity. Moreover, the learning protocol, or more generally the initialization of the adversarial network, does not affect AMI performance. Finally and as expected, adversaries with considerably limited knowledge cannot accurately infer membership, as the malicious neuron cannot encode that information (its activation rate drops with a degradation of the adversarial knowledge).

3.2.4 Active Attribute Inference Analysis

In this section, the reader can find AAI* and MIA2AIA experimental results in which the impact of several components on attack performance is presented in order to test how robust is AAI*. Results of this section show that *AAI* is a powerful active white-box attack that improves attribute leakage* of the most common type of AIAs in the literature.

Similarly to AMI experiments, these also have some fixed hyperparameters to better judge each components' impact on attack performance. For single-target attacks, learning rate is 0.01, whereas for multi-target attacks 0.05 is used instead, since this combination yielded a greater attack performance on the validation and test phases (using the shadow and the target dataset, respectively). In every experiment, the adversary trains the adversarial network for 1000 epochs, then evaluates TPR and FPR on every training sample (including oversampled ones). Note that the choice of a high number of epochs is motivated by the need to overfit target samples, which is an harder task comparing to simply encoding membership status.

3.2.4.1 Impact of the sensitive attribute

Attribute inference is, usually, highly *dependent on the sensitive attribute* - it can be as easy as the adversary wants, according to Jayaraman et al. [28]. An attribute for which the most frequent value takes over the vast majority of samples is the easiest to infer with any kind of AIA, at least when evaluated through accuracy. In practice, the adversary does not choose the sensitive attribute aiming to improve its inference accuracy, but according to its ultimate goal. However, from an audit point of view, one could choose a sensitive attribute which, coupled with some metric, yields a higher attack performance. Therefore, more than choosing fair attack performance metrics, one must choose the sensitive attribute value with care.

Since the promise of AAI* is to increase attribute leakage, one must compare AAI* to MIA2AIA version without data imputation, meaning a version for which inconclusive outcomes

are possible. To see if AAI* improves on the best state-of-the-art AIA, one must also evaluate the strongest version of MIA2AIA (with data imputation).

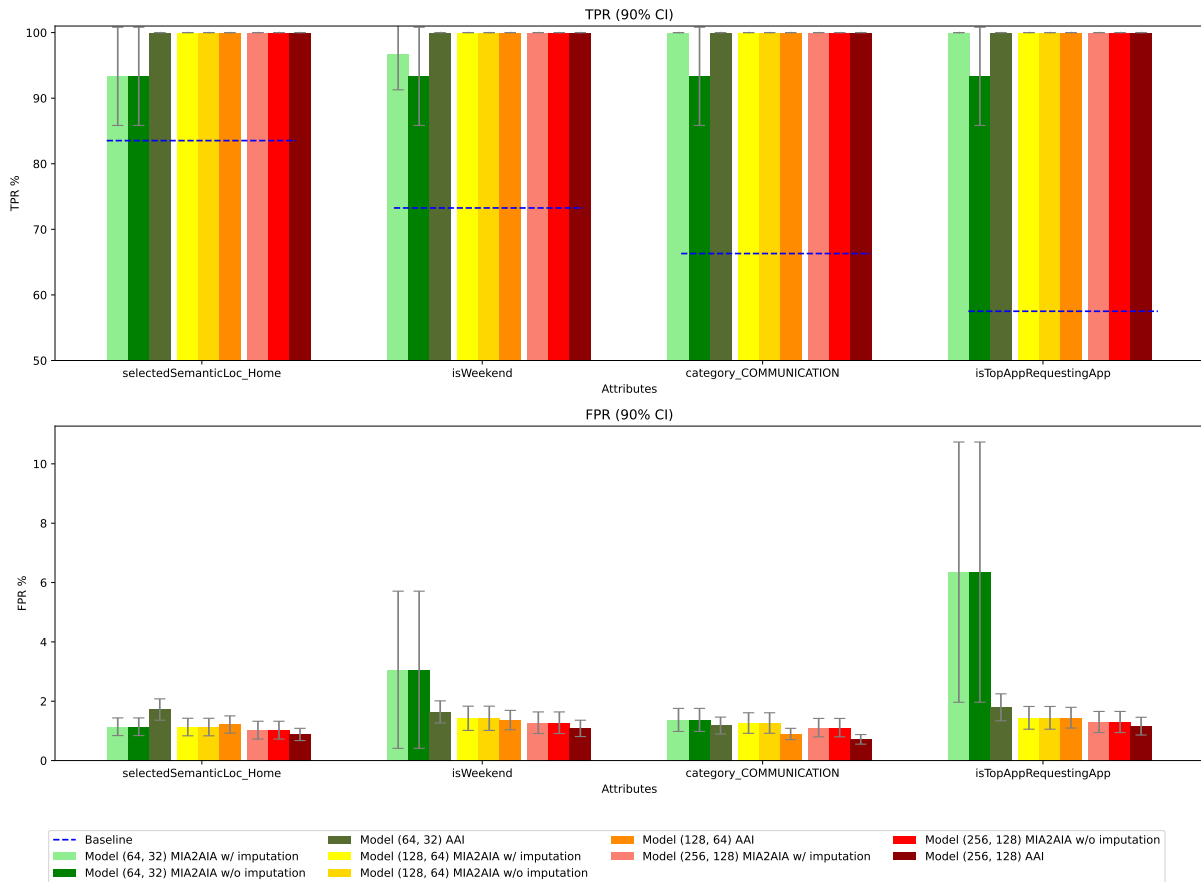


Figure 3.9: Single-target AAI* and MIA2AIA TPR and FPR on federated models (of all complexities) for several sensitive attributes.

Figure 3.9 presents TPR and FPR values for AAI* and both versions of MIA2AIA on several attribute values. In each set of bars, green tones stand for the lowest model complexity, yellow for the standard complexity, and red tones for more complex models. In each set of bars and in each color, different color intensities represent distinct attacks: light tones stand for MIA2AIA with data imputation, intermediate for MIA2AIA without data imputation, and dark tones stand for AAI*. The dashed baseline represents the proportion of samples whose attribute value is the most frequent, meaning a dummy classifier would have that value of TPR. Note that some of these experiments used different formulations of AAI* according to the most frequent value of each attribute, meaning whether it was the positive or the negative label.

Regarding TPR, *AAI** achieves 100% for every sensitive attribute, showcasing its impressive effectiveness. MIA2AIA with data imputation also achieves impressive results, however its distance to AAI*, in terms of TPR, decreases as the proportion of the most frequent attribute value decreases, meaning towards the right. Although this observation only holds for simpler models and any difference is only on average, as it is not statistically significant. However, MIA2AIA without data imputation, over simpler models, yields slightly lower TPR. Differences

between each MIA2AIA version are greater for less imbalanced sensitive attributes, meaning the features towards the right. That could be due to the sensitive attribute’s inherent characteristics (meaning easier to isolate from the rest of the predictive features), but also due to data imputation and its role on MIA2AIA strongest version. As the probability of the less frequent value increases so does the amount of inconclusive attacks when targeting samples with the most frequent value, given the increasing difference between both versions of MIA2AIA. In other words, greater probability of encountering the least frequent value leads to a greater probability of activating its correspondent neuron, which shows the influence of the attribute distribution on FPR (as the incorrect activation of that neuron by the target sample is considered a false positive). This is explained by the fact that AMI incurred in a larger amount of error for simpler models. This insight (influence of the sensitive attribute distribution on FPR) is corroborated by the plot below on the same figure (regarding FPR), since MIA2AIA FPR is considerably higher comparing to other attributes. Although there are no statistically significant differences due to the high variance of these experiments. It is also important to note that MIA2AIA TPR increases with an increase of model complexity, for both versions.

Regarding FPR, there is not a clear trend whether MIA2AIA or AAI* achieves the lowest value. However, the average FPR of AAI* is lower than that of MIA2AIA for the majority of the experiments (except for less complex models when targeting *selectedSemanticLoc_Home*). Even though none of these differences are statistically significant. Additionally, increasing model complexity decreases FPR of every attack, but none of these differences are statistically significant. Moreover, the choice of the sensitive attribute does not have a significant impact on FPR for most model complexities, only for simpler models as explained previously. Please note that when evaluating FPR, versions of MIA2AIA are not relevant, since only attack’s predictive power, meaning TPR, is affected by the modification.

These experiments reveal that both AAI* and MIA2AIA are able to leak precise attribute information for any sensitive attribute. Particularly, AAI* seems to increase model leakage regardless of the proportion of the most frequent attribute value, which is not the case of MIA2AIA on simpler models. Throughout the rest of this section, the selected sensitive attribute is *selectedSemanticLoc_Home*. This choice was motivated by the fact that this attribute is the most imbalanced among this set, meaning that the best version of MIA2AIA would be a better option from the start. Therefore, one must prove that, even on this setting, AAI* is able to increase attribute leakage. Although it could be interesting to compare AAI* performance to state-of-the-art AIAs targeting attributes with less imbalanced distributions, which is left as future work.

3.2.4.2 Impact of the number of target samples

Similarly to the results for AMI, this section evaluates how does AAI* and MIA2AIA performance behaves with an increase of target samples, meaning if the attack model is able to overfit several target samples to the point of leaking attribute information.

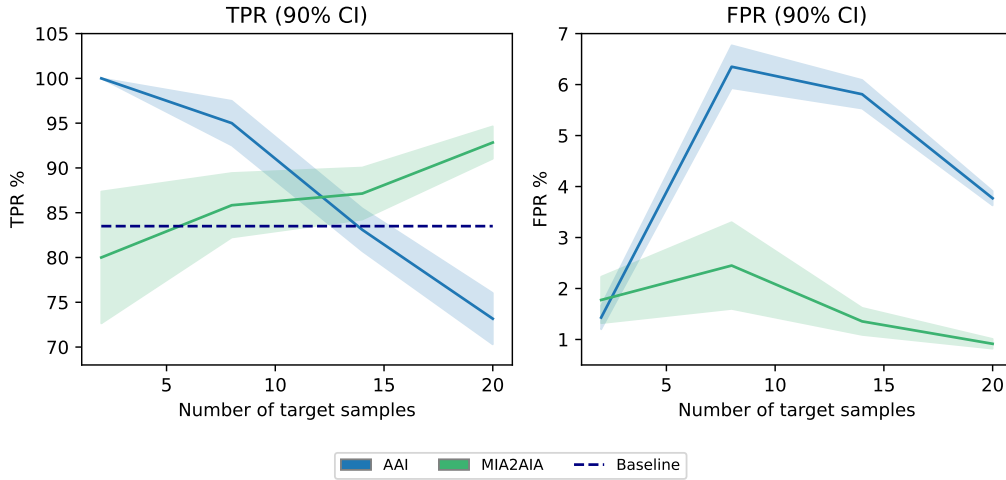


Figure 3.10: Multi-target AAI* and MIA2AIA TPR and FPR on federated models (standard complexity). Sensitive attribute is *selectedSemanticLoc_Home*.

Figure 3.10 illustrates TPR and FPR evolution with an increase of the number of target samples. Note that the percentage of neurons controlled by the attacker is not an adequate measure as MIA2AIA adversary controls 2 times the number of neurons an AAI* adversary does for the same number of target samples. Any attack with TPR below the dashed baseline is not effective. From this figure, one can see that *AAI* TPR decreases as the number of target samples increases*, whereas MIA2AIA TPR increases. For a small number of target samples, AAI* provides significantly better results than MIA2AIA, which does not significantly improve upon the baseline. An AAI* adversary can infer correct attribute value of almost every target sample when that set does not exceed 10 samples but it is ineffective when applied to more than 14 target samples since TPR is lower than the baseline. On the other hand, MIA2AIA is not effective when targeting less than 11 samples, reaching close to 90% of TPR for more than 15 target samples. This behaviour of MIA2AIA is due to the fact that AMI FPR decreases as the number of target samples increases, therefore decreasing the probability of having an incorrect or inconclusive attack. Whereas AAI* behaviour might be explained by the fact that the network is not able to overfit an increasing number of target samples, when it is required that the neuron not only reveals membership but also attribute information.

Taking a closer look at FPR, one must see that both AAI* and MIA2AIA FPR behave similarly for an increase of the number of target samples. However, *MIA2AIA FPR is significantly smaller than AAI**, as it is based on AMI which traditionally had very small FPR. Only for a small number of target samples (less than 3), both AAI* and MIA2AIA have closer FPR. It is also interesting to note that AAI* FPR behaviour is not monotonous across this range: an initial increase of the number of target samples causes FPR to increase (while maintaining TPR close to 100%), however when AAI* becomes ineffective, in terms of TPR as it is lower than the baseline, FPR starts to decrease.

These experiments reveal that both AAI* and MIA2AIA can effectively overfit more than a

single target sample, leaking their attribute information. Contrarily to AMI, AAI* does not seem to have the capacity to reveal the correct attribute information for more than 14 samples, close to the range for which MIA2AIA TPR starts improving with respect to the baseline. Other than showing that these attacks are able to leak sensitive information for more than a single sample, these results start to draw a criteria that an adversary should use to choose which of these attacks to perform. Similarly to the previous analysis of AMI, most of the following adversarial setup parameters, whose impact on attack performance is also carefully studied, must be evaluated for an increasing number of target samples, to see if this diverging trend still holds.

3.2.4.3 Impact of model complexity

As previously seen for single-target attribute inference attacks and for multi-target AMI, model complexity is expected to impact multi-target AAI* and MIA2AIA significantly. In this section, the effect of model complexity on multi-target MIA2AIA and AAI* performance is presented.

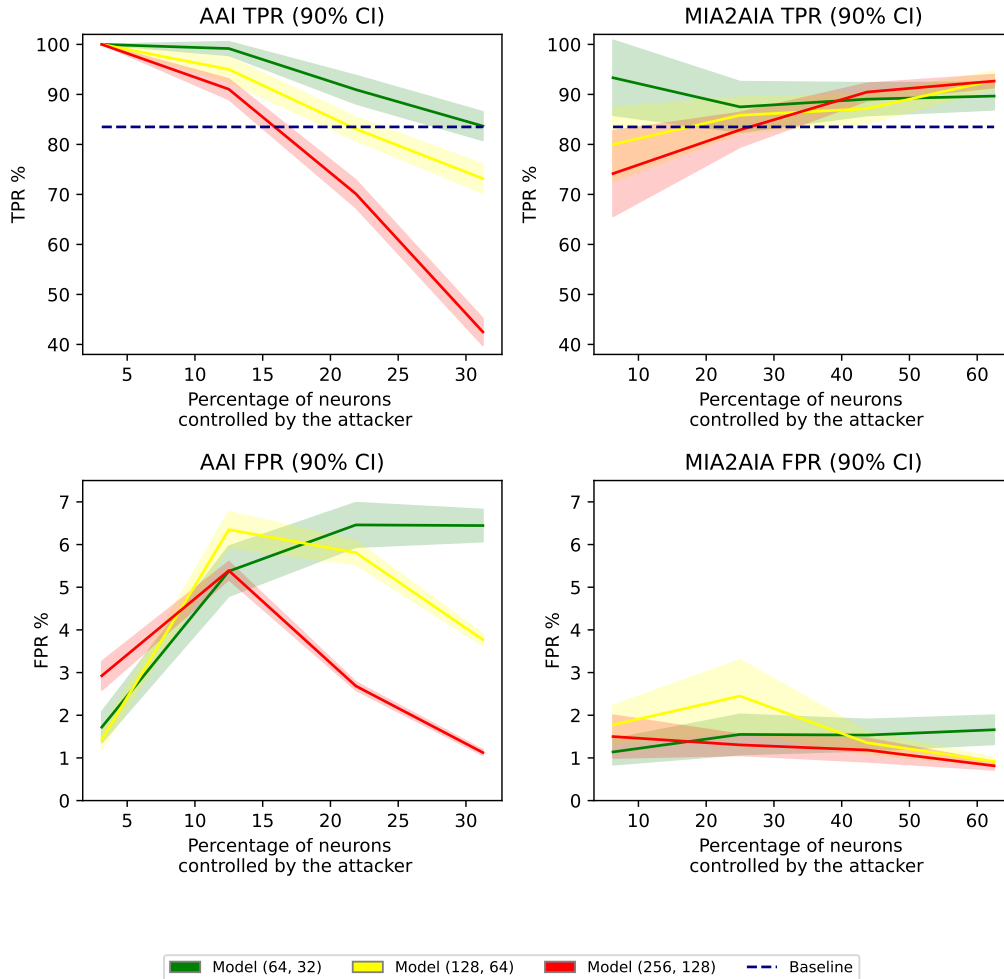


Figure 3.11: Multi-target AAI* and MIA2AIA performance on federated models of multiple complexities. Sensitive attribute is *seletedSemanticLoc_Home*.

Figure 3.11 shows TPR and FPR for multi-target AAI* and MIA2AIA on federated models of increasing complexity. Note that this figure is expressed in terms of the percentage of neurons controlled by the attacker, as an increase of model complexity calls for an increase of the number of target samples (as previously tested on 3.2.3). However, since MIA2AIA dedicates the double of neurons to the malicious task when compared to AAI* dedicated neurons, one shall represent these experiments in separated plots. This figure shows that *target model's complexity impacts AAI* TPR significantly, but does not cause any significant change on MIA2AIA TPR*, namely for greater percentages of controlled neurons. In particular, AAI* TPR decreases as model complexity increases, which is accentuated by an increase of the percentage of controlled neurons. Regarding MIA2AIA, the evolution of TPR is inverted (only on average), increasing with an increase of model complexity but these non significant differences disappear with an increase of the percentage of controlled neurons. Note that for a small percentage of controlled neurons over simpler models, MIA2AIA TPR is actually higher than the rest, as this experiment is a single-target attack. Regarding FPR, both *AAI and MIA2AIA achieve lower FPR with more complex target models*, yet that observation is only statistically significant for AAI*. Please note that along with an increase of the percentage of controlled neurons comes a decrease of both metrics regarding AAI*, particularly for standard and more complex models, which leads to ineffective attacks.

Part of these results were already expected, given the behaviour of AMI performance with varying model complexity, particularly about MIA2AIA TPR. In this case, increasing model complexity also increases the leakage rate. However, AAI* behaviour with an increase of model complexity was not expected, as one would imagine that the representation of inputs in a higher dimension would help the malicious task. Even more unexpected after seeing that single-target AAI* performance for varying model complexities followed the expected behaviour - improvement of attack performance, in terms of TPR and FPR, with an increase of model performance. Considering the incapacity to overfit an increasing number of target samples, this evolution of AAI* TPR may be due to an increase of the number of target samples (to keep the same percentage of controlled neurons) rather than due to an increase of model complexity.

3.2.4.4 Impact of target model's learning protocol

Similarly to AMI's results section, it is interesting to see if AAI* and MIA2AIA performance significantly depends on the adversarial network initialization, particularly for initializations trained under different learning schemes. Once again, any differences must be attributed to the learning scheme itself, not due model performance, as they have the same performance (Tables 3.1 and 3.2).

Figure 3.12 shows TPR and FPR for multi-target AAI* and MIA2AIA with varying number of target samples depending on the type of target model. This figure shows that *federated models are neither significantly harder nor easier to attack than centralized models*, regardless of the attack. Part of this conclusion was already expected for MIA2AIA, since it is an adaptation of

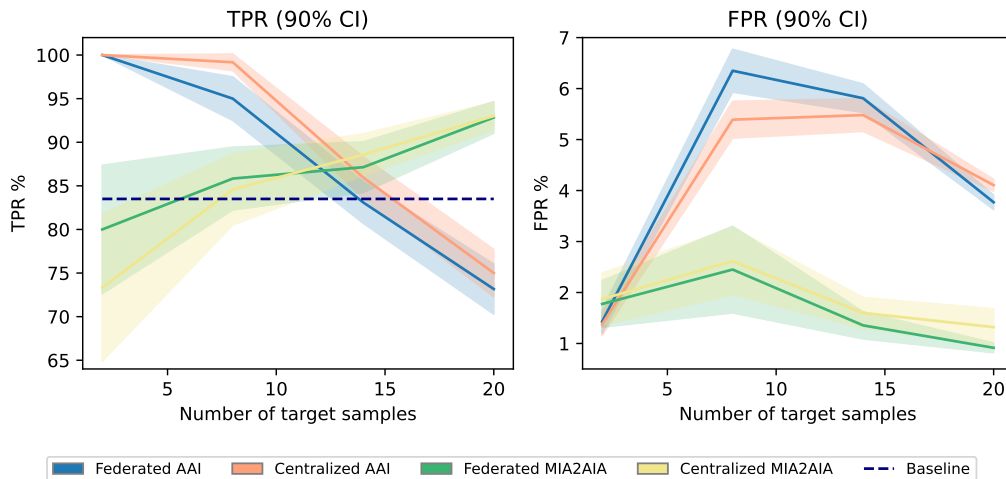


Figure 3.12: Multi-target AAI* and MIA2AIA TPR and FPR on centralized and federated models (standard complexity). Sensitive attribute is *seletedSemanticLoc_Home*.

AMI for coupled (depending on the cardinality of the sensitive attribute) target samples. As previously discussed for AMI over different learning protocols, these experiments suggest that AAI* and MIA2AIA performance does not seem to depend much on the initialization (target model), at least when the target model has the same performance, in terms of ROC AUC. Thus, after a few training rounds, when the target model has converged, these attacks could occur in any of the following training rounds ⁴.

3.2.4.5 Impact of limited adversarial knowledge

Similarly to AMI’s results section, AAI* and MIA2AIA can also be conducted under an adversarial limited knowledge. That is when the attacker does not know every predictive feature of its target samples. Note that the adversary also does not know the sensitive attribute, yet he chooses to estimate, possibly poorly, other unknown features to, hopefully, be able outperform data imputation capacity to predict the sensitive attribute. Thus, starting from the least correlated main task features, one must see if the performance of AAI* and MIA2AIA is still impressive as the percentage of unknown features increases.

Figure 3.13 shows the TPR and FPR for single-target AAI* and MIA2AIA with an increase of unknown features. Once again, it is clear that *this scenario hinders attribute inference significantly*, particularly in terms of TPR. Even for a relatively small percentage of unknown features, both attacks are ineffective. Similarly to the behaviour of AMI under adversarial limited knowledge, both TPR and FPR decrease, meaning the selected neuron’s activation rate decreases, as it is trained for a progressively incoherent goal, as previously explained.

This finding suggests that input perturbation is expected to impact TPR significantly, therefore

⁴Note that AAI* should only be applied after a few training rounds if the adversary does not have the true main task label.

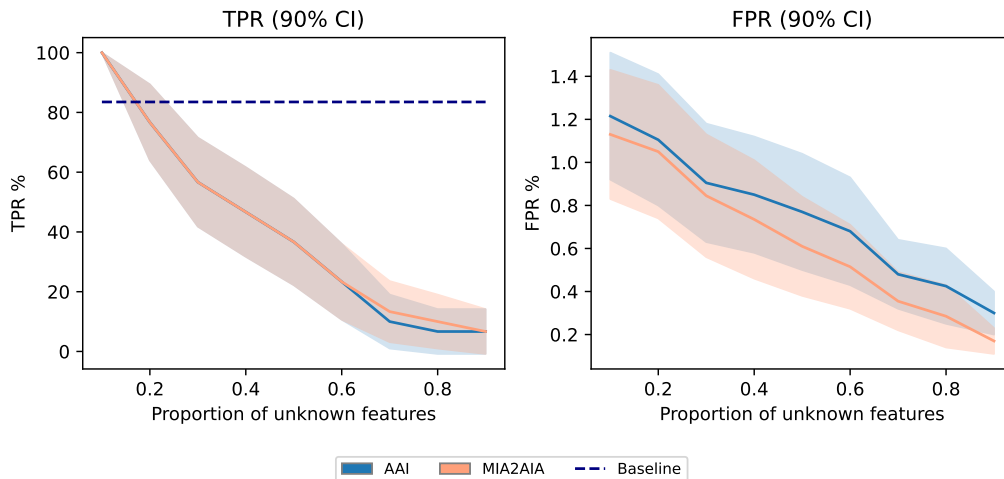


Figure 3.13: Single-target AAI^{*} and MIA2AIA performance under adversarial limited knowledge (standard complexity). Sensitive attribute is *seletedSemanticLoc_Home*.

being a relevant defense strategy to develop, as it would mitigate both active membership and attribute inference attacks (given that the same conclusions were drawn for AMI). Note that these results are aligned with what was previously seen on state-of-the-art papers - *attribute inference typically requires strong adversarial knowledge*, both in terms of the target sample and of the sensitive attribute distribution.

3.2.4.6 Summary

These experiments show that AAI^{*} and MIA2AIA effectively encode the sensitive attribute value on maliciously modified neurons, enabling the adversary to attack well-generalized models. Similarly to AMI, the impact of several components on attack performance was studied to evaluate the robustness of both of these attacks, while possibly drawing the ideal scenario for each of these, meaning combinations of components that favour the choice of AAI^{*} or MIA2AIA. From these experiments, single-target AAI^{*} improves on single-target MIA2AIA, particularly considering MIA2AIA without data imputation (Figure 3.9), although some of those improvements are not statistically significant. From Figure 3.10, one can see that *AAI^{*} TPR improves significantly in comparison with MIA2AIA when the adversary targets few samples*. Such trend is preserved for different target model complexities (Figure 3.11), however AAI^{*} does not benefit from an increase of model complexity coupled with an increase of the number of target samples. Less complex models also benefit MIA2AIA, but only when the adversary targets few samples. Regarding FPR, MIA2AIA improves on AAI^{*} significantly, particularly with an increase of target samples. Similarly to AMI, the learning protocol does not affect attack performance in both cases and an adversary with considerably limited knowledge is unable to accurately infer the sensitive attribute value regardless of the type of AIA. Even though, AAI^{*} outperforms both MIA2AIA versions for a smaller number of target samples, an ambitious adversary that aims to target a greater number of samples may prefer MIA2AIA, at least if the sensitive attribute distribution is

as imbalanced as this case. Although, if the only reason for MIA2AIA to improve upon AAI* is this distribution, then the target model is not the main cause of leakage, but that is yet to be empirically validated with a more extensive analysis, which is left as future work as it would require evaluating both AIAs on less imbalanced attributes.

3.2.5 Model Recovery from Active Attacks Analysis

In this section, the reader can find an analysis of the target model’s ability to recover from active attacks, namely AMI and AAI*. Model recovery from MIA2AIA is out of this analysis, since it is based on AMI, and therefore the same findings apply equally. By the end of this section, the reader should be convinced that active attacks impact model performance significantly, but part of that loss is recoverable after a few training rounds. This shows that *active attacks may occur in FL systems without major long-term model utility loss*.

3.2.5.1 Recovery from AMI

To evaluate the target model’s ability to recover from multi-target AMI after a few training rounds, the same system keeps training after the malicious attack, which occurs after the first 5 training rounds in this case ⁵. Note that the ability to recover is measured with ROC AUC, as the initial target model performance.

Figure 3.14 illustrates ROC AUC evolution over 5 training rounds, after the attack, for models of distinct complexities, for several percentages of controlled neurons, on the train and test set. Recall that ROC AUC is used to measure the probability of classifying a real positive sample more confidently than a real negative one. For comparison purposes, the y-axis is evenly set, per row. The range of colors indicates the percentage of controlled neurons, and the dashed baseline stands for the initial target model average ROC AUC right before the attack. From this figure, one can see that *it is possible to fully recover from AMI on the training set, in at most 5 training rounds, but not on the test set*. As expected, ROC AUC of a maliciously modified target model converges to the initial target model performance as the number of training rounds increases (revealed by the increasing trend of every curve). Moreover, as the target model’s complexity increases (maximum complexity tested is (256, 128)) so does the speed of convergence, as the same performance is achieved sooner on more complex models. Also as product of increasing model complexity, the maximum ROC AUC reached after each round increases, even though these are small increases and namely on the test set. Increasing the percentage of controlled neurons from a point onward does not affect ROC AUC significantly, even though generically targeting more samples benefits target model performance on average (warmer colors tend to be above in almost every curve). Note that in some cases, the ROC AUC on the training set

⁵AMI occurs after 5 training rounds, similarly to the results presented before, because the model as converged, as shown in 3.2.1.2. Attacks over models that have not converged yet, or even over a random initialization, are also effective, although may have a different performance, as previously tested during the experimental phase of this work, and as published by Nguyen et al. [42].

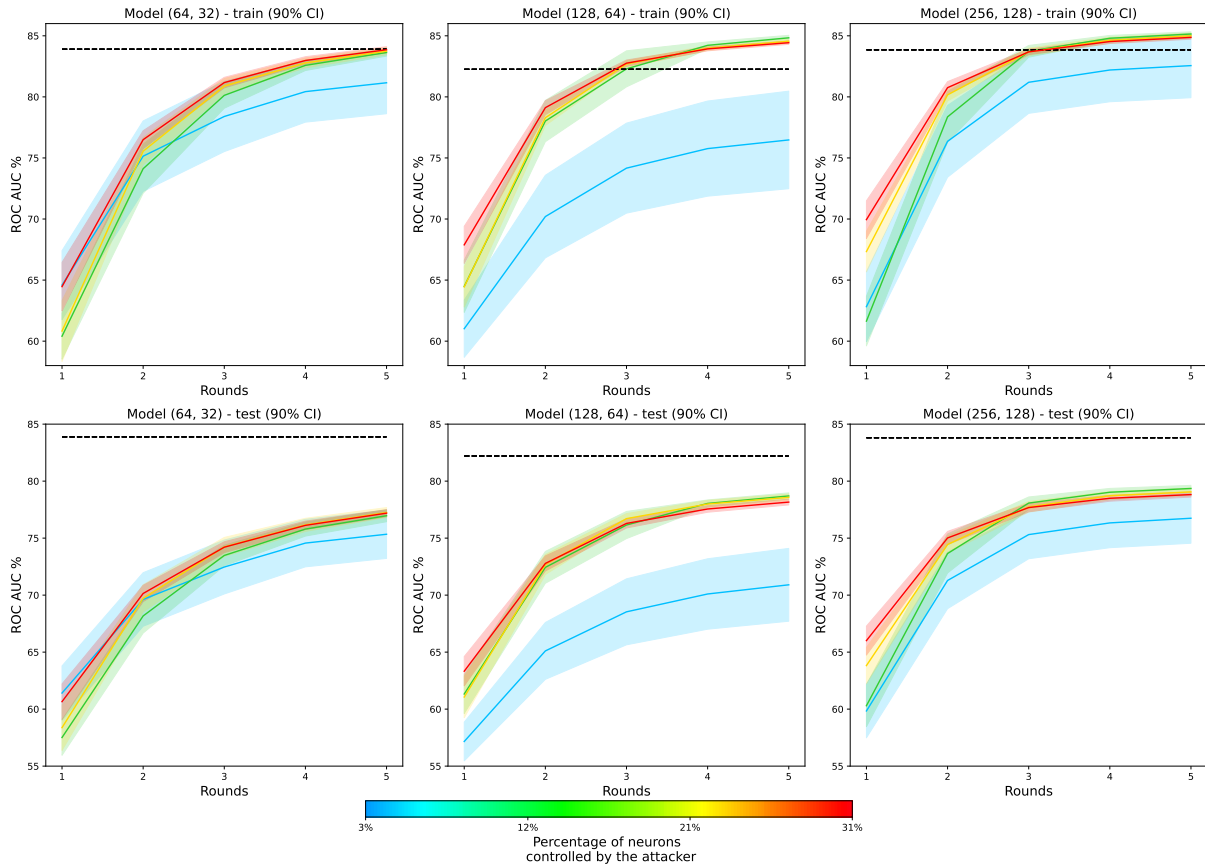


Figure 3.14: Recovery (ROC AUC evolution) from multi-target AMI on train and test set of models with distinct complexities over 5 training rounds.

exceeds the initial target model performance after a few training rounds, for a small and mostly not significant margin.

Additionally, for models of standard complexity (128, 64), when the adversary controls 3% of neurons, the ability to recover from AMI is significantly lower. The same percentage in other complexities also presents an unusually high variance. This might be explained by a combination of the input size (53), the binary nature of most predictive features, and first fully connected layer size. In fact, for small percentages of controlled neurons, that is for a few target samples, given the input’s binary nature, it may be possible that target samples become linearly separable from the rest of the dataset in the first FC layer, making it harder to recover from AMI as only a few samples can update those weights. Increasing the number of target samples in every model complexity decreases variance and increases average, converging to the same behaviour, as can be observed in Figure C.1.

The impact of target model complexity on the ability to recover from AMI is somewhat supported by Zhao et al. [67], which stated that increasing linearly the number of neurons (from the second FC layer) enables the adversary to target more samples without an increase of the main task performance drop. Albeit this is for a different active attack (analytic model inversion attack). When the central server controls 12% of neurons on simpler models and 3% on more

complex models (both target 4 samples), more complex models achieve higher ROC AUC on every round, although the increase is not statistically significant. Comparing to single-target attacks over more simpler models (with 3% of neurons controlled by the adversary), controlling the same percentage of neurons on more complex models, not only enables the adversary to target more samples, but also increases (on average) ROC AUC after a few training rounds.

3.2.5.2 Recovery from AAI*

Similarly to the previous subsection, to evaluate the target model’s ability to recover from multi-target AAI* after a few training rounds, the same system keeps training after the malicious attack for another 5 rounds.

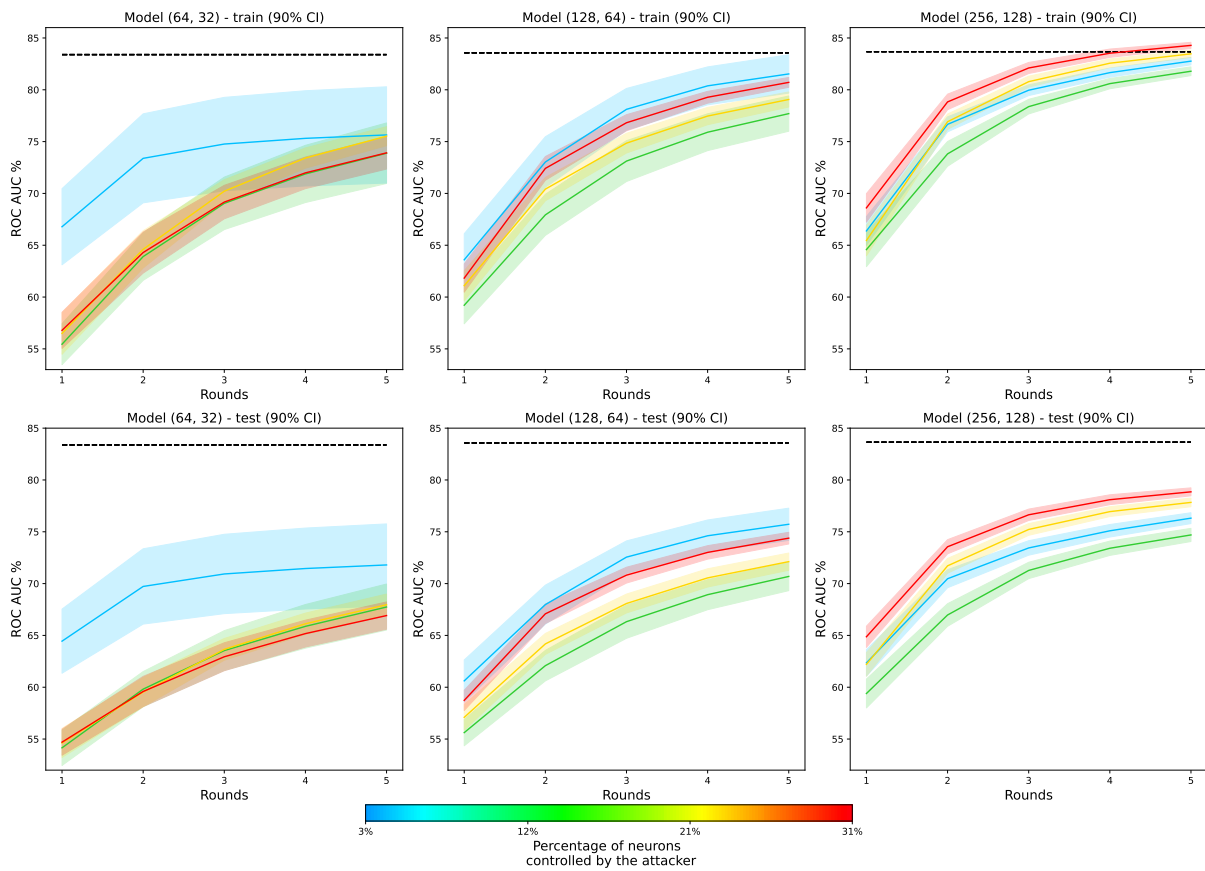


Figure 3.15: Recovery (ROC AUC evolution) from multi-target AAI* on train and test set of models with distinct complexities over 5 training rounds

Figure 3.15 illustrates ROC AUC evolution over 5 training rounds for models of distinct complexities, for several percentages of controlled neurons, on the train and test set. Once again, the range of colors indicates the percentage of controlled neurons, and the dashed baseline stands for the initial target model ROC AUC average (right before the attack). From this figure, one can see that *it is possible to fully recover from AAI* after 5 training rounds, in some cases (more complex models), on the train set but not on the test set*. As expected, ROC AUC converges to,

or approaches in most cases, the initial performance with an increase of training rounds (revealed by the increasing trend of every curve). Even though, it does not approach as much as it did when recovering from AMI. Moreover, the speed of convergence increases as model complexity increases, even more than it did when recovering from AMI, as curves become steeper from the left to the right side of the figure. However, the percentage of controlled neurons does not affect the recover ability monotonously - simpler and standard models benefit, on average, from fewer maliciously modified neurons, contrary to AMI recovery. Whereas more complex models benefit, on average, from a higher percentage of controlled neurons. Yet observable differences are not statistically significant (except on simpler models with only 3% of malicious neurons, that is single-target attacks). Similarly to the previous analysis, some cases exhibit exceptionally high variance, which should also be explained by the same reason.

3.2.5.3 Summary

This section aimed to analyse the target model's ability to recover from active attacks, ultimately showing that these attacks may occur on FL systems without major long-term model utility loss. These experiments showed that target models are able to fully recover from AMI on the training set in less than 5 training rounds. Comparing to AMI, AAI* is significantly harder to recover from, as only more complex models can fully recover from AAI* on the training set. This might be due to the impact of AAI* itself (in terms of ROC AUC right after the attack, or even in terms of the weights distribution, respectively Figures 4.13 and 4.3) or due to the type of model (recall that these models use different activation functions).

These results show that active attacks impact model performance significantly, but part of that loss is recoverable after a few training rounds, at least in some cases. Despite being partially recoverable, the fact that target models are unable to recover on the test set suggests that these models might become more vulnerable to passive MIAs, or AIAs based on passive MIAs, namely those that exploit overfitting. Moreover, the role of certain system-related parameters, such as number of clients and local training epochs, in model recovery is yet to be studied. Nevertheless, the fact that main task performance is partially recoverable motivates the need for defense mechanisms that preserve model utility.

3.2.6 Making inference attacks more robust under gradient uncertainty

In a practical scenario, an adversary aiming to do some inference must be willing to commit some error. An adversary conducting AMI must not expect the malicious neuron's gradient to be exactly zero every time the target sample is a non member. Instead, it must allow for some variability, since not only the target sample activates the malicious neuron, but also its neighbourhood. As in many use-cases, COPMODE's federated models are trained by applying an oversampling technique, or more generally data augmentation, based on training samples. As such, one must expect the neighbourhood of training samples to be better represented on the

final training dataset. Therefore, the malicious neuron’s gradient can be taken as the adversary’s confidence on the target sample’s membership status, as the malicious neuron will most likely be activated by the target sample and its neighbourhood (which one hopes not to be so well represented when the target sample is a non member). An analogous thought can be developed for AAI* inference phase, in which a realistic adversary must allow for a small variability when classifying a target sample as a positive or negative member, as well as a non member. The same idea can be extended for MIA2AIA, in which an adversary must expect theoretically inconclusive attacks (both neurons activated). In such cases, it must infer the attribute value based on the highest gradient.

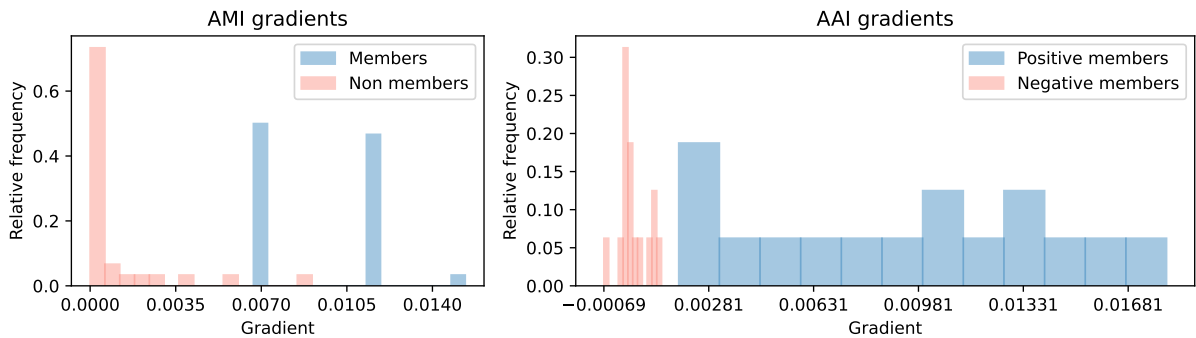


Figure 3.16: Gradient uncertainty illustration.

To show how gradients leak membership and attribute information, Figure 3.16 illustrates the gradient distribution of both members and non members of the malicious neuron selected for AMI (left plot) and positive and negative members of the malicious neuron selected for AAI* (right plot), for a few models. From this figure, one can see that the gradient of members tends to be larger than non members, similarly the gradient of positive members tends to be larger than negative members. Despite having no overlap in this case, it is expected a small overlap, but note that gradients of negative members are, according to the theoretical inference phase, always negative. In this practical scenario and as can be seen in this figure, the adversary must relax that assumption. Intuitively, an increase of FPR, in terms of neuron activation, will lead to an increase of the absolute value of gradients which is expected to reduce the difference between distributions, as they are already affected by false positives, otherwise the gradient of negative members would be always negative. Therefore, the previous theoretical evaluation aimed to maximize TPR while minimizing FPR, which is expected to lead to greater ROC AUC (equivalent to reduced overlap) on a practical scenario. It is important to note that the expectation that gradients of members are higher than that of non members is only with respect to the malicious neuron, as they were trained to be only activated by the target sample which only does if it passes through the model during the training process. This observation does not contradict the intuition behind passive MIAs, which expected the loss (or gradient) of members to be smaller, given that these attacks infer membership status over the gradient of the whole network (or just part of it), but most importantly they do not perturb the optimization process.

As a threshold dependent inference phase (on a practical scenario), one should measure ROC

AUC. However, since ROC AUC is most commonly taken on the full range of FPR, parts of this range do not correspond to an acceptable level of error, even if TPR is higher, as argued by Carlini et al. [7]. Ideally, one should evaluate ROC AUC restricted to a small range of maximum FPR allowed, removing from the equation unacceptable FPR values. Therefore, in this section ROC AUC restricted to a maximum FPR of 25% will be used to measure attack performance. Furthermore, as argued by Yeom et al. [62] and explained in Section 2.4.2, to measure the attribute leakage of a ML model, attack performance on members and non-members must be measured separately. Therefore, to measure AAI* and MIA2AIA performance on a practical scenario, the restricted ROC AUC of their performance on members and non-members is presented separately. Additionally, to measure the real threat of AMI assuming an optimal threshold choice, one shall also see TPR, that is the probability of a training sample being identified as such, analogously the probability of an accurate attribute inference to illustrate the real threat of AIAs. For these experiments, federated models of the highest complexity (256, 128) are used for single-target attacks on a balanced, with respect to membership, set of target samples. Similarly, AIAs experiments were conducted on a balanced set of samples with respect to both membership and attribute value. A confidence interval (90%) is taken over the average restricted ROC AUC per model. For simplicity, gradients are taken over the first pass through the model of the whole training dataset, including oversampled samples, as in previous experiments.

Given this setup, an adversary conducting AMI on a practical scenario would reach a restricted ROC AUC in between [48.91%, 52.18%] with 90% of confidence. Thus, AMI is not effective, under this setup. This might be due to the data augmentation technique and not from a failure on attack performance, in terms of TPR and FPR. In this case, random oversampling is used, but a data augmentation technique that focuses on generating samples from the neighbourhood of randomly selected training samples is expected to increase restricted ROC AUC, since in those cases local density of training samples is higher than non training samples. Although it is important to note that these results do not mean that AMI is not effective, but rather that this level of FPR (in terms of neuron activation) is too high for an adversary to succeed on a practical scenario (in this case). Neither does it mean that not using data augmentation techniques or insuring that the neighbourhood of training and non-training samples is equally well represented on the training set prevents a model to be vulnerable to AMI, as it only forces the attacker to improve its attack performance (reducing theoretical FPR), for example, by increasing the number of target samples or model complexity.

To overcome this limitation, one can evaluate AMI on a non random set of samples, particularly choosing target samples that are members among those that have a higher local density and non members among those that have a lower local density. This setup aims to mimic what would happen with a data augmentation technique that focused on sampling from the training sample's neighbourhood, leaving non training sample's neighbourhood under represented. Note that local density was measured on the whole training dataset, including oversampled samples, using Local

Reachability Density (**LRD**), which is commonly used for outlier detection⁶. Target samples selection could change depending on this parameter. Local reachability density of a point is the inverse of the average reachability distance of its k nearest neighbours. The reachability distance is defined as the maximum distance between the point and its neighbours and that point's k -distance. Intuitively, greater average reachability distance means that even the nearest neighbours are far away, therefore taking the inverse leads to a small LRD (local density). In fact, LRD is moderately, and positively, correlated with the absolute value of gradients (34.36% on average). Moreover, the greater the local density of members the greater the attack performance, since median LRD of members is moderately correlated with the restricted ROC AUC (20.65%), whereas median LRD of non members is only slightly, but negatively, correlated with the restricted ROC AUC (-5.88%).

Given this setup, AMI would reach a restricted ROC AUC in between [82.49%, 89.89%], which is significantly better than random guessing. Assuming an optimal threshold choice, that is one that maximizes TPR over FPR, the probability of a training sample to be identified as such lies in [89.38%, 95.95%], 92.6% on average, which implies non training samples to be missclassified with probability in between [15.77%, 34.46%]. It is worth mentioning that sampling members with greater local density, keeping the random non members selection, also yielded an improvement. Particularly, restricted ROC AUC would be in between [60.32%, 69.47%]. Assuming an optimal threshold choice, TPR and FPR are not statistically different than the first non random setup ([84.22%, 92.67%], 88.5% on average, and [14.97%, 23.7%], respectively). This setup would mimic the same data augmentation technique but such that some non training samples may be as well-represented as training ones, which is a more realistic scenario. Other than proving that AMI effectively works on a practical scenario, it also shows that it is possible to exploit local density of training samples for membership inference purposes. It has been previously noted that some data augmentation techniques may have a side effect on membership risk, even though traditionally they are commonly used to improve main task performance and reduce overfitting (therefore reducing loss based MIAs success). However, these results prove that a data augmentation technique that increases local density of training samples exposes the target model to a new vulnerability, despite eventually preventing loss based MIAs. This insight has been highlighted on prior work, namely by Carlini et al. [7], but it has not been yet empirically proved (as far as the author's knowledge goes). These experiments suggests that local density may be a cause of membership risk for some type of MIAs, which is a relevant direction for future work.

Additionally, one could argue that passive MIAs could also improve upon random guessing with a different target sample selection. Given that loss is not correlated with local density (correlation is less than 1%), than BBL and WBG are not expected to improve on the supramentioned setup, as they are based on the loss distribution and a product of it (gradient distribution). Not even WBIG would improve, as these models do not seem to exhibit a different behaviour when predicting training and non training samples from the early stages of model training. However, there might be a different setup for which passive MIAs are better than random guessing, which

⁶The number of neighbours used for this computation was equal to 3% of the whole training dataset.

would be to sample training samples among those that are easier to predict and non training samples among those that are harder to fit. Naturally, BBL, the simplest baseline, is expected to improve as, in this case, training samples would have lower loss than non training ones. However, this setup aims to mimic a practical setting in which there is some train test gap. Therefore, passive MIAs only exploit train test gap and when this gap is non-existent, then these attacks are not able to increase model leakage in order to improve, since they are passive. Unlike AMI, which can increase model leakage by reducing FPR, for example by increasing the number of epochs, as shown by Nguyen et al [42].

To evaluate inference of this sensitive attribute, experiments assumed that the adversary was most interested in inferring correctly the least frequent attribute value (taken as the positive label on restricted ROC AUC computation). Assuming a random target selection, an adversary conducting AAI* on a practical scenario would reach ROC AUC in between [73.39%, 82.32%] on members and [72.67%, 82.81%] on non-members, for a maximum FPR of 25%, with 90% of confidence. In its turn, an adversary conducting MIA2AIA would reach restricted ROC AUC in between [63.61%, 70.67%] on members and [63.73%, 70.67%] on non-members. Recall that MIA2AIA FPR was traditionally lower than AAI* FPR, even though not statistically lower on single-target attacks (see Figure 3.9). Despite that, AAI* performance, on this scenario, is significantly better than MIA2AIA regardless of the membership status. However, none of these attacks was able to increase attribute inference on members comparing to the performance on non members, which is explained by the same reason that explains the fact that AMI is random guessing under a random target sample selection. However, it may come as a surprise that MIA2AIA, an extra set of AMI, is significantly better than AMI on this setup. That is explained by the fact that the target sample paired with the incorrect attribute value (equivalent to a non member) belongs to the meaningless part of the feature space, therefore its neighbourhood is poorly represented on the dataset, enabling the adversary to improve its inference, as the gradient corresponding to the incorrect attribute value is naturally smaller.

Experiments with other datasets, namely CelebA whose results are presented in Appendix D, support the insight that AAI* outperforms MIA2AIA on well-generalized federated models, both in terms of TPR/FPR (theoretical evaluation) and restricted ROC AUC. Furthermore, these results (Figure D.3) show that AAI* significantly outperforms MIA2AIA, in terms of restricted ROC AUC, on members (close to 90% comparing to 60%) but on non-members is not significantly different than random guessing (since a model does not leak information regarding non-training samples), whereas MIA2AIA is not significantly worse on this set. Therefore, MIA2AIA did not increase the target model's leakage regarding training samples. These results are explained by the fact that AAI* FPR (non target samples activation) was not significantly different from 0, whereas MIA2AIA was significantly higher, as shown in Figure D.2.

As argued by Jayaraman et al. [28], AIAs performance should also be compared with data imputation techniques. Since the adversary is most interested in accurately inferring the least frequent label, a simple data imputation technique would be to set every target sample with that attribute value. Considering that the frequency of this attribute value (user is not at home)

is 16.5%, than this technique, on a balanced set with respect to the attribute value, reaches 16.5% of precision⁷. Given this baseline and assuming an optimal threshold choice, AAI* and MIA2AIA are both successful as both outperform the baseline, reaching precision in between [75.43%, 82.9%] and [72.63%, 77.32%] on members, respectively. Note that on a balanced set, the probability to make an accurate guess of this kind (setting every sample to the same value) is 50%, yet AAI* can make an accurate guess with probability in between [81.21%, 90.46%] and MIA2AIA in between [62.76%, 75.57%].

Given that the adversary is the central server, to increase its ability to make accurate inference, it could change some system-related parameters, such as number of clients and number of local epochs. Security questions may arise from these attempts, as FL systems may be equipped with safety measures, such as minimum number of clients at each training round. Note that reducing the number of clients could also imply a reduction on TPR, as the targeted client may not be included. However, one must not trust a system that hands privacy guarantees to chance. Nevertheless, these system-related parameters and their influence in the real threat of these attacks is yet to be explored.

All in all, besides effective theoretical results, these active attacks pose a real threat to FL systems, being considerably more powerful than their respective alternatives even on a practical scenario. Assuming that the neighbourhood of members is exploited via augmentation, AMI is able to accurately infer the membership status of 88.5% of the times (92.6% if non members are considerably less well-represented). Regarding attribute inference, AAI* and MIA2AIA can accurately determine if the user is at home in 85.8% and 69.2% of training samples (on average and assuming an optimal threshold), respectively. These results are relevant to prove that well-generalized models are exposed to MIAs and AIAs, namely active novel attacks, and to pave the way for a possible cause of membership risk - local density, that (by consequence) should also increase attribute risk.

3.2.7 Connection between membership and attribute inference

One of the primary goals of this work is to study the connection between different privacy attacks, particularly between MIA and AIA, to empirically show that membership risk is an upper bound for other privacy risks. That insight is supported by a direct comparison between AMI, AAI* and MIA2AIA empirical results. AMI TPR is always close to 100% regardless of the setup (except under adversarial limited knowledge), and the FPR is always under control (less than 3% on models of standard complexity). Whereas AAI* TPR is only close to 100% for a few target samples with FPR below 7% for those cases, and MIA2AIA only reaches TPR close to 90% for a bigger set of target samples and FPR always below 4%.

Theoretically, in the worst-case scenario, AMI TPR is reduced to random guessing, meaning $\frac{1}{2}$. On its turn, AAI* TPR is reduced to $\frac{1}{3}$, since it has 3 classes to randomly sample from

⁷Precision is a metric that balances true positives and false positives and it is calculated as follows $\frac{TP}{TP+FP}$

(non members whose gradient must be 0, positive members whose gradient must be positive and negative members whose gradient must be negative). For MIA2AIA, and letting p be the probability of the most frequent attribute value, TPR is reduced to: $\frac{1}{2}\frac{1}{2} + \frac{1}{2}\frac{1}{2}p = \frac{1+p}{4}$, where the first part of the expression stands for the probability of only one neuron being activated. The second part of this expression stands for the probability of an inconclusive attack (meaning both neurons were activated) followed by data imputation to lead to an accurate inference, for which the probability of being correct is precisely p . Analogously, the worst case scenario FPR is $\frac{1}{2}$, $\frac{2}{3}$, and $\frac{1}{2}$, respectively. Note that an attack for which TPR is equal to FPR is ineffective, as the adversary needs to commit the same amount of error for the same amount of correctly inferred information.

Results of AMI on a practical scenario from the previous section (3.2.6), show that AMI successfully exploits local density for malicious purposes. Similarly, AAI* should also benefit from a different data augmentation technique, particularly one that exploits a training sample's neighbourhood. Nevertheless, despite restricted ROC AUC (assuming a random target sample selection) being greater on AAI* case, this does not contradict previous insights, specifically that *membership inference risk is an upper bound for attribute inference risk*. Instead, it shows an improvement of the adversary's ability to increase leakage of the sensitive attribute value with AAI*. Following guidelines from Yeom et al. [62] to empirically analyse the connection between MIAs and AIAs, comparison between both adversaries must be through membership and attribute advantage⁸, defined by the same authors and explained in Chapter 2. Therefore, and assuming an optimal threshold choice, AMI achieves [10.93%, 17.07%], AAI* [-9.15%, 2.48%], and MIA2AIA [-7.75%, 7.75%], which shows that membership advantage is significantly greater than attribute advantage. Note that despite restricted ROC AUC on members, as well as on non members, being greater for AAI*, its attribute advantage is smaller than that of MIA2AIA, although only on average. This might be explained by the tendency of AAI* to incur in a greater amount of false positives.

Indeed, membership inference is a much simpler task than attribute inference, as the latter is actually an extra step of the former. Meaning AIAs aim to infer the sensitive attribute value of training instances, thus one must start by identifying those samples. Please note that worst-case analysis does not contradict previous findings (Salem et al. [47] and Yeom et al. [62]) regarding MIA and AIA being mutually reducible: in the worst-case scenario membership risk is greater than attribute risk, in the best case scenario they are equal. The same holds for the practical analysis based on adversarial advantage. All in all, *theoretical insights and empirical results show that membership inference risk is an upper bound for attribute inference risk*.

⁸Membership advantage is given by $TPR - FPR$, whereas attribute advantage is given by $TPR_{\text{members}} - TPR_{\text{non members}}$

3.3 Summary

In this chapter, novel active membership and attribute inference attacks were developed (Section 3.1) and proven to be powerful (Section 3.2), outperforming state-of-the-art approaches. Furthermore, this chapter intended to overcome some limitations of prior work on privacy attacks, namely develop powerful attacks on a realistic scenario, measure attack performance adequately, and study the connection between different types of privacy attacks. Ultimately, the goal was to show that well-generalized models are vulnerable to these powerful active membership and attribute inference attacks, contrasting with prior work that constantly attacks extremely overfitted models (Shokri et al. [40]). For that purpose, novel active membership and attribute inference attacks (Sections 3.1.3 and 3.1.4) were compared to passive MIAs (Section 3.1.2) and a MIA-based AIA (Section 3.1.5), used as baselines. These novel active attacks were designed to be effective on a realistic scenario, namely by not requiring access to individual gradient updates nor unrealistic FL-protocol conditions. An overview of the theoretical results highlights that it is possible to successfully encode membership and sensitive attribute information incurring in a small error in terms of neuron activation. Remarkably, AMI is able to identify a training sample 92.6% of the times, on a practical scenario by exploiting the neighbourhood of training samples, whereas AAI* is able to determine that the user is not at home in 85.8% of the times on a practical scenario.

Both theoretical (neuron activation) and realistic (gradient uncertainty) evaluation methodologies aimed to overcome a limitation of prior work, meaning adequately measured attack performance. The realistic evaluation (in Section 3.2.6) aimed to account for the cost of false positives on the gradient of the malicious neuron, which must not deceive an adversary from conducting the malicious task. In fact, previous active attacks of this kind (Nguyen et al. [42]) do not evaluate attack performance under gradient uncertainty. Even though the relation between both types of evaluation is clear - the overlap between members and non members distribution depends on FPR, what is not so clear is the amount of FPR that still allows for a sufficiently accurate inference. Therefore, this research not only evaluates attack performance adequately, but also measures the real threat posed by active attacks.

An extensive study of attack performance depending on several attack components aimed to test the robustness of these novel active attacks under several setups. Particularly, parameters controlled by the adversary, which could be tuned conveniently to improve its inference, that is the number of target samples, model complexity, and learning protocol. Parameters that are not controlled by the attacker were also tested to see if any of them would impact attack performance to a point of deceiving the adversary from even attempting to perform an attack (limited adversarial knowledge, and sensitive attribute distribution). Among these attack components, the number of target samples and model complexity impacted AMI, MIA2AIA, and AAI* results the most (Sections 3.2.3 and 3.2.4, respectively). Particularly, more complex target models give extra advantage to an adversary conducting AMI, given that, not only is it able to target more samples, but also does it while reducing FPR. MIA2AIA also benefits with increasing model complexity

and the number of target samples is responsible for most of that improvement. On the other hand, AAI* seems to benefit from more simple models, despite their reduced capacity to target more samples. However, AAI* only improves on MIA2AIA for a smaller set of target samples, at least for this sensitive attribute. Nevertheless, MIA2AIA depends on the sensitive attribute distribution to deal with inconclusive attacks, whereas AAI* does not. Moreover, adversaries with limited knowledge were not capable of inferring membership or attribute information, showcasing that MIAs and AIAs require a certain level of adversarial knowledge. Finally, the learning protocol, or more generally a different initialization, do not affect attack performance.

However, from theory to practice some performance may be lost. On a practical scenario (Section 3.2.6), one must allow a small margin of error in order to make inference. An adversary conducting AMI can achieve great restricted ROC AUC ([82.49%, 89.89%]) from expecting a malicious neuron’s gradient to be higher when it is trained to overfit a member, particularly one with high local density comparing to under represented non training samples. In its turn, an adversary conducting AAI* can achieve even higher restricted ROC AUC of randomly selected training samples ([73.39%, 82.32%]) from expecting a smaller gradient of neurons trained to overfit negative samples. However, MIA2AIA achieves a significantly lower restricted ROC AUC ([63.61%, 70.67%]) from expecting the correct neuron to have a greater gradient. More than showcasing the real threat of these novel active attacks, this evaluation paves the way for a relevant direction of future work - local density as a cause of membership risk.

One of the primary goals of this work is to study the connection between membership and attribute inference to empirically show that membership risk is an upper bound of attribute inference risk. Conceptually, attribute inference is an extra step of a MIA, therefore one can only infer the sensitive attribute (given access to a ML model) of a sample whose positive membership status can be determined by a MIA, which is theoretically supported by Salem et al. [47] and Yeom et al. [62]. Results, in terms of neuron activation, show that AMI always achieves greater results in terms of TPR and FPR when compared to AAI*, even though MIA2AIA achieves equally small FPR (if not smaller) its TPR is not as high. However, on a practical scenario, AMI is not better than random guessing, assuming a random target selection, fact explained by use-case specific limitations. But comparing active MIAs with active AIAs using membership and attribute advantage (defined by Yeom et al. [62]), membership advantage is significantly higher than attribute advantage for both active attacks. Therefore, on a practical scenario, membership risk is greater than attribute inference risk.

All in all, this chapter introduces novel active membership and attribute inference attacks that outperform state-of-the-art alternatives. This study shows that well-generalized models are vulnerable to these active attacks, that pose a real threat to FL systems, suggesting that traditional countermeasures (to improve generalization) do not mitigate them. Thus, effective defense mechanisms are needed, ideally reducing model utility loss, since the target model is able to recover from these attacks (Section 3.2.5).

Chapter 4

Defense Mechanisms

From the state of the art and the previous chapter, the reader should already be convinced that active attacks are underexplored and pose a real threat to Federated Learning (FL) systems. Defense mechanisms to mitigate them are also underexplored, particularly attack detection strategies that are expected to reduce model utility loss given that the mitigation phase is only applied when an alarm is triggered. Given that Active Membership Inference (AMI) and Active Attribute Inference (AAI) do not exploit the difference between target model’s behaviour on members and non-members, state-of-the-art defenses, such as differential privacy and regularization techniques, do not mitigate these attacks. Local Differential Privacy (LDP), proposed by Nguyen et al. [42], is also not effective at detecting AMI, as the adversary can adapt AMI formulation in order to train the malicious neuron to be activated by the perturbed versions of the target sample, which means that the adversary can evade the defense. Instead, detecting and responding to ongoing attacks presents a viable alternative, especially since recovery from AMI and AAI* can be achieved in a few training rounds, minimizing model utility loss.

This chapter introduces and evaluates three detection strategies tailored to different levels of the defender’s knowledge: black-box with hard labels (Section 4.2.1), black-box with soft labels (Section 4.2.2), and white-box (Section 4.3). All of these defenses monitor some metric, either an evaluation or a divergence metric, and detect an ongoing attack, or a malicious neuron, if a significant perturbation to the expected convergence of that measure is identified. To help the reader understand these approaches before diving into details, an overview of how these defense mechanisms work is presented in Section 4.1.

Based on Section 2.5, these approaches cover previously identified limitations, such as ineffective defenses against privacy attacks and low model utility loss of the latter approach (white-box). The major innovation is that these strategies offer robust defense mechanisms against both AMI and AAI*, enhancing privacy-aware FL systems by providing a defense against multiple types of privacy attacks that only interferes with the learning process when an attack is detected, which is the key to reduce model utility loss. Furthermore, these approaches overcome the lack of detection mechanisms to detect active privacy attacks, since most attack detection algorithms aim to detect cybersecurity (Chen et al. [9] and Janardhana et al. [27]) or passive

privacy attacks (Ko et al. [30], Li et al. [33], and Chen et al. [11]), conducted by outsiders, through malicious query detection.

4.1 Overview

This section outlines the methodology behind both black and white-box defense strategies. Figure 4.1 illustrates the general methodology of defense mechanisms applied by each client in a FL system. On the top of this figure, a common FL system is represented in which several participants articulate with the central server to collaboratively train a global model. The contribution of each client is aggregated without the intervention of any entity (assuming that Secure Aggregation is applied) and sent to the central server. The role of the central server is to join the aggregated contribution from this training round to the global model from the previous round, sending it back to clients to start another training round.

At some point of this training process, the central server performs an active attack in which it locally changes the global model so as to leak information. This is illustrated in this figure by the red arrows connecting the central server to each client, which receives a malicious global model, as previously outlined in Section 3.1.1 and further detailed in the same chapter. However, each client can apply a defense mechanism locally, since it cannot trust the central server to collaboratively defend against these threats. The bottom of this figure explains how these defense mechanisms work according to the knowledge of the defender (e.g. application). Each defense is divided into a detection (indicated by yellow triangle) and a mitigation (indicated by red triangle) phase.

A defender with black-box access can detect these active attacks by comparing train and test performance using the current and the previous model received from the central server. The most limited type of black-box access, that is access to the predicted class only, allows accuracy comparison, whereas access to the output vector enables Area Under the Receiver Operating Characteristic Curve (ROC AUC) comparison. On the other hand, if the defender has white-box access, meaning access to the global model parameters, then it can detect maliciously modified neurons (used to leak precise membership or attribute information) by comparing current and previous weights distribution of each neuron. Details on how this comparison works are latter explained in Section 4.3. In this figure, benign weights distribution is associated with a bell-shaped histogram, whereas the malicious is associated with a skewed one. This linkage is supported by empirical evidence, latter explained in Section 4.3. If an alarm is triggered, the mitigation phase is applied in which a client using a black-box defender must leave the system, while one using a white-box defender can still participate in this training round after replacing the malicious by the benign neuron.

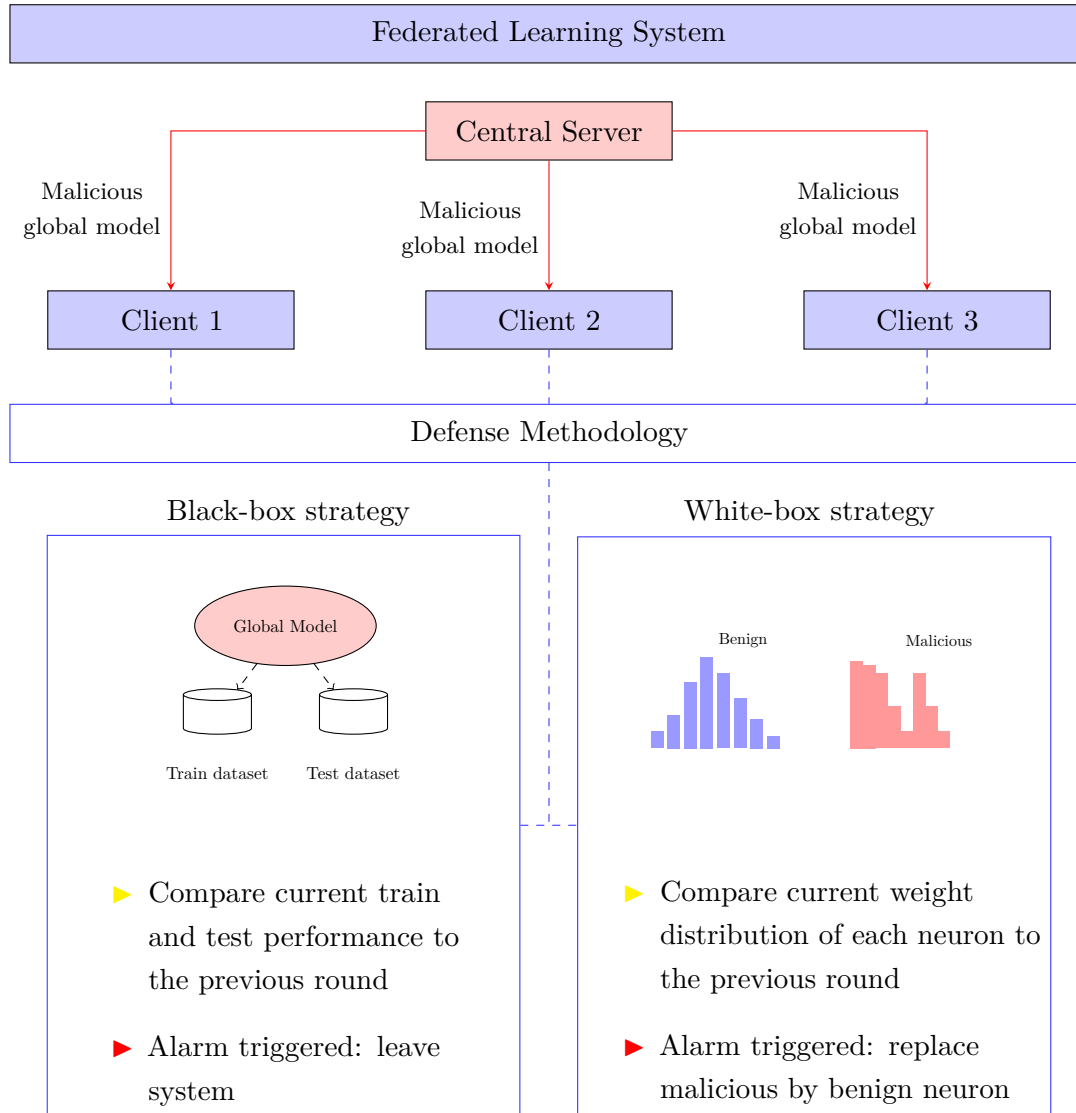


Figure 4.1: Overview of defense mechanisms applied by each client on a FL system.

4.2 Black-box Attack Detection

In order to detect an active attack, evaluation metrics, such as accuracy and ROC AUC, monitorization are used to identify a significant perturbation to the expected convergence. However, not every practical setup, or not every client, is suitable for each of these detection strategies, as monitoring ROC AUC requires a soft label (confidence score) access from the defender side, or the client's ability to calculate it. Therefore, for cases in which these conditions are not met, accuracy can be monitored instead, as it only requires access to the predicted class. Note that these detection strategies are employed before local training starts at the beginning of a new round, as local training, after the attack, can reduce the probability of detecting an ongoing attack by reducing the drop of evaluation metrics during training.

4.2.1 Black-box Attack Detection based on Accuracy (BADAcc)

Black-box Attack Detection based on Accuracy (BADAcc), inspired in well-known drift detection statistical control algorithms, such as Statistical Process Control (SPC, Gama et al. [16]), uses the 3 standard deviations (3-sigma) rule and detects an ongoing attack if local accuracy drops more than expected. This is a statistical-based approach that detects if there is a deviation from the mean of more than 3 times the standard deviation of the previous training round, which is equivalent to falling out of the 99.7% confidence interval, approximately. This strategy is applied at every training round, before local training.

In practice, BADAcc calculates the expected probability and respective variance of the 0 – 1 loss at each training round, since $0 - 1 \text{ loss} = 1 - \text{accuracy}$. Under normal conditions, it is expected that the error converges to a lower bound, or equivalently the accuracy converges to an upper bound. Under an attack, the model behaviour is perturbed and the error increases significantly, resulting in a significant accuracy drop. Note that the 0 – 1 loss of a sample follows a Bernoulli distribution, therefore the 0 – 1 loss of a set of samples follows a Binomial distribution. For a sufficient number of samples, Binomial approaches Gaussian distribution, thus one can fully describe the error distribution with only 2 sufficient statistics: average and standard deviation. In order to describe the convergence of the error distribution, 2 statistics are kept in memory: the error rate p_{min} and its standard deviation s_{min} . Their role is to represent the lower and narrower error interval, which represents the best stage of the model's error so far. The full BADAcc algorithm is presented in Algorithm 1, where these variables are updated at the beginning of each training round (lines 11-13). At every training round, a defender receives the current error and number of samples, calculates p (expected error probability) and s (respective standard deviation) which are compared to the lower and narrower error interval (lines 6-7). Then the defender calls the function UPDATE (lines 3-14) to detect whether there is a ongoing attack (lines 16-19).

Other than working under the most restricted model access (from the defender side), this approach guarantees statistical properties, that is a significant perturbation to the expected convergence is only classified as an ongoing attack when it falls out of the 99.7% confidence interval. Even though this precision can be adjusted as needed, for example reducing in-control margin from 3 times sigma to 2 times sigma (approximately 95% of confidence). However, this strategy is user dependent, as dataset size and local accuracy from previous rounds may influence attack detection, other than a possibly different precision level per user.

4.2.2 Black-box Attack Detection based on ROC AUC (BADAUC)

Similarly to the previous approach, Black-box Attack Detection based on ROC AUC (BADAUC) identifies a significant perturbation to the expected convergence of an evaluation metric. The sensitiveness of the baseline of accuracy to class imbalance and the fact that its computation is threshold dependent are the reasons behind the choice of ROC AUC monitorization instead

Algorithm 1 BADAcc algorithm

```

1:  $p_{min} < 1$ 
2:  $s_{min} < \infty$ 
3: function UPDATE( $n, e$ )                                ▷ Total and number of incorrectly predicted samples
4:    $p \leftarrow e/n$ 
5:    $s \leftarrow \sqrt{p(1-p)/n}$ 
6:   if  $p + s \geq p_{min} + 3 * s_{min}$  then
7:      $status \leftarrow alarm$ 
8:   else
9:      $status \leftarrow normal$ 
10:  end if
11:  if  $p + s < p_{min} + s_{min}$  then                        ▷ Update lower and narrower error interval
12:     $p_{min} \leftarrow \min(p, p_{min})$ 
13:     $s_{min} \leftarrow \sqrt{p_{min}(1-p_{min})/n}$ 
14:  end if
15:  return  $status$ 
16: function DETECT( $n, e$ )                                ▷ Total and number of incorrectly predicted samples
17:    $status \leftarrow UPDATE(n, e)$ 
18:   if  $status$  is alarm then
19:     Leave system
20:   else
21:     Continue
22:   end if
23: end function

```

of accuracy. Accuracy can hide a significant drop in model's performance because the baseline is too high (on class imbalance problems), or it can increase false alarms due to an inadequate threshold choice. In this case, an ongoing attack is detected if the distance (difference in absolute value) between the current and previous ROC AUC is greater than a preset threshold, therefore being a distance-based approach. Note that this approach does not preserve statistical properties, as there is no statistical guarantee of how rarely a normal situation falls from the preset threshold onward interval.

Ideally, one would apply the same strategy as BADAcc to detect an ongoing attack with statistical properties, only this time monitoring ROC AUC instead of accuracy. However, since ROC AUC's distribution is hard, if not impossible, to represent through sufficient statistics, one would need to store more than 2 statistics to represent current ROC AUC. Moreover, the notion of ROC AUC for each sample hinders a statistical-based attack detection, since one does not have a large set of samples to build the confidence interval. Instead, the defender has ROC AUC of consecutive rounds for which independency is not guaranteed (needed to construct a confidence interval). Therefore, a defender cannot detect an ongoing attack through ROC AUC monitorization using BADAcc algorithm.

This approach works under a restricted model access, but it is threshold dependent. This

hyperparameter may be tuned, previously or adaptatively during training, according to the use-case. Different clients may choose different thresholds and, in case of an adaptative threshold during training, each client must tune this hyperparameter based on local information, since it cannot trust the central server for a collaborative tuning. Moreover, it is also user dependent, as local ROC AUC from previous rounds may influence attack detection, other than a different threshold per user.

4.3 White-box Attack Detection and Mitigation (WADM*)

Black-box attack detection mechanisms, presented in the above section, may result in significant false alarms, especially for non-IID (independent and identically distributed) data. Evidently, a user with a particularly hard to fit data might result in a missed alarm. To avoid this local dataset dependency, an alternative is to use global model's weights. These are a product of the aggregation of all participating clients' updates, thus removing the individual dependency on local datasets, and consequently being more robust. Additionally, leaving the system after an ongoing attack was detected incurs in the highest model utility loss, despite effectively protecting users' privacy. White-box Attack Detection and Mitigation (WADM) emerges as an alternative to BADAcc and BADAUC to overcome the aforementioned issues by identifying malicious neurons and changing their distribution instead of leaving the training.

The difference between a benign and a malicious neuron relies on the evolution (between training rounds) of their weights distribution. Figures 4.2 and 4.3 illustrate the evolution of the weights distribution of a standard neuron and the change in weights distribution of a neuron that was selected to leak membership/attribute information, respectively. From Figure 4.2, one can see that the weights distribution of a standard neuron is expected to vary little between training rounds, at least when the model has converged. However, both active attacks perturb this small variation, causing a significant change in weights distribution, as observed in Figure 4.3. Particularly, both attacks extend the range in which weights fall in, going from $[-0.2, 0.35]$ and $[-0.25, 0.15]$ to $[-1.0, 0.75]$ and $[-4.0, 6.0]$, respectively. Furthermore, these attacks also distort the shape of the weights distribution of a malicious neuron, which goes from close to Gaussian/bell-shaped (right plot) or slightly biased (left plot), in Figure 4.2, to remarkably biased, in Figure 4.3. Therefore, to distinguish standard from malicious neurons one should summarize this clear distortion of weights distribution into a single metric (to automate the detection process), for example using KL-Divergence.

The procedure of WADM* in Algorithm (2) is divided in 2 phases: detection (lines 1-10) and mitigation (lines 11-17). In the detection phase, KL-Divergence (line 4) is used to measure the difference between current and previous weights distribution of each neuron, as shown in equation (4.1). Note that to account for distribution shift, one must represent both distributions on the same range, determined by the maximum and minimum of both sets of weights. Since KL-divergence is not a symmetrical metric, the minimum between both directions is taken to be extra cautious. A neuron is flagged as malicious if this value exceeds a preset threshold (lines

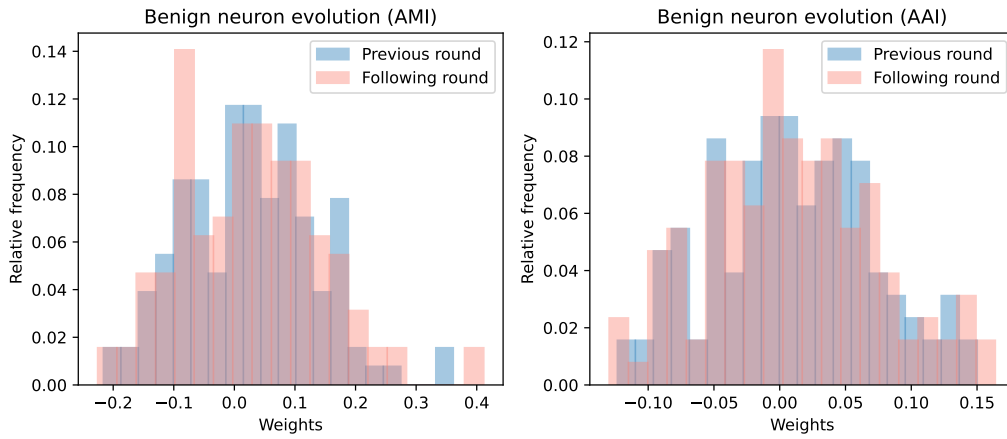


Figure 4.2: Evolution of the weights distribution of a standard neuron, computed from the fifth and sixth training round, once the model (of standard complexity) has converged. Each plot represents a different type of model (trained with ReLU and ELU respectively).

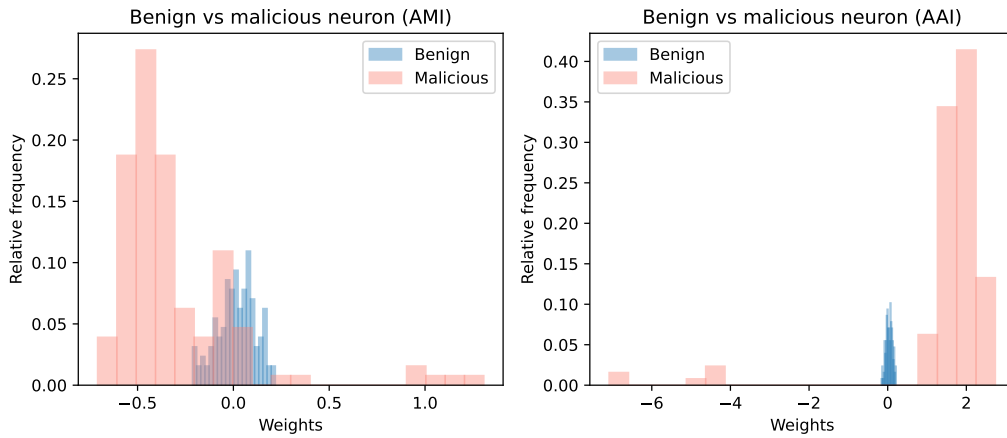


Figure 4.3: Change in weights distribution of a malicious neuron (after single-target attacks), computed from the fifth and sixth training round, once the model (of standard complexity) has converged. Each plot represents a different type of model (trained with ReLU and ELU respectively).

5-9), therefore being a distance-based approach.

$$\text{KL-DIVERGENCE}(X_{prev}, X_{cur}) = \sum_i X_{prev} \log \left(\frac{X_{prev}}{X_{cur}} \right) \quad (4.1)$$

where i ranges across the number of bins used to represent both distributions

From equation (4.1), one can see that if a bin from the distribution X_{cur} has 0 frequency, then KL-Divergence is infinite. Other than taking the minimum between both directions, one can choose an adequate number of bins to represent the weights distribution of each neuron, possibly reducing false alarms. Therefore, WABM ability to detect malicious neurons depends

both on the threshold and number of bins.

In the mitigation phase, the defender changes current weights to the previous weights of any neuron flagged as malicious, discarding the most recent ones (lines 12-13). A benign neuron keeps its weights and the previous weights are updated (lines 14-15). Algorithm 2 illustrates WADM* detection and mitigation phase, where lines 18-21 shows how a defender combines both phases.

Algorithm 2 WADM* Detection and Mitigation phase

```

1: function DETECT( $W_{prev}, W_{cur}, n_{bins}, threshold$ )  $\triangleright$  Previous and current neuron's weights, and
   number of bins to represent both distributions
2:    $Dist_{prev} = histogram(W_{prev}, n_{bins})$ 
3:    $Dist_{cur} = histogram(W_{cur}, n_{bins})$ 
4:    $div = \min(KLDiv(Dist_{prev}, Dist_{cur}), KLDiv(Dist_{cur}, Dist_{prev}))$ 
5:   if  $div \geq threshold$  then
6:      $status \leftarrow alarm$ 
7:   else
8:      $status \leftarrow normal$ 
9:   end if
   return  $status$ 
10: end function
11: function MITIGATE( $W_{prev}, W_{cur}, status$ )  $\triangleright$  Previous and current neuron's weights and detection
   status
12:   if status is alarm then  $\triangleright$  Change neuron's weights
13:      $W_{cur} \leftarrow W_{prev}$ 
14:   else  $\triangleright$  Update previous neuron's weights
15:      $W_{prev} \leftarrow W_{cur}$ 
16:   end if
   return  $W_{prev}, W_{cur}$ 
17: end function
18: function WADM( $W_{prev}, W_{cur}, threshold$ )  $\triangleright$  Combine detection and mitigation phases
19:    $status \leftarrow UPDATE(W_{prev}, W_{cur}, 5, threshold)$   $\triangleright n_{bins} = 5$  to improve robustness
20:    $W_{prev}, W_{cur} \leftarrow MITIGATE(W_{prev}, W_{cur}, status)$  return  $W_{prev}, W_{cur}$ 
21: end function

```

Note that this approach does not preserve statistical properties and, similarly to BADAcc and BADAUC, it is applied before local training starts to prevent any change in neurons distribution (due to local training) that might influence the detection. It is worth mentioning that in the second consecutive round (first time WABM is applied), the baseline of neurons weights is random (from the first training round), therefore false alarms are likely to occur for lower thresholds particularly in early training rounds. This hyperparameter may be tuned, previously or adaptatively during training, according to the use-case. Moreover, different clients may choose different thresholds and, in case of an adaptive threshold during training, each client must tune this hyperparameter based on local information. This approach only works under privileged model access, i.e. the defender has access to the model parameters, and requires memory to store previous and current model weights at each round.

4.4 Analysis

This section presents a comprehensive analysis of defense mechanisms proposed in this chapter, applied on the same real-world dataset (COP-MODE) as the proposed membership and attribute inference attacks. Several components that may have a significant impact on defense performance are studied, such as number of target samples and model complexity. Defense performance is also evaluated under different attack setups. First, details about practical setup are clarified in 4.4.1, then black and white-box attack detection results are presented in Section 4.4.2, 4.4.3 and 4.4.4, respectively.

4.4.1 Experimental Setup

All experiments conducted in this study were repeated 30 times to ensure the robustness and reliability of findings. A *confidence interval* of 90% confidence was calculated, providing a rigorous statistical guarantee for the reported findings. Note that the term *significant* is used throughout this result section to denote statistical significance of findings, meaning if the confidence intervals involved are disjoint. To account for a possibly different significance level, when intervals are disjoint but close to each other the expression *significant but for a small margin* (or similar) is used. The dataset used for these experiments is the same as in the results Section 3.2, whose details are summarized in 3.2.1.1. Similarly, the same target models are used to conduct these experiments, whose details are summarized in 3.2.1.2.

Regarding attack detection strategies, some parameters are fixed, except for those whose impact on defense performance is carefully studied. Precision level of BADAcc is fixed at 3 times standard deviation, since decreasing this value does not improve attack detection significantly and may increase false alarms. BADAUC has a single parameter - the threshold above which a drop in ROC AUC is considered abnormal. The following experiments present the best result in a range of 100 equally spaced possible thresholds in the range $[0.00005, 0.11]$. Black-box attack detection strategies are based on the monitorization of an evaluation metric, which might differ when calculated on training and testing sets. One could apply these strategies on the training, testing, or even on both sets. The following experiments of BADAcc and BADAUC are based on training set monitorization, as it revealed to be the best choice. Regarding testing set monitorization and having in mind that this set was not oversampled, BADAcc performance is significantly worst as the accuracy baseline (of a random classifier) is higher (approximately 66% compared to 50% on the balanced training set). Therefore, the training accuracy drop derived from an ongoing attack is higher and thus easier to capture. Performance of attack detection based on testing ROC AUC tends to be worst on average, but differences are not statistically significant. Finally, WADM* compares weights distribution represented on a histogram of 5 equally spaced bins which range from the minimum to the maximum value observed on both sets of weights. The threshold above which one considers KL-Divergence to be abnormally high is carefully chosen in the following experiments.

4.4.1.1 Metrics

Adequate defense performance measures are important to better judge the real effectiveness of defense mechanisms while preserving model utility. As for attack detection strategies, one must use True Positive Rate (**TPR**) (successfully detected ongoing attack) and False Positive Rate (**FPR**) (false alarm, meaning benign round classified as malign). In some cases, Missed Attacks, another metric, may be presented and it is such that summing TPR, FPR and Missed Attacks yields 100% for black-box strategies. Thus, Missed Attacks measures the probability of applying a detection mechanism and still be vulnerable to an attack. Note that WADM^{*} detects malicious neurons, therefore TPR stands for the probability of successfully detect malicious neurons and FPR stands for the rate of non malign neurons that are detected as one. Contrarily to black-box strategies, WADM^{*} is evaluated per round, as it does not force the client to leave the system). Meaning, results presented below are from applying WADM^{*} on the training round in which an attack occurs. Latter in this chapter (Section 4.4.4.5), WADM^{*} is also applied on early training rounds. To measure white-box attack mitigation performance, one can evaluate attack TPR and FPR after the mitigation phase. For black-box attack detection strategies, evaluating reduced attack TPR and FPR is irrelevant, since after an attack is detected the client must leave the system, thus preventing leaky gradients to be shared with the central server. Finally, one shall see throughout the rest of this chapter, that attack detection rate needs to be maximized, whereas attack performance needs to be minimized, as when developing a defense mechanism one should interpret results differently from the previous chapter, in which the goal was to maximize attack performance.

4.4.1.2 Adversarial setup

An adversary that is aware that clients may employ a defense mechanism against active attacks may try to reduce its influence in order to reduce the probability of being detected. Naturally, this scenario may reduce attack performance. Throughout the rest of this chapter, the effectiveness of these defenses is evaluated on different types of adversarial setups.

Experiments suggested that smaller learning rates decreased **AMI** and multi-target **AAI** performance, in most cases by slightly increasing FPR. If the adversary decides to decrease the learning rate, other than reducing attack performance (which is not ideal yet sometimes tolerable) it could also decrease its influence on the target model, both in terms of model performance and distortion of weights distribution. The intuition behind this strategy is that, with a smaller learning rate, smaller steps (gradient updates) are made to optimize the adversarial network. Fixing the number of epochs, the malicious neuron's weights distribution would not be so distort. This distortion could go unnoticed for certain thresholds. A similar thought can be applied for the first hidden layer whose weights would be less distort, allowing for a greater number of samples to activate these (and the subsequent), neurons¹, hopefully reducing attack performance

¹Insight empirically supported by the fact that FPR always increases, most of the times this increase is statistically significant.

drop. Although, how to master the trade-off between attack performance and target model performance is left as future work.

For empirical results of AMI in Section 3.2.3, the combination of hyperparameters was chosen to maximize attack performance (learning rate of 0.05 and 100 epochs). Given those great results (from the adversary’s point of view), one can wonder whether an adversary conducting AMI can balance attack performance maximization and reduced influence. Figure 4.4 illustrates AMI TPR and FPR evolution with an increase of the percentage of neurons controlled by the attacker, for each model complexity, and for 2 different setups: standard, in which the adversary maximizes attack performance, and restricted, in which the adversary aims to reduce its influence.

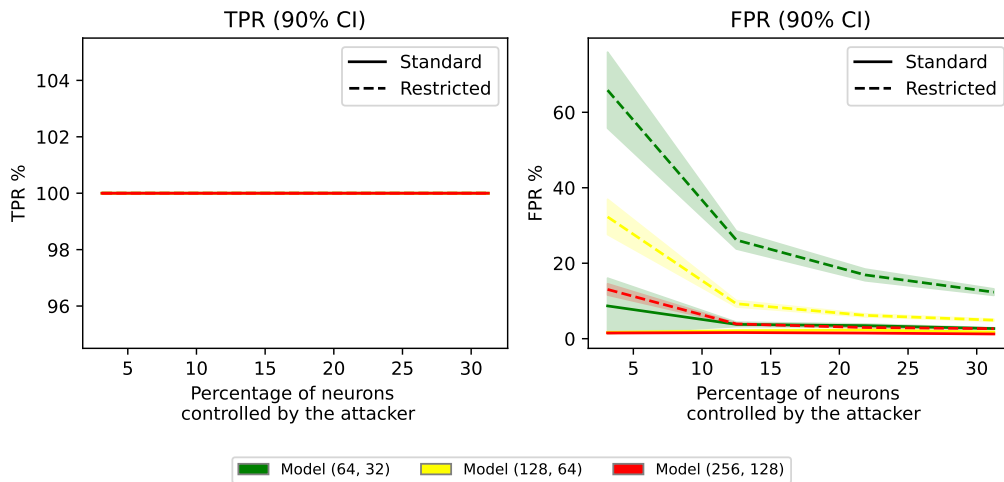


Figure 4.4: Multi-target AMI performance on standard (learning rate 0.05, 100 epochs) and restricted setup (learning rate 0.005, 50 epochs), on federated models of multiple complexity and for an increasing percentage of neurons controlled by the attacker

From Figure 4.4, one can see that TPR is still not statistically different from 100% under a restricted setup regardless of model complexity. Note that TPR of AMI over other model complexities is not visible due to the overlap with the TPR for more complex models. However, FPR increases in the restricted case, even though this increase is not statistically significant when the number of target samples increases, particularly for more complex models (as can be observed by the overlap for more than 10% of controlled neurons). However, controlling up to 31% of neurons is still not enough to remove the statistical significance of the increase of FPR on distinct setups over simpler models. Nevertheless, the decreasing trend of every curve in the left plot shows that FPR decreases when the number of target samples increases regardless of model complexity and adversarial setup. Note that, for comparison purposes, these experiments stop when the percentage of neurons controlled by the attacker reaches approximately 31%, since AAI* is ineffective for a percentage greater than 31% (Figure 3.10).

For empirical results of AAI* in Section 3.2.4, the combination of hyperparameters was chosen so as to maximize attack performance (learning rate of 0.05 for multi-target, 0.01 for single-target, and 1000 epochs). But TPR’s hard to achieve baseline poses challenges to an

adversary aiming to reduce its influence. Figure 4.5 illustrates TPR and FPR evolution with an increase of the percentage of neurons controlled by the attacker, for each model complexity, and for 2 different setups: standard, in which the adversary maximizes attack performance, and restricted, in which the adversary aims to reduce its influence.

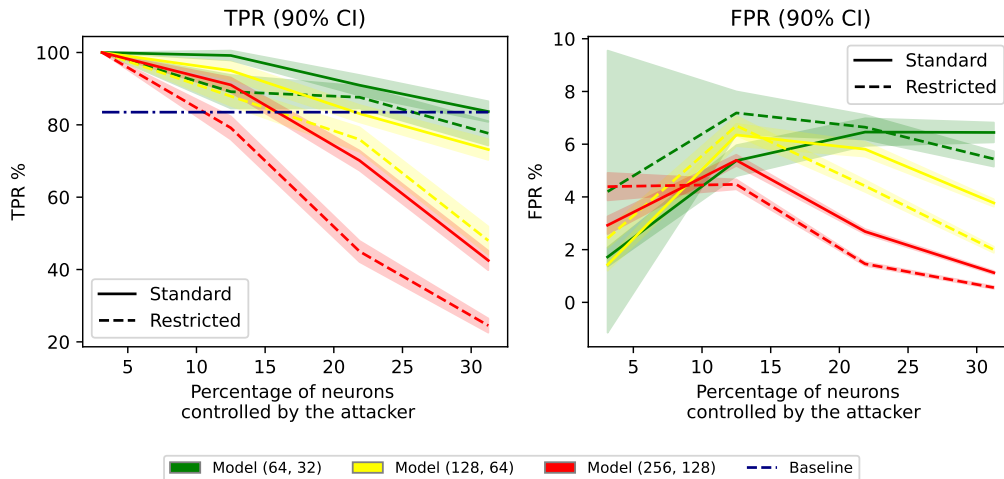


Figure 4.5: Multi-target AAI* performance on standard (learning rate 0.05, 1000 epochs) and restricted setup (learning rate 0.025, 1000 epochs), on federated models of multiple complexity and for an increasing percentage of neurons controlled by the attacker

From Figure 4.5, one can see that FPR is still small, even though a slight and not statistically significant increase was registered from a standard to a restricted setup regardless of model complexity (when few samples are targeted, which is when AAI* is the most effective). Experiments with attacks over simpler models that control a small percentage of neurons (3%), considered single-target attacks, incurred in an unusual high variance, but the average is not too distant from the rest of the experiments. However, TPR decreases significantly, up to a point where some of these experiments result in an ineffective attack, since their TPR is lower than the baseline (blue dashdotted line). Although AAI* on both adversarial setups has the same behaviour, TPR under a restricted setup decreases faster when compared to the standard setup.

4.4.2 Black-box Attack Detection based on Accuracy

In this subsection, the reader can find BADAcc experimental results for both AMI and AAI*, in which the effect of different components on attack detection performance, in terms of TPR, is evaluated in sections 4.4.2.1, 4.4.2.2, and 4.4.2.3, respectively for each of the studied components. Finally, false alarms (FPR) are evaluated separately in Section 4.4.2.4, an explanation of how local features affect attack detection performance is presented in Section 4.4.2.5, and a small summary is presented in Section 4.4.2.6.

According to experimental results from Chapter 3.2, the number of target samples and model complexity significantly impacted attack performance. Thus, it is relevant to see how both of

these components affect BADAcc performance. Moreover, one is also interested in finding if an adversary that is aware of a potential defense mechanism and is willing to exchange attack performance for a reduced probability of being detected, i.e. the supramentioned restricted setup, can escape accuracy monitorization based detection. Figures 4.6 and 4.7 illustrate TPR evolution when detecting AMI and AAI* with an increase of the percentage of neurons controlled by the attacker, for each model complexity, and for 2 different setups. Note that the y-axis limits are equal to ease comparisons. From these figures, one can see that BADAcc effectively detects both attacks, as TPR is significantly higher than 50% (baseline of random guessing).

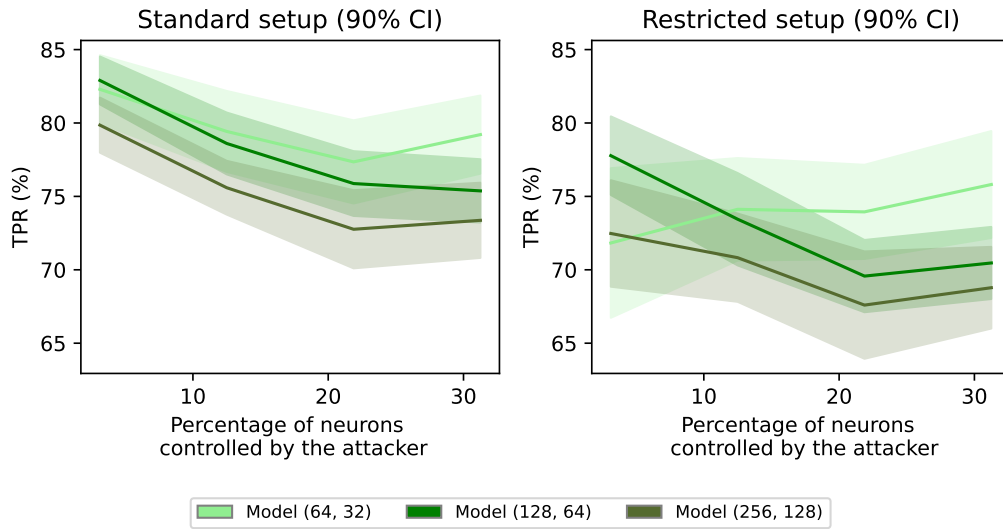


Figure 4.6: BADAcc performance on detecting AMI over federated models of multiple complexities, for distinct adversarial setups, and for an increasing percentage of neurons controlled by the attacker.

4.4.2.1 Impact of restricted setup

Based on Figure 4.6, an adversary willing to trade AMI performance for a reduced probability of being detected fails at this goal, since results of detecting AMI on a standard setup are not significantly different from those of a restricted setup, in most cases. Statistically significant differences are only registered on a small percentage of neurons controlled by the attacker, yet these are always marginal (at most 3% for simpler models and at most 1% for standard and more complex models). Albeit BADAcc performance on a restricted setup tends to be worst on average for every experiment. Note that precisely the combination of attack-specific components for which an adversary can marginally reduce detection probability are the ones that reduce attack performance the most. Particularly, AMI controlling 3% of the neurons over simpler models incurs in the largest attack performance drop, as well as the largest BADAcc TPR drop.

Regarding AAI* results, these present no statistically significant difference for distinct adversarial setups, not even a drop on average is registered on most cases, according to Figure 4.7. These findings show that BADAcc is robust to defense-aware adversaries, even though average

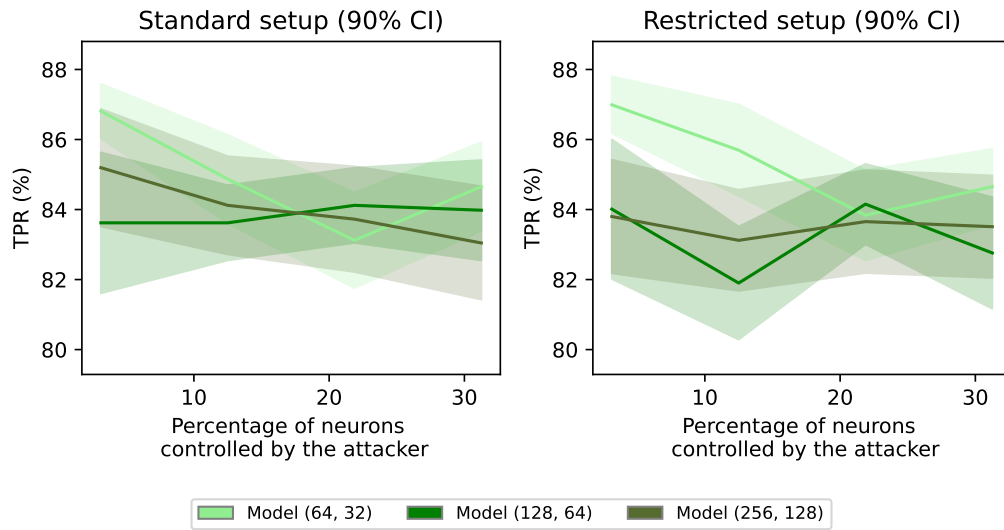


Figure 4.7: BADAcc performance on detecting AAI* conducted on federated models of multiple complexities, for distinct adversarial setups and for an increasing percentage of neurons controlled by the attacker.

detection probability drops marginally when conducting AMI on a restricted setup.

4.4.2.2 Impact of the number of target samples

Based on Figure 4.6, a membership attack that targets an increasing number of samples can be statistically harder to detect, as TPR is significantly different among some of the lower and greater percentages of neurons controlled by the attacker, fixing model complexity. Although these differences are mostly marginal. Even when not statistically significant, there is always a slight tendency to decrease TPR with an increase of the number of targets, as can be observed by the decreasing curves on both plots of this figure. This behaviour is slightly more accentuated on standard setup results. However, according to Figure 4.7, BADAcc performance on AAI* detection does not share the same behaviour, as no statistically significant differences were registered, not even a clear tendency to decrease TPR with an increase of the percentage of controlled neurons.

These results show that an ambitious adversary that increases the number of target samples can be equally detectable by BADAcc depending on other attack-specific components. Even though increasing the number of target samples reduces the detection probability, that reduction is marginal and only for AMI adversaries. Yet, precisely on those cases, the adversary not only reduces BADAcc performance, but also increases attack performance (by decreasing FPR).

4.4.2.3 Impact of model complexity

Based on Figure 4.6 and fixing other attack-specific components, higher or lower model complexity does not reduce AMI detection probability, since most of these results are not significantly different across distinct model complexities. However, allied with an increase of the percentage of neurons controlled by the attacker, it can reduce BADAcc TPR significantly. As it is the case for AMI detection, for which TPR significantly decreases with an increase of the number of samples. As it can be observed in the plot curve that, for more than 20% of controlled neurons, TPR decreases up to a point of being lower than it started (accounting with the margin of error). This observation holds for standard and more complex models on a standard setup, and only for models of standard complexity on a restricted setup. As with the rest of the attack-specific components, model complexity does not impact BADAcc performance on AAI* detection significantly, not even allied with other components, according to Figure 4.7.

These results show that a more complex model alone does not give extra advantage to the adversary even though, intuitively, it could still pass more information from one layer to the other, hopefully reducing accuracy drop derived from a malicious interference. In fact, AMI and AAI* impact on accuracy is, in most cases, independent of these components, as accuracy drops to a random guessing level or close, according to Figure 4.8.

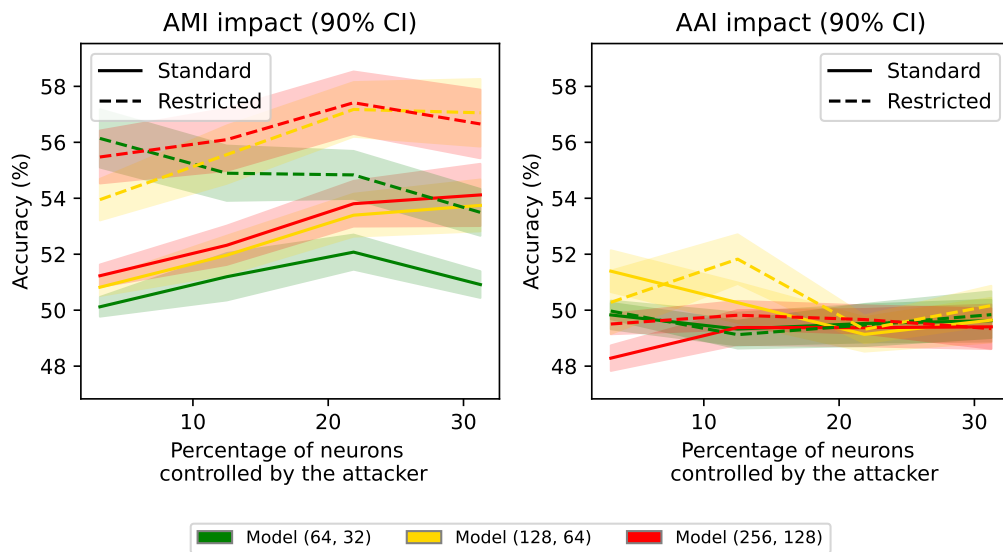


Figure 4.8: Impact of AMI and AAI* on training accuracy on federated models of multiple complexities, for distinct adversarial setups and for an increasing percentage of neurons controlled by the attacker

Figure 4.8 illustrates impact of AMI and AAI* on local training accuracy for varying attack components. In this figure, one can see that accuracy drops to a random guessing level after AAI* regardless of the adversarial setup, percentage of controlled neurons, and model complexity. However, accuracy drop after AMI can be statistically different depending on a combination of attack-specific components. In particular, the same network used to leak membership status of an

increasing number of target samples somewhat impacts accuracy differently. Intuitively, overfitting a higher number of samples may increase first Fully Connected (FC) layer’s generalization ability, potentially leading to a smaller drop in accuracy, particularly when the network itself has a higher capacity (more complex architecture). Moreover, the adversarial setup also leads to a statistically different impact on accuracy, namely when the attack targets fewer samples. Model complexity alone does not cause a significantly different drop in accuracy in most cases.

4.4.2.4 False Alarms

As any detection algorithm, minimizing false alarms (FPR) is the priority, as minimizing the other type of error is trivial (meaning all users would trigger an alarm on every training round, therefore reducing missed attacks but loosing model utility completely). False alarms occur before the malicious training round (the fifth round in this case) and are independent, by definition, of any adversarial parameter. Yet, they might depend on model complexity, because it is the only attack-specific component that must be determined a priori (before the attack) as it changes the target model. Figure 4.9 illustrates FPR per model complexity and type of attack, as each attack requires a slightly different model. From this figure, one can see that model complexity does not affect FPR significantly on both cases. Moreover, false alarms do not differ significantly when they occur on distinct types of models, which might be explained by the fact that both achieve a similar performance. Some local features are also expected to affect FPR, which is explored in the following section.

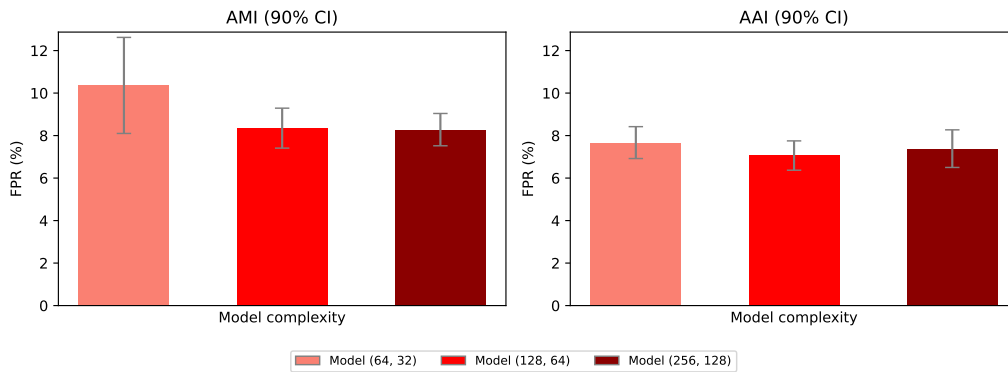


Figure 4.9: BADAcc FPR while detecting AMI and AAI* on federated models of multiple complexities.

4.4.2.5 User dependency

From the previous empirical results, neither the percentage of controlled neurons, adversarial setup or model complexity had a significant impact on BADAcc performance. Intuitively, one might think that local features must be playing an important role in attack detection, more specifically local accuracy and training dataset size. In order to show that these features affect attack detection through local accuracy monitorization, Figure 4.10 illustrates the Spearman

correlation matrix of these local features with both types of error: *False alarms* (FPR), and *Missed attacks*. In each cell, the interval of correlation values represents the maximum and minimum correlation registered among all experiments. That is, each correlation matrix summarizes the correlation matrix of each attack detection for several percentages of controlled neurons, both adversarial setups, and several model complexities. Please note that these are intervals of values, not confidence intervals, since the independence of experiments is not guaranteed. *Metric* stands for local accuracy before the malicious training round and *Size* stands for local training dataset size.

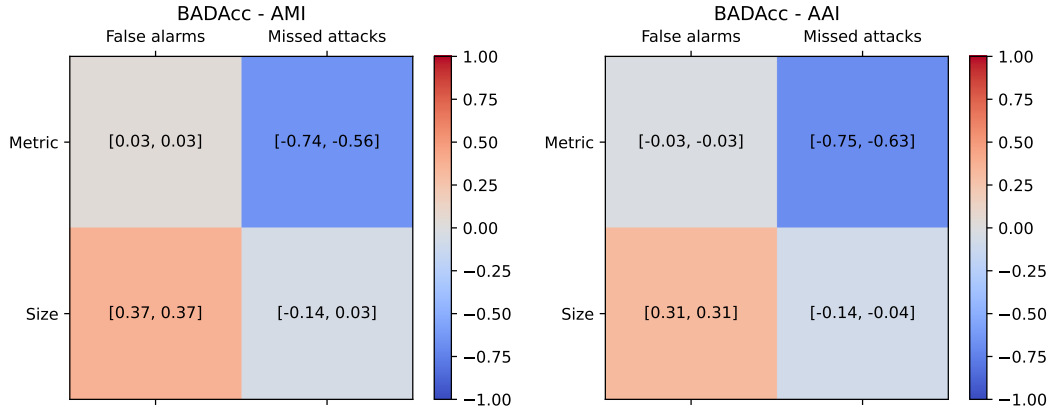


Figure 4.10: Correlation matrix of BADAcc results (of AMI and AAI* detection) with local features

From this figure, one can see that both *Metric* and *Size* play a significant role on BADAcc performance on AMI and AAI* detection, which does not differ according to the type of attack. Moreover, given that most intervals are narrow, the correlation structure does not differ when varying attack-specific features (see Figure 4.8). *Metric* has a moderate to strong negative correlation with *Missed attacks* and *Size* has a weak positive correlation with *False alarms*. This correlation structure reveals that the greater the local accuracy (*Metric*) the lower the probability of missing an ongoing attack (*Missed attack*), and the greater the local training dataset size (*Size*) the greater the probability of incurring in a false alarm (*False alarm*). Given that accuracy drops to a random guessing level regardless of any attack-specific feature, it becomes natural that the greater the local accuracy the greater the drop, therefore the easier the detection. However, it may sound surprising that larger training datasets increase false alarm probability. This can be explained by the standard deviation formula used to calculate the error margin within which any variation is classified as normal:

$$\sigma = \sqrt{\frac{p(1-p)}{n}}$$

Where n denotes *Size* and p current 0 – 1 loss (complementary of accuracy). This safety margin (σ) becomes narrower as the local training dataset becomes larger, thus any small decrease of accuracy may fall out and be incorrectly classified as an ongoing attack.

However, false alarms correlation with training dataset size is only weak. In fact, false alarms are mainly caused by 3 to 6 users, depending on model complexity (even though FPR does not change significantly). Among these users, the training dataset size does not appear to be the main reason for an abnormally high false alarm probability. Instead variability of local accuracy allied with a large dataset, in some cases, explains why these users are prone to false alarms.

4.4.2.6 Summary

As the detection strategy that requires the least amount of trust in the defender's party, BADAcc achieves great results (TPR around 80%), therefore being an effective defense mechanism. These results show that BADAcc is robust to defense-aware adversaries and to several changes in attack-specific components, yet it incurs in between 7% to 12% of false alarms. Moreover, both types of error (false alarms and missed attacks) depend on the specific user, which could result in users that systematically cannot use the system (consequence of false alarms) or are constantly exposed to both attacks (consequence of missed attacks). These findings motivate the need to reduce both types of error, particularly their cost/consequence, and user dependency.

4.4.3 Black-box Attack Detection based on ROC AUC

In this Section, the reader can find BADAUC experimental results for both AMI and AAI, in which the effect of different components on attack detection performance is evaluated in sections 4.4.3.1, 4.4.3.2, and 4.4.3.3, respectively. Additionally, the question of threshold selection is addressed in Section 4.4.3.4, as well as how local features affect attack detection performance in Section 4.4.3.5. Finally, Section 4.4.3.6 presents a small summary relating BADAUC and BADAcc results.

Similarly to BADAcc results section, it is relevant to see how does the number of target samples and model complexity affect BADAUC performance, as well as the type of adversarial setup to see if an attacker can balance the trade-off between attack performance and probability of being detected. Figures 4.11 and 4.12 illustrate TPR and FPR evolution when detecting AMI and AAI* with an increase of the percentage of neurons controlled by the attacker, for each model complexity, and for 2 different setups. Note that the y-axis limits are equal for each metric being evaluated (TPR and FPR) to ease comparisons. These results represent the best defense performance, assuming an optimal threshold choice, which balances TPR, FPR, and missed attacks equally. Performance for a different threshold choice is further studied latter in this section. From these figures, one can see that BADAUC effectively detects both attacks, as TPR is significantly different from the baseline of random guessing, being always close to 90%. Moreover, FPR is always around 3%, which is significantly lower than BADAcc FPR (see figure 4.9). Note that BADAUC FPR may vary according to attack-specific components and adversarial setup, contrarily to the previous strategy, as BADAUC is threshold dependent.

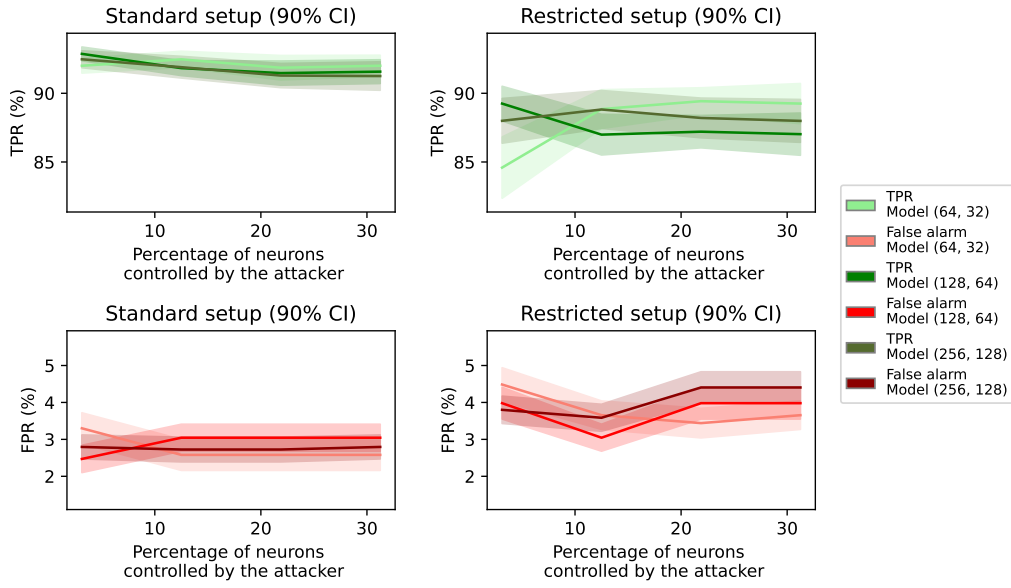


Figure 4.11: BADAUC performance on detecting AMI over federated models of multiple complexities, for distinct adversarial setups, and for an increasing percentage of neurons controlled by the attacker.

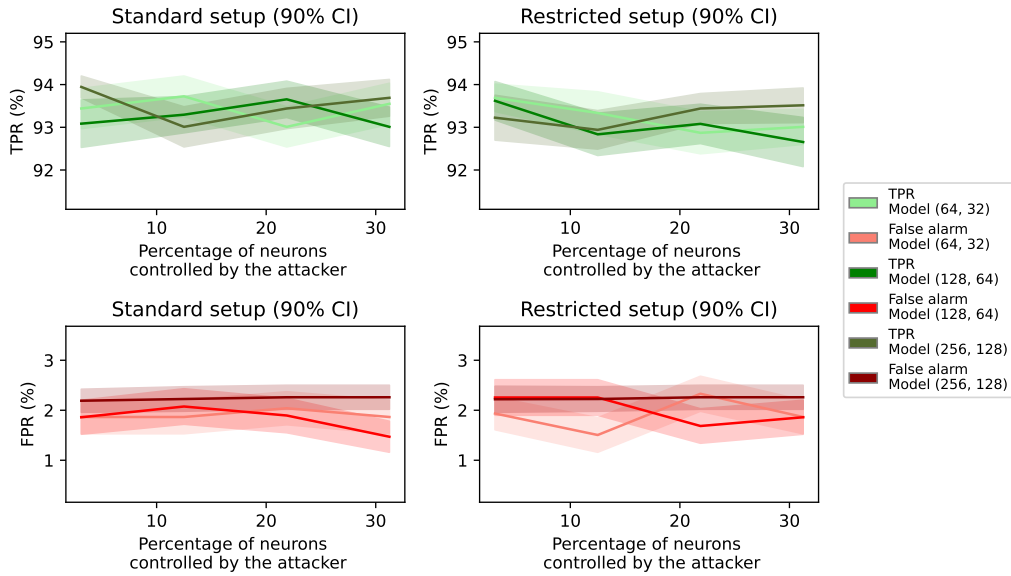


Figure 4.12: BADAUC performance on detecting AAI* over federated models of multiple complexities, for distinct adversarial setups, and for an increasing percentage of neurons controlled by the attacker.

4.4.3.1 Impact of restricted setup

From Figure 4.11, one can see that an adversary willing to trade AMI performance for a reduced probability of being detected fails at this goal, since results of detecting AMI on a standard setup are statistically different from those of a restricted setup but for a very small margin (at most 1%

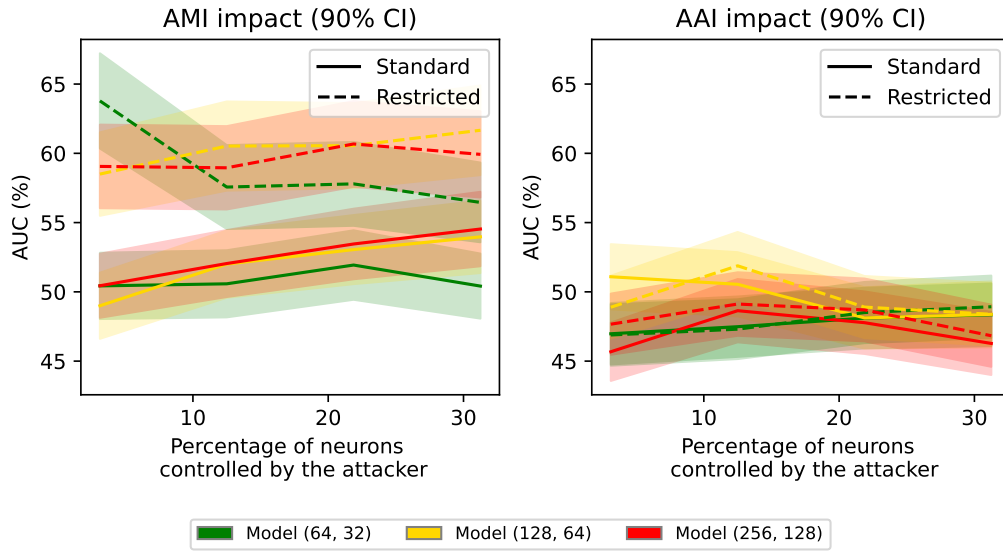


Figure 4.13: Impact of AMI and AAI* on training ROC AUC on federated models of multiple complexities, for distinct adversarial setups and for an increasing percentage of neurons controlled by the attacker.

for almost all experiments) and TPR is still very high. This small decrease leads to an equally small increase of FPR. This behaviour is to be expected since AMI impacts training ROC AUC differently depending on the adversarial setup, as clearly illustrated in Figure 4.13. However, observed differences in the impact slightly decrease along with an increase of the percentage of neurons controlled by the attacker, as can be confirmed by a slightly increasing trend of ROC AUC on the standard setup, while the impact of the restricted setup does not share the same behaviour.

In its turn, no statistically significant differences, both in terms of TPR and FPR, are registered on AAI* detection results, according to Figure 4.12. Non significant differences are also observable in Figure 4.13 and explain BADAUC behaviour on AAI* detection, since if varying adversarial setup does not impact ROC AUC differently then it must not impact ROC AUC monitorization. From this figure, one can see that AAI* reduces local ROC AUC to a random guessing level regardless of the adversarial setup, or any other attack-specific component. Similarly to the BADAcc results in Section 4.4.2.1, these findings show that BADAUC is robust to defense-aware adversaries, even though it is possible to marginally reduce TPR when performing AMI on a restricted setup.

4.4.3.2 Impact of the number of target samples

Figures 4.11 and 4.12 show that targeting an increasing number of samples does not reduce, nor increase, the detection probability of both AMI and AAI. This observation is to be expected given that impact of both attacks on ROC AUC does not present any trend as the percentage of neurons controlled by the attacker increases, according to Figure 4.13. Even though the slightly

increasing trend of ROC AUC after AMI on a standard setup explains a negligible decrease of the respective BADAUC TPR. These results show that an ambitious adversary that targets a large number of samples is equally detectable, by BADAUC, as one that targets just a few. This finding is particularly useful for AMI detection, as increasing the percentage of controlled neurons decreases FPR while maintaining perfect TPR.

4.4.3.3 Impact of model complexity

Based on Figures 4.11 and 4.12, higher or lower model complexity does not reduce AMI and AAI* detection probability, since TPR is not significantly different across distinct model complexities, nor FPR. This finding can be explained by Figure 4.13, in which no significant differences are registered between the impact on ROC AUC over models of distinct complexities. Similarly to BADAcc, these results show that a more complex model does not give an extra advantage to the adversary when BADAUC is used as the detection strategy.

4.4.3.4 Threshold dependency

As a threshold dependent detection algorithm (as explained in Section 4.2.2), BADAUC performance illustrated above was obtained assuming an optimal threshold choice per experiment. Meaning among several thresholds tested, only the one which maximized $\frac{TPR}{FPR + Missed\ attack}$ is chosen, where *Missed attack* stands for the proportion of users whose detection system did not signal any alarming ROC AUC drop. This expression represents BADAUC utility as it balances detection ability and robustness to benign ROC AUC drops.

To select the optimal threshold, one must balance the trade-off between detection ability and robustness. However, the utility function depends on the cost of each error (equivalently gain of accurate detection), which affect their relative importance on the utility function. Figures 4.14 and 4.15 show BADAUC performance on AMI and AAI* detection for varying error costs. In each side of these figures, the relative importance of each error ranges from $[0.15, 0.85]$ and the complementary relative importance is equally distributed for the rest of the utility function's components. Only standard setup attacks that control 3% of neurons over models of standard complexity are represented in these figures, as none of attack-specific components significantly impacted BADAUC performance for a considerable margin. The baseline, labelled as optimal, represents the respective results previously showed, that is when the relative importance of all metrics is balanced $\left(\frac{1}{3}\right)$. From these figures, one can see that even for varying error costs, BADAUC effectively detects both attacks, as TPR is close to 90% (if not higher) and FPR is always below 7.5%. Moreover, across this wide range (of the cost of errors), small differences were registered. Note that increasing FPR's relative importance above a certain point requires the minimization of FPR to be a priority, which means larger thresholds are better. On the other hand, increasing Missed attacks relative importance up to a certain point favours the choice of smaller thresholds, thus increasing TPR. It is worth mentioning that, for very small values of

relative importance of FPR, TPR decreases, unlike one would expect, but note that all 3 of these metrics sum up to 100%, meaning that TPR decreases due to an increase of FPR. In fact, this is supported by common practices of detection algorithms - FPR minization must have at least some relative importance, otherwise detection seems non sense. All in all, these findings show that, despite being threshold dependent, BADAUC is robust to a different choice of priorities.

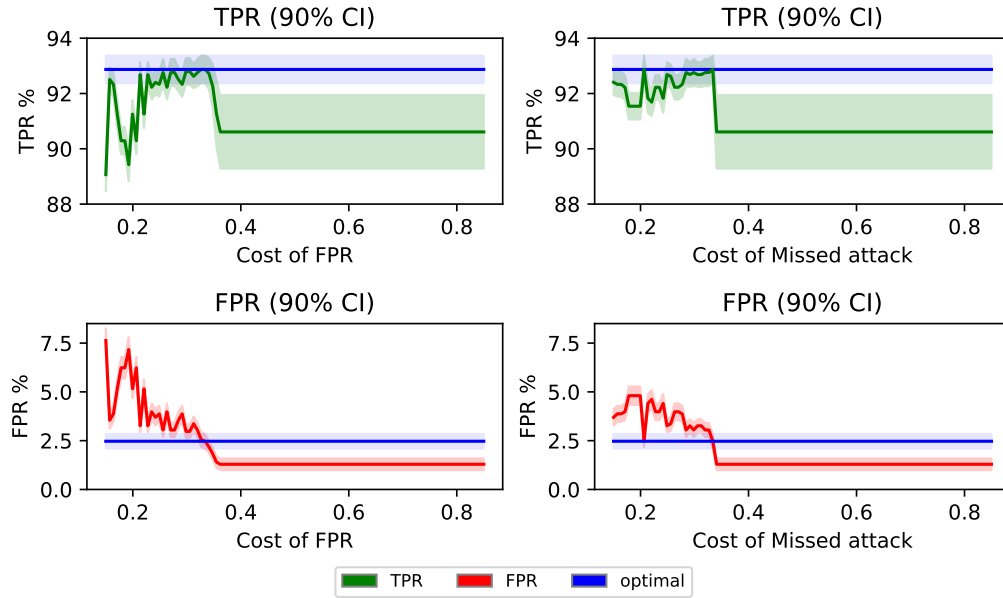


Figure 4.14: BADAUC performance for varying error costs. Detection of standard setup AMI that controls 3% of neurons over models of standard complexity.

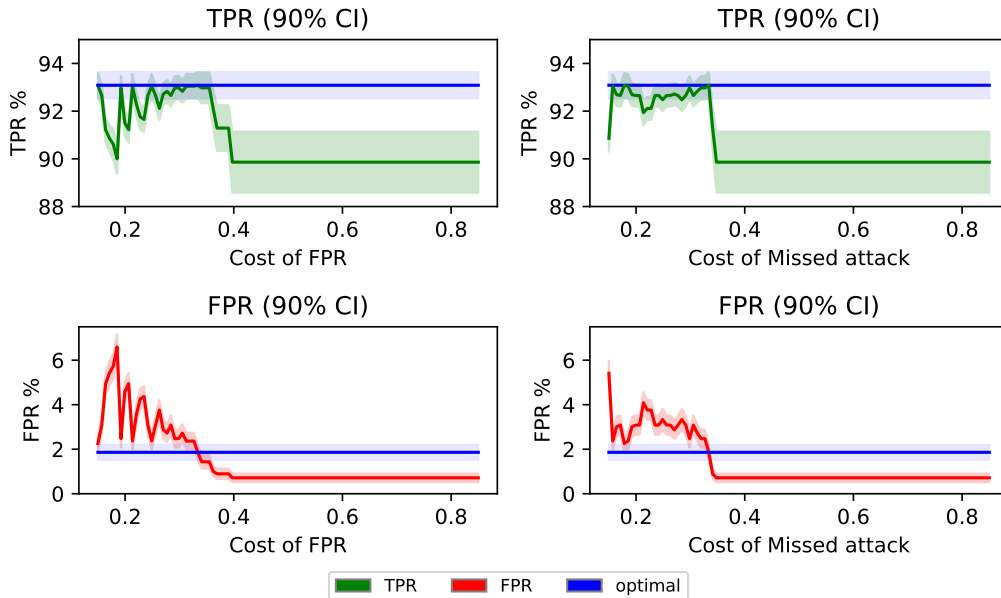


Figure 4.15: BADAUC performance for varying error costs. Detection of standard setup AAI* that controls 3% of neurons over models of standard complexity.

In a practical scenario, selecting the best threshold is a problem itself, as it may depend on

several adversarial components, as well as on the use-case. To prove the real effectiveness of BADAUC on a practical scenario, one must show that it still detects active attacks effectively for a large range of thresholds. Figure 4.16 shows BADAUC performance on AMI and AAI* detection for thresholds on standard and restricted setup attacks that control 3% of neurons over models of standard complexity. Note that the threshold represents the maximum ROC AUC (%) above which the user triggers an alarm. From this figure, one can see that BADAUC effectively detects both attacks in a wide range of thresholds. Additionally, regarding BADAUC behaviour when detecting AMI, there is a small but significant difference between TPR on a standard and restricted adversarial setup, as previously discussed in 4.4.3.1, equivalently the same difference on missed attacks. That small difference increases along with an increase of the threshold. Moreover, as also discussed in Section 4.4.3.1, AAI* detection is in fact independent of the adversarial setup regardless of the threshold. Furthermore, FPR achieves a reasonably good level soon, that is for a small threshold (3% for AMI, and 2% for AAI), meaning that, in a normal situation, only a small variability of ROC AUC is expected. In other words, in this case, allowing greater drops in ROC AUC to be expected leads to an increasing number of users being systematically exposed to these active attacks, as they are not able to detect neither of these attacks using BADAUC. Please note that FPR is independent of any adversarial setup for any fixed threshold, as explained in Section 4.4.2.4.

Varying several attack components highlights a common pattern of ineffectiveness for very small thresholds. However, the threshold above which detection becomes effective might be determined by the use-case, and the attack being detected, since FPR when detecting AAI* decreases considerably faster. Intuitively, an unstable, in terms of ROC AUC, classification problem requires a larger threshold. In this case, ROC AUC converges in less than 5 training rounds, therefore one must look for the optimal threshold in a smaller range.

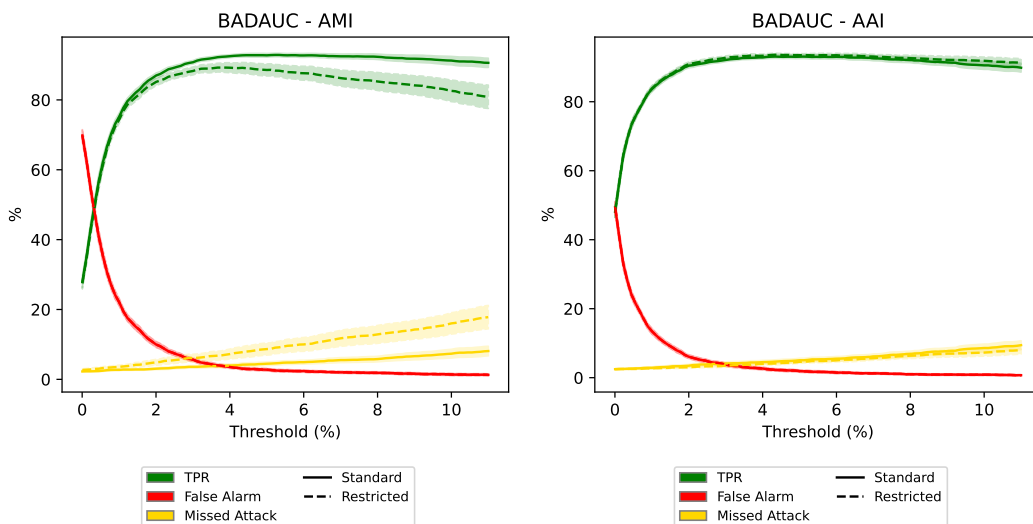


Figure 4.16: BADAUC performance for varying thresholds. Detection of standard and restricted setup AMI and AAI* that control 3% of neurons over models of standard complexity.

4.4.3.5 User dependency

Similarly to BADAcc and given that none of the attack components had a significant impact on BADAUC performance, statistically and in absolute value, one might think that local features play an important role in attack detection.

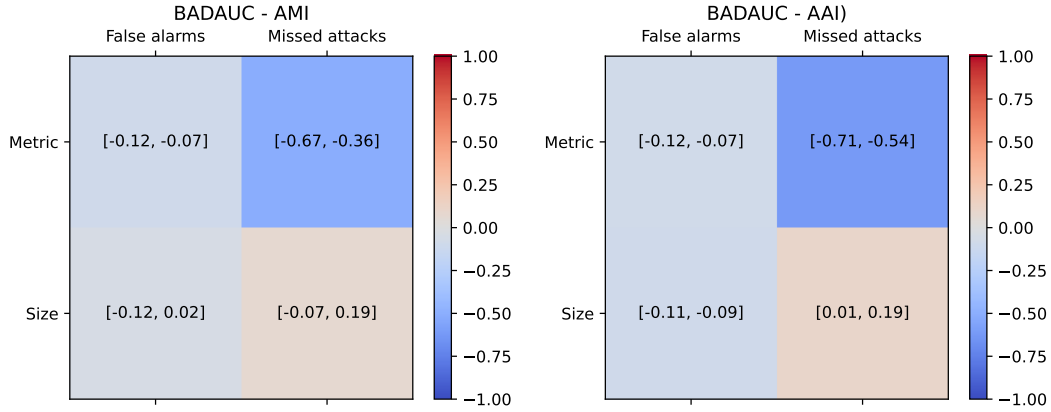


Figure 4.17: Correlation matrix of BADAUC results (of AMI and AAI* detection) with local features

Figure 4.17 illustrates the Spearman correlation between local ROC AUC and dataset size with both types of error (*False alarms* and *Missed attacks*) depending on the attack. Once again, each cell contains an interval of correlation values that summarizes the correlation matrix of each attack detection for several percentages of controlled neurons, both adversarial setups, and several model complexities. From this figure, one can see that only *Metric* plays an important role on BADAUC performance on both attacks' detection. Moreover, given that most intervals are narrow, the correlation structure does not differ when varying attack-specific features. *Metric* has a weak to strong negative correlation with *Missed attacks*, which derives directly from the fact that this strategy is distance-based, thus the higher the ROC AUC (on a benign round) the higher the probability of being detected. It is worth mentioning that correlation between *Missed attacks* with *Metric*, regarding AMI detection, is weak to moderate on a restricted setup and moderate to strong on a standard setup. Size does not play any role on BADAUC, therefore it is not expected that it has any correlation with these evaluation metrics, as observed in Figure 4.17. This correlation structure reveals that the greater the local ROC AUC (*Metric*) the lower the probability of missing an ongoing attack (*Missed attacks*).

However, none of these local features is correlated with false alarms. In fact, false alarms are mainly caused by 3 to 6 users, depending on model complexity. Users abnormally prone to false alarms when BADAcc is applied are also highly likely to commit the same error when using BADAUC instead. ROC AUC unstability explains why these users incur in false alarms much more frequently.

4.4.3.6 Summary

As one of the detection strategies that does not require unlimited trust in the defender's party, BADAUC achieves great results (TPR around 90% and FPR less than 5%), therefore being an effective defense mechanism. These results show that BADAUC is robust to defense-aware adversaries and to several attack-specific components, similarly to BADAcc but improving upon this one in terms of TPR and FPR. An in-depth study of the threshold dependency of BADAUC performance reveals that this is not a concern in practice, as long as the target model converges, given that BADAUC is effective under a wide range of thresholds and for a variety of priorities (regarding both types of error). Similarly to BADAcc, both types of error (false alarms and missed attacks) depend on the specific user. The same users that are abnormally prone to false alarms in BADAcc are still in the same situation when BADAUC is used instead, thus these cannot use the system, as they systematically leave it on a benign training round. On the other hand, some users are constantly exposed to both attacks (consequence of missed attacks) regardless of the black-box attack detection strategy they apply. These findings motivate the need to reduce the cost of each error, and ideally also reducing their occurrence.

4.4.4 White-box Attack Detection and Mitigation Analysis

In this section, the reader can find WADM^{*} experimental results. The impact of several components on attack detection performance are presented. By the end of this section, the reader must be convinced that white-box attack detection and mitigation effectively detects malicious neurons and mitigates the ongoing attack. This result section proves that malicious neurons weights distributions are significantly different from the distribution of a benign neuron, which can be easily detected by a distance-based approach. By reducing user dependency, as well as model utility loss, WADM^{*} outperforms black-box strategies.

In the following subsections, the reader can find WADM^{*} experimental results for both AMI and AAI^{*} in which the effect of different components on attack detection is evaluated in sections 4.4.4.1, 4.4.4.2, and 4.4.4.3, respectively. As a threshold dependent strategy, one must evaluate its effectiveness under different threshold choices, which is explored in Section 4.4.4.5. Finally, since clients using WADM^{*} not only detect but mitigate (Section 4.3) both attacks without leaving the system, it is relevant to evaluate the impact that WADM^{*} mitigation phase has on a model's ability to recovery from active attacks, which is explored in Section 4.4.4.6.

Similarly to the empirical study of black-box detection strategies, and given that the number of target samples and model complexity significantly impacted attack performance, one must evaluate their impact on WADM^{*} detection and mitigation performance. Moreover, a robust defense strategy is one that is still effective under more challenging setups, in which the adversary might be willing to exchange attack performance for a reduced probability of being detected. Figures 4.18 and 4.19 illustrate TPR and FPR evolution when detecting AMI and AAI^{*} with an increase of the percentage of neurons controlled by the attacker, for each model complexity,

and for 2 different setups. Note that the y-axis limits are equal to ease comparisons. From these figures, one can see that WADM^{*} effectively mitigates both attacks, as TPR tends to be close to 100% (only slightly lower when detecting AAI^{*} in some cases) and FPR is always 0, meaning not a single benign neuron was flagged as malicious, in the training round in which attacks were performed. Even though a fair comparison cannot be made as the detection domain is different, WADM^{*} improves upon black-box detection rate, as it detects close to 100% of malicious neurons, therefore detecting close to 100% of attacks, whereas BADAUC detected close to 90% and BADAcc close to 80%.

A small threshold (in this case 0.1556 for both attacks) is the key to reach such a good performance. However, this small threshold could raise some false alarms, which does not happen in this case. This observation can be explained by the fact that these target models, by the time attacks occur and this strategy is applied over neurons of the second FC layer, have already converged, therefore little variability on weights distribution must be expected on a benign situation. This insight is further explored in Section 4.4.4.5. Since FPR does not vary across any combination of attack-specific components and adversarial setup, the following sections (4.4.4.1, 4.4.4.2, and 4.4.4.3) comment only on WADM^{*} TPR.

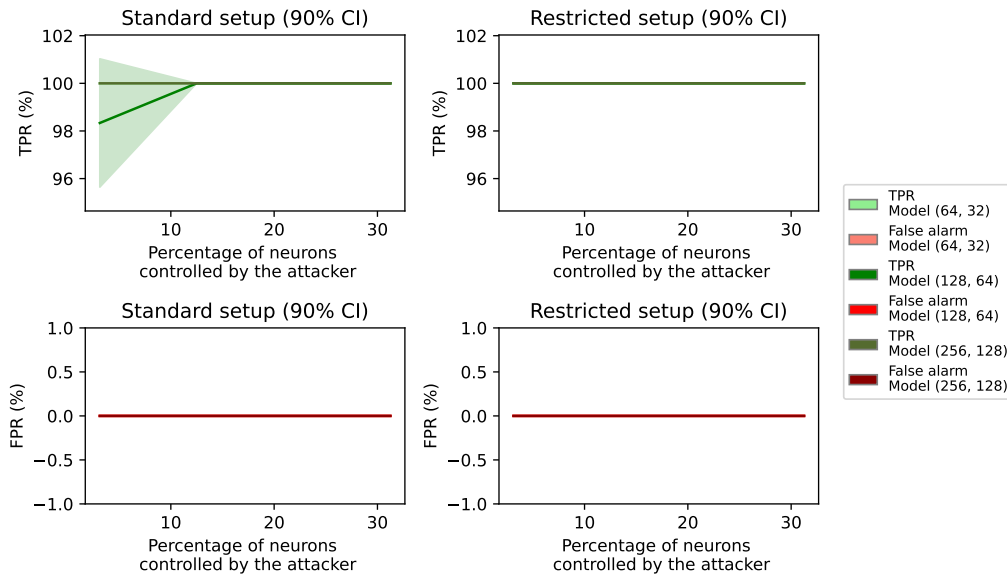


Figure 4.18: WADM^{*} performance on detecting AMI conducted on federated models of multiple complexities, for distinct adversarial setups and for an increasing percentage of neurons controlled by the attacker.

4.4.4.1 Impact of restricted setup

Based on Figures 4.18 and 4.19, an adversary willing to trade AMI performance for a reduced probability of being detected fails at this goal, since detection results on a standard setup are not significantly different from those of a restricted setup. Regarding AAI^{*} detection, WADM^{*} is also effective, yet it presents little variability when only the adversarial setup varies. Since AMI is

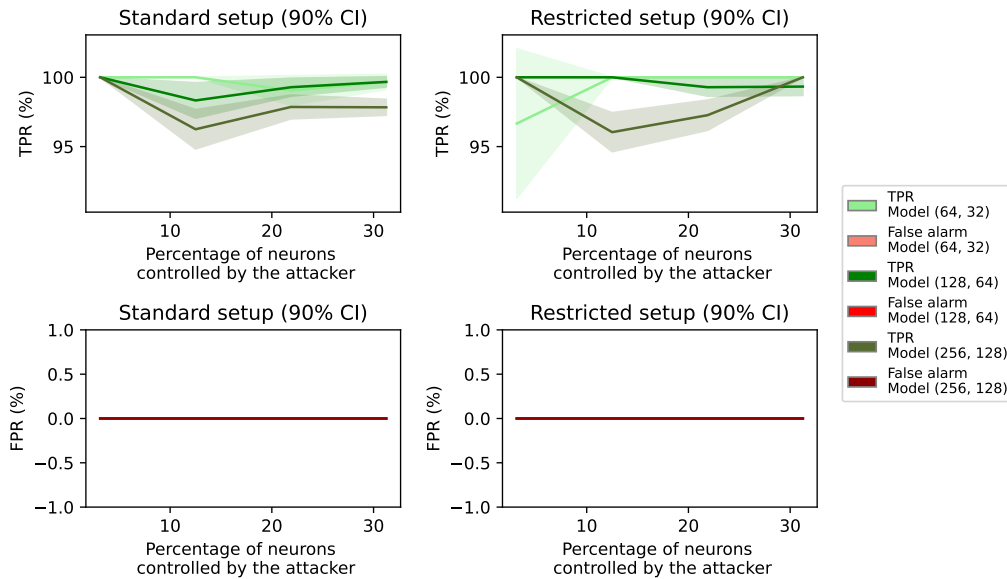


Figure 4.19: WADM* performance on detecting AAI* conducted on federated models of multiple complexities, for distinct adversarial setups and for an increasing percentage of neurons controlled by the attacker.

always effective on a standard setup, the weights distribution must have changed significantly to linearly separate (after passing through the first layer) the target sample from the rest. Therefore, imposing a small threshold is sufficient to capture those changes. What is most surprising is that, on a restricted setup, some combinations of attack-specific components lead to ineffective attacks (given the large FPR on Figure 4.4), yet WADM* is still able to detect those malicious neurons, even if they do not overfit the target sample properly. Analogously, some combinations of attack-specific components lead to ineffective AAI* attacks, particularly on a restricted setup, yet WADM* is, in most cases, equally able to detect neurons dedicated to AAI* regardless of the adversarial setup. These results suggest that WADM* detection does not seem to depend on attack performance but on any malicious interference in neuron’s weights.

4.4.4.2 Impact of the number of target samples

According to Figures 4.18 and 4.19, attacks that target an increasing number of samples are not statistically more or less easier to detect, as neither TPR or FPR vary significantly across this range, for both attacks. Which means that any additional neuron that is maliciously modified is detected. Even though only AMI performance improves as the percentage of neurons controlled by the attacker increases, these results show that an ambitious adversary that increases the number of target samples is equally detectable by WADM*, as with other detection strategies.

4.4.4.3 Impact of model complexity

According to Figures 4.18 and 4.19, higher or lower model complexity does not reduce attack detection probability, as most of these results are not significantly different across distinct model complexities. Albeit there is a small, but sometimes statistically significant difference, particularly on more complex models, on WADM* TPR when detecting neurons dedicated to AAI*, yet there is not a clear trend. This can be explained by the fact the same percentage of controlled neurons on a higher complexity leads to more neurons being dedicated to the malicious task, which may leave room for more errors (missed neurons).

However, it may sound surprising that model complexity does not affect malicious neuron detection probability, as different model complexities imply different neuron's weights distribution. That is, in a model of standard complexity (128, 64) each neuron's weights distribution is described by 128 bins, analogously 64 and 256 bins describe neuron's weights distribution on more simpler and more complex models, respectively. Additionally, the fact that FPR does not change across distinct model complexities may sound surprising for the same reasons. In fact, to improve robustness only 5 equal width bins are used to calculate KL-Divergence (to improve WADM* robustness). Thus, despite the original weights distribution being described by an increasing number of bins, only a reduced representation of 5 bins is used to calculate KL-Divergence regardless of model complexity, which reduces the potential impact of increasing the dimension of each neuron's weights distribution.

4.4.4.4 Attack mitigation

The greater advantage of WADM* over black-box detection strategies is that when a client detects a malicious neuron it does not need to leave the system in order to mitigate the ongoing attack. Instead, it can replace that neuron's weights distribution for a benign version, namely the previous set of weights, thus reducing model utility loss. Even though black-box strategies could also require the client to replace previously saved benign model weights, instead of leaving the training. However, black-box strategies are mostly useful when the defender/client has a restricted model access, for example in closed source FL systems. To evaluate the extent to which WADM* effectively mitigates both attacks, one must prove that it reduces attack performance. Figures 4.20 and 4.21 illustrate attack performance after the mitigation of both AMI and AAI* over federated models of multiple complexities, for distinct adversarial setups and for an increasing percentage of neurons controlled by the attacker. The baseline of comparison, that is attack performance before the mitigation, is presented in Figures 4.4 and 4.5.

From Figure 4.20, one can see that WADM* mitigation phase effectively reduces AMI performance. Despite the fact that TPR is still not statistically different from 100% on almost all experiments, FPR increases significantly, ranging from 30% to 80%. Moreover, WADM* mitigation phase did not remove the impact of attack-specific components on attack performance. That is, the percentage of neurons controlled by the attacker still impacts FPR (although

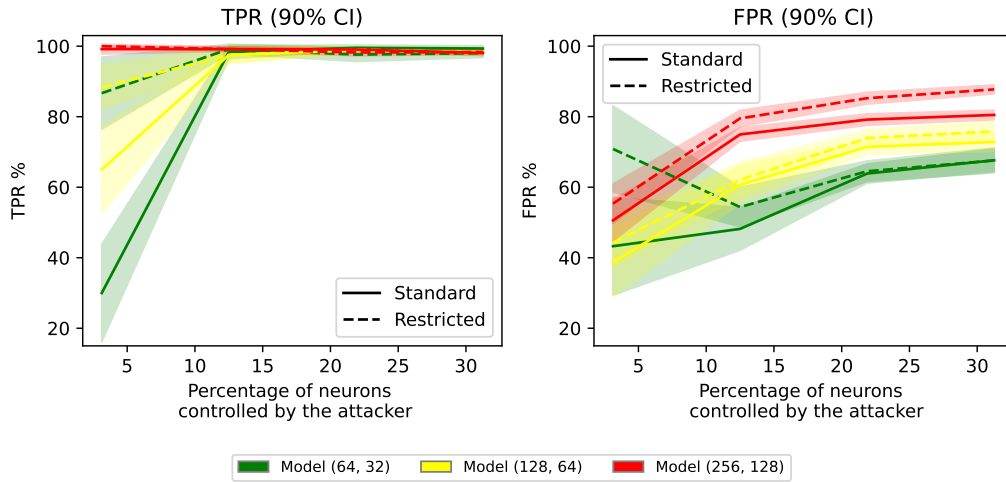


Figure 4.20: AMI performance, after applying WADM^{*}, on federated models of multiple complexities, for distinct adversarial setups and for an increasing percentage of neurons controlled by the attacker.

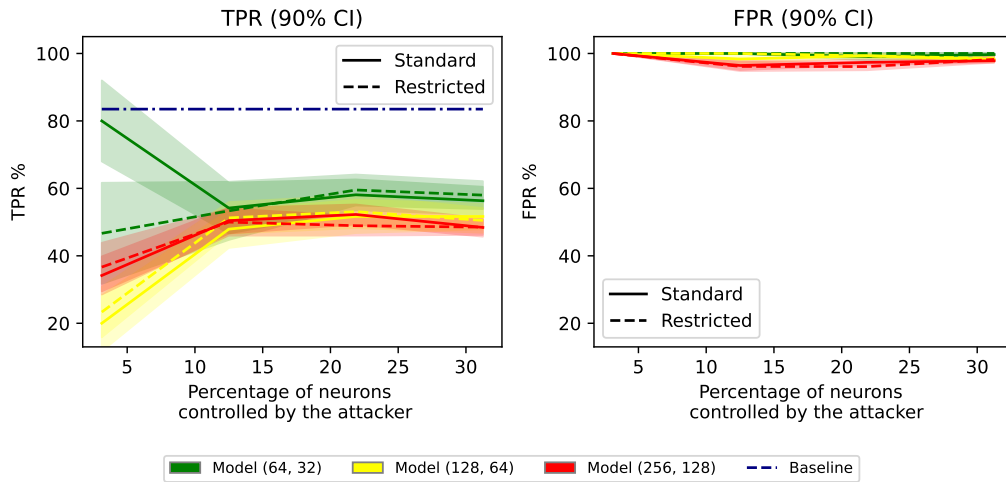


Figure 4.21: AAI^{*} performance, after applying WADM^{*}, on federated models of multiple complexities, for distinct adversarial setups and for an increasing percentage of neurons controlled by the attacker.

differently), and, only after mitigation, it can also have a significant impact on TPR (although it converges shortly). Note that TPR may be more prone to drastic variations, in terms of rate, since its domain, that is the number of target samples, is much smaller than FPR domain (with thousands of samples). For a small percentage of controlled neurons, TPR decreases, namely for standard and simpler model complexities, and FPR is around 40% to 50% on average for most of those experiments. Increasing the percentage of controlled neurons increases FPR, but only up to a point. Increasing model complexity leads to an increase of FPR, contrarily to the effect prior to WADM^{*} mitigation phase. Finally, the adversarial setup (standard or restricted) has a minor and not statistically significant effect on FPR (on most cases), which is the opposite of its

effect on attack performance prior to WADM^{*} mitigation phase, as shown in Figures 4.4 and 4.5.

From Figure 4.21, one can see that WADM^{*} mitigation phase also effectively reduces AAI^{*} performance. TPR is reduced to a random guessing level, considerably lower than the baseline (blue dashdotted line) and FPR is not significantly different from 100% on all experiments. This is a completely ineffective attack, even if the baseline was 50% (the lowest one can have). Even though single-target attacks on a standard setup (meaning 3% of controlled neurons over simpler models) achieve significantly greater TPR, this attack is still ineffective due to extremely high FPR. Note that every attack-specific component's impact on attack performance was removed, not even adversarial setup impacts the mitigation.

These experiments show that, other than almost perfect malicious neuron detection, WADM^{*} is robust as it is able to mitigate both attacks under several combinations of attack-specific components and adversarial setup. Effective mitigation is a crucial part of this defense strategy, as solely detecting the malicious neuron without mitigating the attack would not prevent the adversary from being successful, as clients do not leave the system.

4.4.4.5 False alarms minimization

Previous results were obtained with a very small threshold in order to maximize TPR, knowing that FPR was totally under control, as the model converged to a stable state. However, in a practical scenario, the defender does not know which training round is affected by an ongoing attack. Thus, it is crucial to evaluate WADM^{*} performance on the first training rounds, in which the model is still converging. Intuitively, one must expect to see an increase of false alarms, at least for small thresholds.

Figure 4.22 illustrates false alarms evolution of WADM^{*} applied on the first training rounds of models of multiple complexities. Different models are used for different attacks, thus one must distinguish these results accordingly. From this figure, one can see that FPR decreases along with model convergence, as expected. Naturally, the higher the threshold the lower the number of false alarms. Particularly, setting the threshold to 0.4722, in this case and for models of standard complexity used for AMI, causes a very small FPR from the beginning. That same level of FPR is achieved with an even lower threshold, in this case, for models of standard complexity used for AAI^{*} (0.2611). Furthermore, as model complexity increases so does the speed with which FPR is reduced to a neglectable level, as it can be observed by the progressive overlap of curves of several thresholds, which happens earlier for higher model complexities. Note that model type (either with ReLU or ELU activation functions over which AMI and AAI^{*} are applied, respectively) does not impact these findings. Additionally, WADM^{*} incurs in a significantly lower number of false alarms when the model is used for AAI^{*} comparing to models used for AMI. This observation might be explained by the fact that the former type of models (used for AAI^{*}) converged quicker than the latter (used for AMI), which was seen during the experimental phase of this work.

Reducing false alarms along the training rounds is imperative to reduce WADM^{*} potential

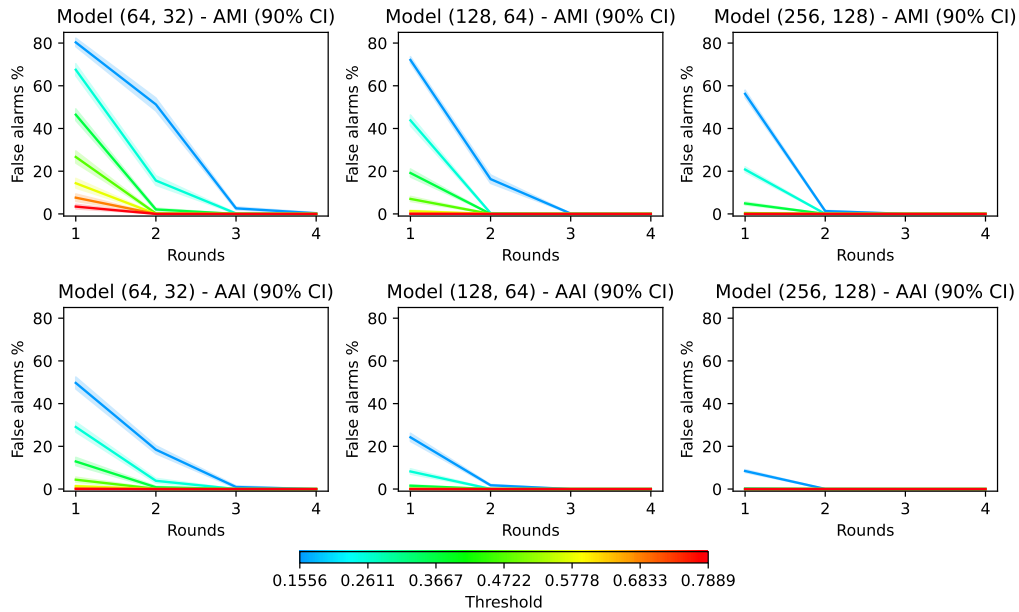


Figure 4.22: WADM* false alarms (FPR) evolution along the first benign training rounds of models, used for AMI and AAI*, of multiple complexities.

impact on model’s convergence, as the defender would be constantly rewinding on the benign progress of those neurons. Even though FPR’s cost from applying WADM* is much lower when compared to black-box strategies, as these force the client to leave the system. Figures 4.23 and 4.24 illustrate WADM* detection of both attacks under FPR minimization on early training rounds. Note that FPR is omitted as increasing the threshold would only decrease FPR, but it was already 0 for all experiments. WADM* uses 0.4722 as the threshold on AMI detection and 0.2611 on AAI* detection. The choice of these thresholds balanced FPR minimization on early training rounds, as shown in Figure 4.22, and detection performance.

From these figures, one can see that even under FPR minimization, WADM* effectively detects maliciously modified neurons from both attacks. However, AMI detection results show that an adversary willing to trade attack performance for a reduced detection probability succeeds at this goal depending on model complexity and percentage of controlled neurons. For a small number of targets and a model of standard complexity, an attacker on a restricted setup is able to reduce TPR to less than 60% on average. Malicious neuron detection is even harder on simpler models as TPR ranges from 40% to 80% on average depending on the number of target samples. Although, attacking a simpler model on a restricted setup significantly reduces attack performance, for a large margin as shown in Figure 4.4. AAI* detection results also reveal statistically significant differences (comparing to previously presented detection results). Particularly when the model is the most complex, for which TPR can be close to 80%, but one must keep in mind that these attacks are no longer effective (due to the fact that TPR is lower than the baseline). However, an AAI* adversary willing to trade attack performance for a reduced detection probability fails at this goal, since there are no significant differences when both adversarial setups are compared.

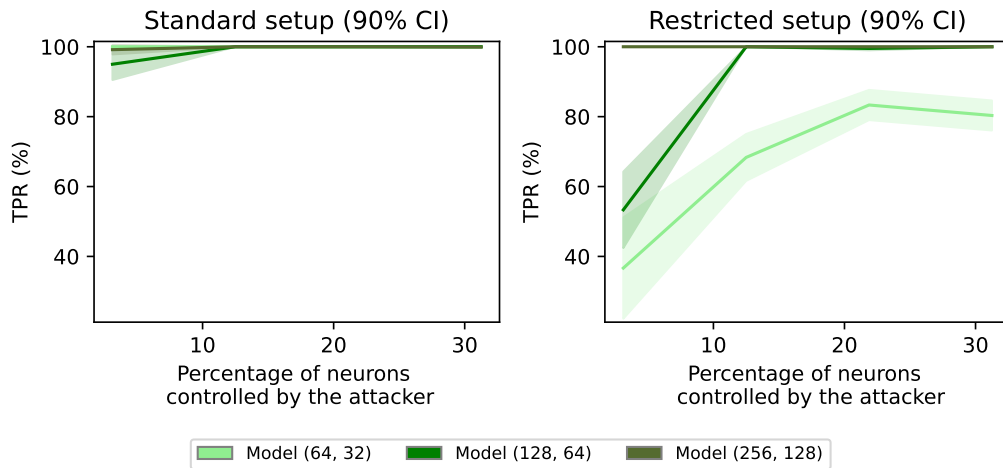


Figure 4.23: WADM* performance on detecting AMI over federated models of multiple complexities, for distinct adversarial setups and for an increasing percentage of neurons controlled by the attacker. The threshold was chosen taking into account FPR minimization (0.4722).

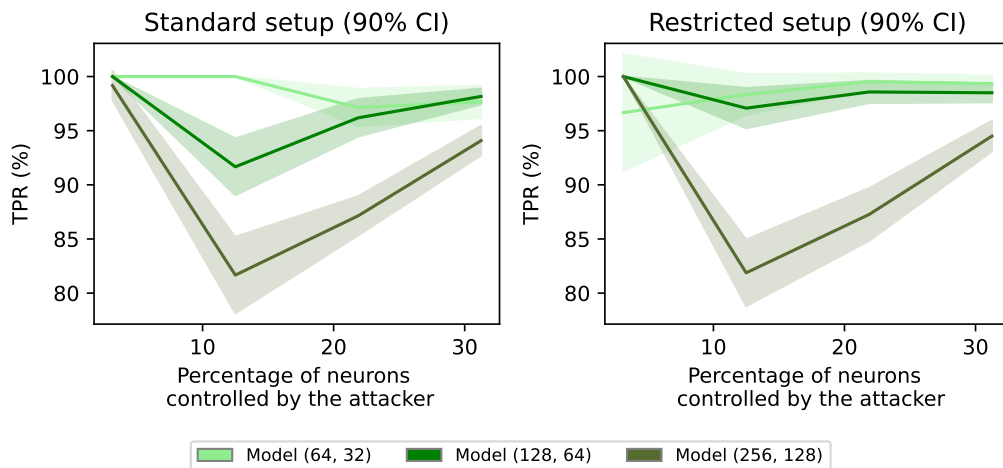


Figure 4.24: WADM* performance on detecting AAI* over federated models of multiple complexities, for distinct adversarial setups and for an increasing percentage of neurons controlled by the attacker. The threshold was chosen taking into account FPR minimization (0.2611).

Despite the fact that the detection probability dropped in some experiments, WADM* mitigation phase is still effective, as attack performance is reduced to a point such that the adversary no longer has a guarantee that conducting any of these attacks yields a small FPR. These conclusions can be taken from Figures 4.25 and 4.26 which show attack performance after WADM* mitigation under FPR minimization. The impact of attack-specific components on attack performance after mitigation remain the same as in Section 4.4.4.4. However, it may sound surprising that even with such small detection probability for some restricted AMI experiments, WADM* mitigation is still effective. This can be explained by the fact that, for most of those experiments, AMI was not successful, meaning FPR was already high, prior to any mitigation (see Figure 4.4). Same conclusions can be drawn for AAI* detection results under FPR minimization,

as they are not statistically significant from those obtained with an optimal threshold (in Figure 4.21).

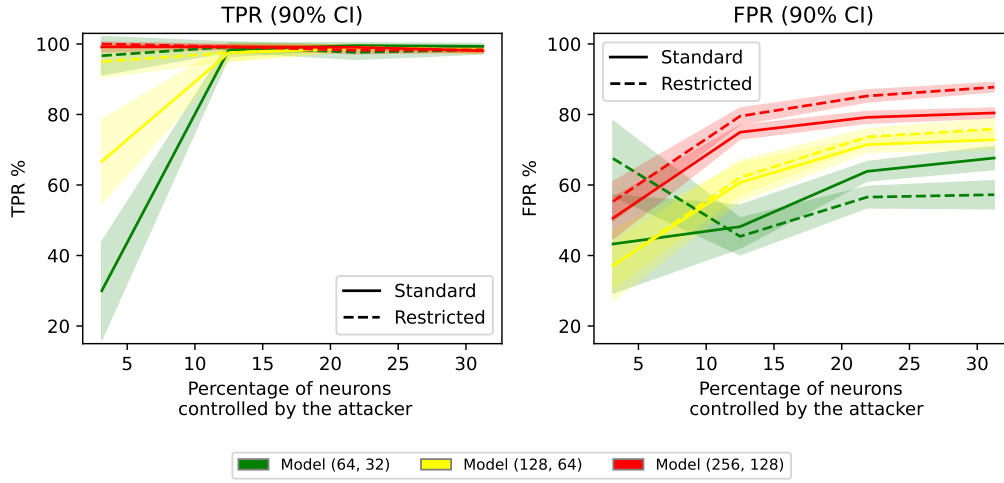


Figure 4.25: WADM* performance on mitigating AMI over federated models of multiple complexities, for distinct adversarial setups and for an increasing percentage of neurons controlled by the attacker. The threshold was chosen taking into account FPR minimization (0.4722)

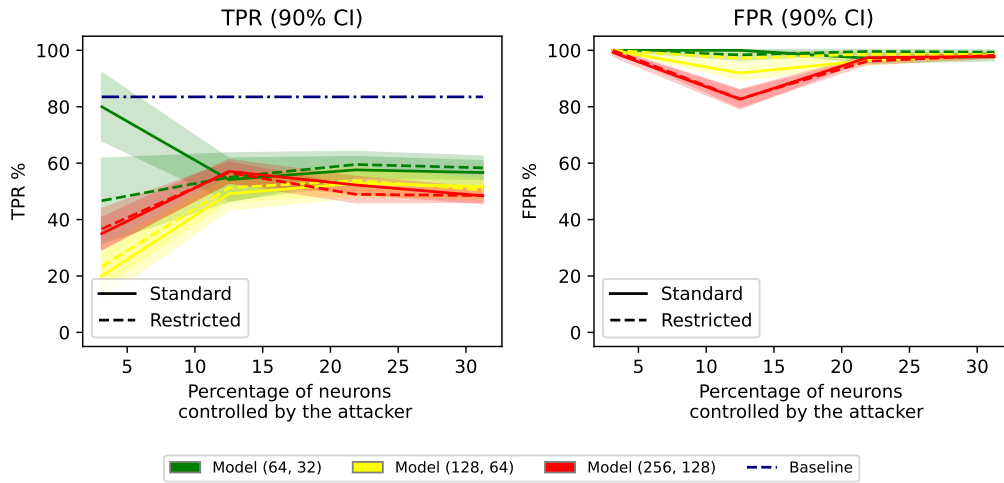


Figure 4.26: WADM* performance on mitigating AAI* over federated models of multiple complexities, for distinct adversarial setups and for an increasing percentage of neurons controlled by the attacker. The threshold was chosen taking into account FPR minimization (0.2611)

4.4.4.6 Impact on model's recovery from active attacks

To evaluate the extent to which WADM* model utility loss is reduced, one must evaluate the target model's ability to recover from AMI and AAI* after WADM* mitigation phase. Similarly to Section 3.2.5, the ability to recover is measured with ROC AUC. Note that reducing model utility loss is only possible because the target model is able to recover after both attacks, as

shown in Section 3.2.5. In fact, WADM^{*} mitigation is evaluated in light of these results (Section 3.2.5) to see if WADM^{*} mitigation phase speeds, or improves, the recovery process. Figures 4.27, and 4.28 illustrate ROC AUC evolution over 5 training rounds for models of distinct complexities, for several percentages of controlled neurons, on the train and test set. For comparison purposes, the y-axis is evenly set, per row. The range of colors indicates the percentage of controlled neurons, and the dashed baseline stands for the initial target model average ROC AUC (right before the attack). As expected, ROC AUC (on the training set) of a maliciously modified target model converges, or approaches, to the initial target model performance as the number of training rounds increases and that does not change when WADM^{*} mitigation phase is applied. Note that results of recovery from both attacks on a standard setup after WADM^{*} mitigation phase are omitted as no significant differences were registered.

Figure 4.27 illustrates the convergence of the target model that was modified by an adversary conducting AMI (on a restricted setup) and after WADM^{*} mitigation. From this figure, one can see that it is possible to fully recover from this attack in 2 training rounds on the training set, but not on the test set even after 5 rounds. Although the model is still not able to recover on the test set, ROC AUC drop is smaller (comparing to recovery from AMI on a standard setup without any defense mechanism from Section 3.2.5). Even though ROC AUC drop on the test set is still statistically significant. Moreover, the effect of attack-specific components on the recovery ability remains the same (comparing to results from Section 3.2.5). Note that for a very small percentage of controlled neurons on simpler models, the speed of convergence is even higher. That might be explained by the fact that those attacks were not effective, as FPR was 50% on average, according to Figure 4.4.

In its turn, Figure 4.28 illustrates the convergence of the target model that was modified by an adversary conducting AAI^{*} (on a restricted setup) and after WADM^{*} mitigation. From this figure, one can see that none of these experiments revealed significant differences comparing to the target model's ability to recover from AAI^{*} on a standard setup, according what was analysed in Section 3.2.5 in which no defense mechanism was applied. Although there was a slight improvement on more complex models, but the respective attacks are not effective (see Figure 4.5). Naturally, the effect of attack-specific components on recovery ability remains the same.

These findings reveal that the improvement, mainly on AMI recover ability, is due to the adversarial setup (which aimed to reduce the influence of attacks on the target model) and not due to WADM^{*} mitigation phase. Despite this observation, WADM^{*} is still able to reduce model utility loss, as its mitigation phase allows for the target model to recover from active attacks, therefore recovering the system's utility.

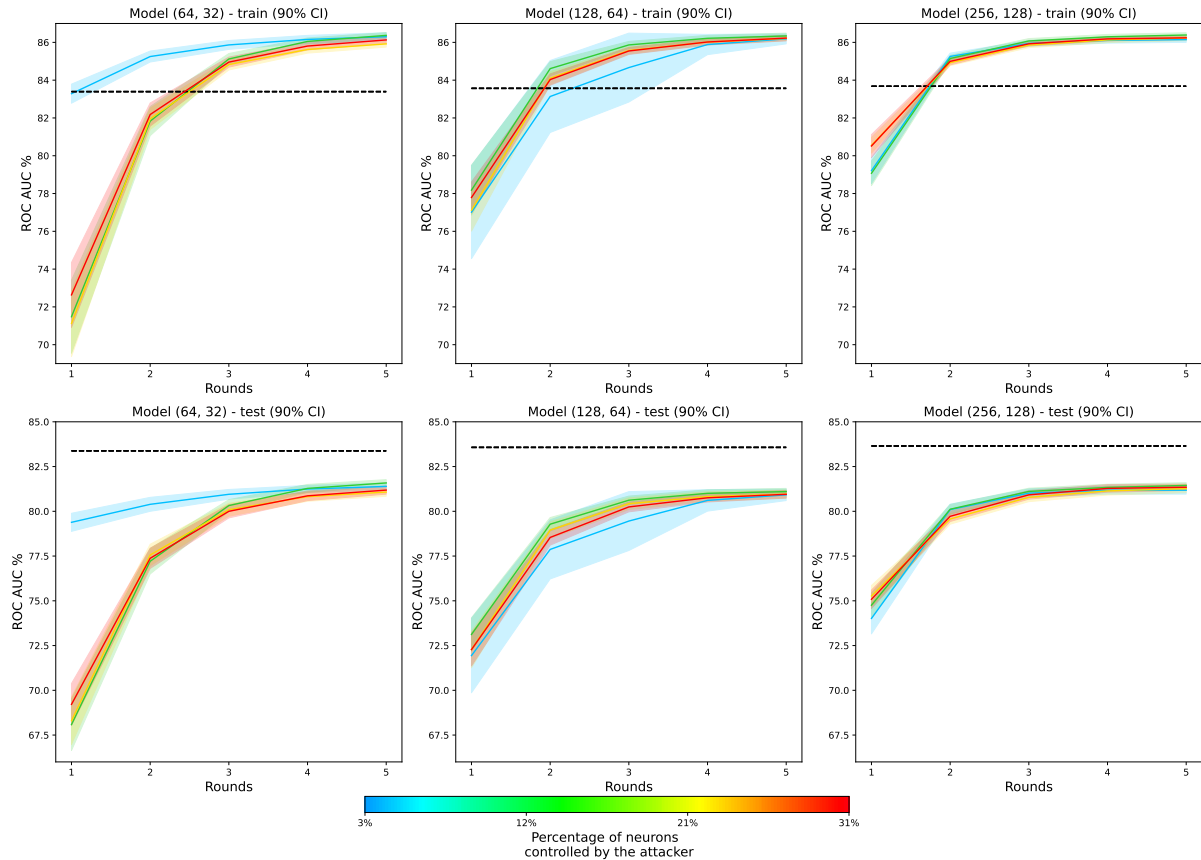


Figure 4.27: Recovery (ROC AUC evolution) from multi-target AMI (restricted setup) on train and test set of models with distinct complexities over 5 training rounds.

4.5 Discussion and Summary

In this chapter, black-box attack detection algorithms were proposed to detect AMI and AAI*, showcasing that these attacks are easily detected by an efficient monitorization of evaluation metrics on the training set, with the ROC AUC based strategy (BADAUC) the best among these. BADAUC detects around 90% of both active attacks, whereas BADAcc detects around 80%. None of the attack-specific components (percentage of controlled neurons and model complexity) impacted BADAcc and BADAUC ability to detect both AMI and AAI* significantly, even though these components may impact attack performance significantly. Furthermore, if the adversary is aware of a potential defense strategy, then it might be tempted to trade attack performance for a reduced probability of being detected. However, any of these attempts fail at significantly (statistically and in absolute value) reducing attack detection probability, meaning that a slightly less successful attack is equally detectable. Given that these strategies rely on an abnormal drop of different evaluation metrics, these findings suggest that these active attacks affect model performance regardless of their effectiveness.

Despite having a considerably high attack detection rate (TPR), BADAcc results in 7% to 12% of false alarms. Once again, BADAUC outperforms BADAcc by resulting in less than 5% of

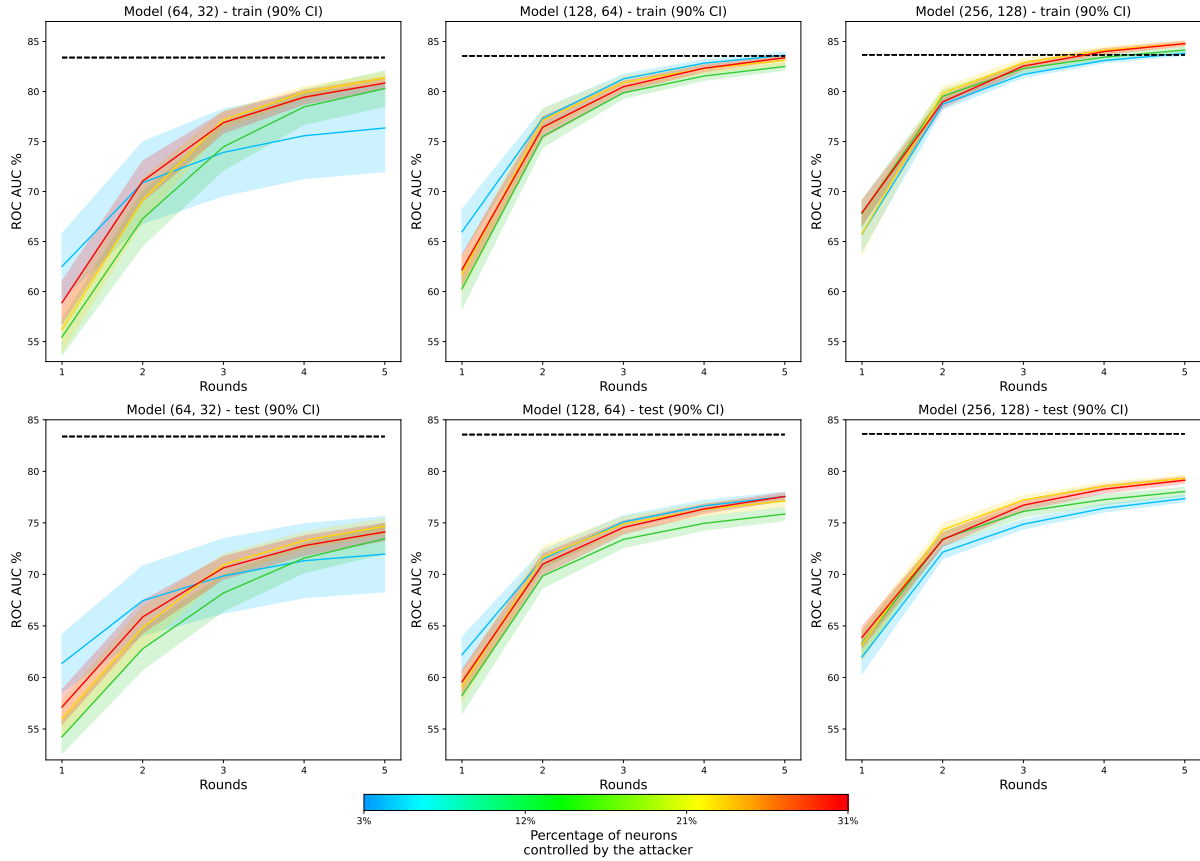


Figure 4.28: Recovery (ROC AUC evolution) from multi-target AAI* (restricted setup) on train and test set of models with distinct complexities over 5 training rounds.

false alarms, when an optimal threshold is used. These results show that when black-box attack detection mechanisms are applied, some users may systematically be forced to leave the system on a benign situation (consequence of false alarms), while others may constantly be exposed to both attacks (consequence of missed attacks) even after applying a defense mechanism.

However, other than maximum model utility loss, as the client must leave the system after detecting an ongoing attack, black-box attack detection strategies have another major limitation which is attribution. These strategies are based on well-known anomaly/drift detection algorithms that are currently applied on predictive systems to deal with the dynamic nature of processes behind data generation. In this chapter, any abnormal drop in model performance was attributed to an active privacy attack, however, in a real-world scenario, that may not be true. These strategies would most likely require additional steps to distinguish benign model performance reduction from a malign one.

White-box attack detection and mitigation strategy outperformed the black-box strategies, even though one cannot make a direct comparison as the domain of evaluation is composed of malicious neurons, not active attacks as a whole. Similarly, none of the attack-specific components affects WADM* performance significantly, not even a defense-aware strategy is able to trade attack performance for reduced probability of being detected. Besides significantly reducing

false alarms, the major distinction factor is that WADM^{*} is able to mitigate AMI and AAI^{*} while reducing model utility loss. By switching malicious neurons weights by the previous set of weights, a client is able to reduce attack performance to the point of it simply being a random guess. However, this mitigation phase does not improve the target model’s ability to recover from active attacks.

As detection strategies, all of them must reduce false alarms, despite their cost being considerably lower in the latter case, since WADM^{*} does not require the client to leave the system. False alarm minimization may depend on a preset threshold and/or user-specific characteristics, namely local training dataset size and performance prior to the attack. False alarms caused by black-box strategies are mainly due to user-specific characteristics, whereas false alarms caused by WADM^{*} are due to the threshold choice. Therefore one must tune the threshold in order to balance FPR and TPR. Even under FPR minimization on early training rounds, WADM^{*} is still able to effectively detect malicious neurons used for both AMI and AAI^{*}, at least when their performance is better than random guessing. Although tuning this threshold may depend on the use-case, as well as on the target model’s architecture. Furthermore, as training evolves so does FPR to the point of it being zero, or within a negligible range, after a few rounds.

Despite being considerably different, both AMI and AAI^{*} are successfully detected by any of these defense strategies. In particular, BADAUC presents no statistically significant differences between TPR on both cases, whereas BADAcc may present some significantly different TPR values but for a small margin. On average, AAI^{*} is easier to detect. WADM^{*} perfectly detects both attacks significantly improving upon previous strategies, however, under FPR minimization, WADM^{*} may present statistically significant differences between AMI and AAI^{*} detection, namely on a restricted setup. More importantly, these defenses may be considered attack agnostic (a statement to be validated as future work) as they are successful at detecting and mitigating several active attacks.

Measuring the real utility of defenses is an evaluation step of major importance, as to defend against real attacks, one must reduce their real threat. That is, the real utility of defenses is the ability of mitigating an attack evaluated under a practical scenario, as in Section 3.2.6. Keeping the same setup, both attacks’ real threat was measured after WADM^{*} mitigation phase. Note that, AMI is only effective when a non random target sample selection is performed, so one must only mitigate an effective attack. Moreover, for single-target attacks on a standard adversarial setup, malicious neuron detection rate was perfect (100%), therefore these experiments assume a perfect detection. Results after applying WADM^{*} mitigation phase show that an adversary conducting AMI on a practical scenario would reach restricted ROC AUC in between [45.78%, 54.49%], with 90% of confidence. On their turn, an adversary conducting AAI^{*} would reach restricted ROC AUC in between [49.73%, 55.08%] and MIA2AIA in between [48.81%, 52.86%]², with

²As stated in the beginning of this chapter, MIA2AIA detection is equivalent to AMI detection for paired malicious neurons. This result shows that WADM^{*} effectively mitigates MIA2AIA, as well as AMI for the same reasons.

90% of confidence. Thus, WADM^{*} reduces the real threat of both attacks to random guessing, showcasing its effectiveness on a practical scenario.

All in all, these defense strategies effectively detect both active attacks and successfully mitigate them. Among all these strategies, WADM^{*} outperforms the rest while overcoming a major state-of-the-art limitation - reduced model utility loss. Notably, none of the attack-specific components impacted detection rate significantly, therefore these strategies are robust to any attempt to evade detection.

Chapter 5

Conclusion

Powered by sophisticated machine learning models driven by data, Artificial Intelligence (AI) leads a technological paradigm shift, that raises concerns about data privacy. These are especially pertinent in centralized learning, whereby all data is held by a single entity. However, privacy concerns also arise from distributed learning mechanisms. Hence, this thesis delves into data privacy issues in Federated Learning (FL), a distributed learning mechanism in which decentralized data is used to collaboratively train Machine Learning (ML) models. However, FL systems are vulnerable to several types of privacy attacks, from both internal and external adversaries.

Privacy attacks, such as Membership Inference Attacks (MIAs) and Attribute Inference Attacks (AIAs), try to extract information that was not intended to be shared. A MIA aims to infer whether a given sample was used in the training process, whereas an AIA aims to infer the sensitive attribute value of a training sample. These raise severe privacy risks, take for example the case of clinical records used to train ML models to assist with diagnoses or sensitive data from social networks used for personalized advertisements. An extensive review of prior work on privacy attacks revealed that papers tend to focus on attacks, leaving effective defense mechanisms behind. Moreover, state-of-the-art attacks effectively extracted information, namely membership, of extremely overfitted models. This suggests that well-generalized models were not vulnerable to those attacks, as these mainly exploited overfitting by expecting that the target model would behave differently according to the membership status of the target sample. Furthermore, past research primarily focused on passive attacks or unrealistic scenarios (such as access to individual gradient updates), leaving a gap in understanding and defending against active attacks and more realistic scenarios. An extensive review of prior work also revealed that privacy attacks have been inadequately measured (e.g., using accuracy to measure MIA performance). Moreover, the connection between different privacy attacks is another under-explored research topic, yet critical for a better understanding about the exposure of ML models to these vulnerabilities and how to mitigate them.

Defenses are even less explored, mainly aiming to mitigate passive attacks (that do not interfere in the learning process) by improving the generalization ability, such as regularization

techniques. Some of the most effective defenses led to a high model utility loss when applied, such as differential privacy. These defenses are all preventive, meaning that the learning process is privacy-preserving by default and, in some cases, the task of reducing the advantage of a potential adversary leads the whole training process, leaving the main task to the background. The focus on preventive defenses leaves a gap on research about reactive ones that only mitigate attacks that were detected, such as privacy attack detection.

5.1 Active Attacks

To cover these limitations, this research proved that well-generalized FL models are vulnerable to powerful active membership and attribute inference attacks, which encode information on selected neurons to make inference based on aggregated gradients sent by clients on the following training round. The fact that well-generalized models are vulnerable to these novel active attacks challenges part of the prior knowledge about MIAs, and AIAs, in the sense that it shows that overfitting is not a necessary condition for a model to be vulnerable (as also shown by Yeom et al. [62]). For that goal, results from a novel Active Membership Inference (AMI), showcased that well-generalized models carry a false sense of security, contrasting to state-of-the-art passive MIAs whose performance was not significantly different than random guessing. Furthermore, a novel Active Attribute Inference (AAI) was developed to increase model leakage regarding the sensitive attribute value of training samples. Comparing to powerful state-of-the-art approaches, Attribute Inference based on Membership Inference (MIA2AIA), which are MIA-based attacks, AAI* was effective, outperforming the baseline especially in terms of increasing model leakage on a practical scenario. Particularly, to adequately measure attack performance, a theoretical (based on neuron activation) and practical (based on gradient uncertainty) evaluation methodologies were applied. While the practical one aimed to account for the effect of false positives, an improvement of the theoretical evaluation is expected to lead to an improvement of the advantage of the adversary on a practical scenario. This practical evaluation also addressed the issue of unrealistic setups, a common limitation of prior work, as well as realistic assumptions of these novel active attacks and respective baselines.

An extensive evaluation of attack performance revealed that AMI and AAI* are robust to several combinations of attack-specific components, some of which determined by the adversary. Particularly concerning is the ability to increase attack performance by conveniently tuning some parameters, such as number of target samples and model complexity. Not even a slightly blurred adversarial knowledge, or a different initialization of the target model, is able to deceive an adversary from conducting these active attacks, since not knowing a small part of target samples still yields effective attacks. Results from the practical evaluation showed that the proposed AMI and AAI* methods can pose a real threat to FL systems. Assuming that the neighbourhood of members is better represented than that of non members on the training set, AMI is able to identify a training sample 92.66% of the times, whereas AAI* determines that the user is not at home 85.84% of those training samples. Local density of target samples, meaning how well

represented they are on the training set, was correlated with these results and such insight paves the way for a possible cause of membership risk - local density, that should also increase attribute risk. Other than showcasing the real threat of these active attacks, the fact that the target model is able to fully recover its performance on the training set (while approaching model performance on the test set) reveals that active attacks may occur in FL systems without major long-term model utility loss, which motivates the need for defense mechanisms that preserve model utility.

Analysing both membership and attribute inference attacks allowed to the study their connection, another underestimated topic, as by understanding it one could develop defenses that mitigate both vulnerabilities. Conceptually, an AIA can be viewed as an extra step of a MIA and theoretically they are mutually reducible under some conditions. An analysis of the worst case scenario of AMI, AAI*, and MIA2AIA showed that membership risk is an upper bound for attribute inference risk. Empirical results support this insight given that AMI always achieved higher performance, in terms of neuron activation, and, under gradient uncertainty, membership advantage (proposed by Yeom et al. [62]) was significantly higher than attribute advantage (proposed by the same authors).

5.2 Defenses

With the goal of defending against these attacks, several detection strategies were developed for different levels of the defender’s knowledge. Particularly, a novel White-box Attack Detection and Mitigation (**WADM**) was developed to detect and mitigate both AMI and AAI* while preserving model utility. This novel approach detects malicious neurons based on abnormal changes in their weights distribution on consecutive training rounds, achieving practically perfect detection (100%) on most cases. Two novel black-box attack detection strategies, Black-box Attack Detection based on Accuracy (**BADAcc**) and Black-box Attack Detection based on ROC AUC (**BADAUC**), were also proposed to detect ongoing active attacks based on the monitorization of model performance in terms of accuracy and Area Under the Receiver Operating Characteristic Curve (**ROC AUC**), respectively. Black-box strategies were effective, detecting around 80% and 90% of active attacks while incurring in less than 5% and 12% of false alarms, respectively for each strategy. However, these black-box defenses required the client to leave the system. All of these defenses proved to be robust to several adversarial setups, particularly when the adversary attempted to evade detection by trading attack performance by a reduced probability of being detected. This insight suggests that active attacks affect model performance and weights distribution regardless of attack performance. Despite being considerably different, AMI and AAI* are effectively detected and mitigated by any of these defense strategies. Thus, according to these experiments, these defenses may be considered attack agnostic.

As detection strategies, minimization of false alarms is the key to reduce model utility loss, namely of BADAcc and BADAUC as triggering a false alarm forces the user to leave a benign system. False alarms, as well as missed attacks, of BADAcc and BADAUC were mainly caused by user-specific characteristics, such as local training set size and performance prior to the

attack. Hence, when black-box detection strategies are applied, some users may systematically be forced to leave a benign system while others may constantly be exposed to both active attacks, consequences of false alarms and missed attacks respectively. Remarkably, despite being theoretically threshold-dependent, BADAUC is effective on a wide range of thresholds and under several threshold selection policies (according to the priority of each type of error). WADM* may also incur in some false alarms, mainly on early training rounds when the model has not converged yet, however their cost is considerably smaller, as the user only changes the weights of a neuron classified as malicious. An adequate threshold choice and increasing model complexity may help to minimize this error, which naturally decays to zero along the training process. Meaning that weights distributions are expected to have very small variability as the model converges, only a malicious interference changes this behaviour.

Measuring the real utility of defenses, that is the ability to mitigate attacks on a practical scenario, is a relevant evaluation step to see whether the reduction of attack performance in terms of neuron activation (which significantly hindered active attacks) lead to a significant reduction in the adversary's ability to deal with gradient uncertainty. These experiments revealed that WADM* effectively reduces both AMI and AAI* to random guessing. However, WADM* mitigation phase does not improve the ability of the target model to recover from these attacks, although one must expected an approximate training performance after 5 training rounds. Nevertheless, WADM* improves upon black-box strategies, not only in terms of detection, but also in terms of the preservation of model utility while being equally effective at the mitigation, as it mitigates both active attacks by switching malicious for benign neurons instead of leaving the system.

Comparing to state-of-the-art defenses that mainly focus on improving generalization, which would not mitigate these attacks, as well-generalized models are also vulnerable, all of these defenses are effective, overcoming a major limitation of prior work while also covering the lack of detection strategies for privacy attacks. More importantly, the novel white-box defense mechanism (WADM*) overcomes another major limitation of prior work - reduced model utility loss.

By understanding the risks posed by active membership and attribute inference attacks and developing effective and robust countermeasures, this research contributes for an improvement of privacy-preserving FL systems and its ability to protect data privacy without sacrificing model utility. In particular, it proposes active attacks and attack detection strategies to address the limitations of unrealistic and inadequately measured attacks, as well as defense mechanisms that are effective and incur in low model utility loss.

5.3 Limitations and future research

While the present study has provided valuable insights into privacy risks of FL systems, it is important to acknowledge some limitations and how they might have impacted results. Following

up on identified gaps, one shall discuss suggestions of future research.

Regarding privacy risks evaluation, this study was limited to COPMODE dataset [1], mainly composed of binary input features. Effectiveness of both attacks is not expected to change significantly for a different dataset, however the real threat of these attacks might. A use-case in which a different data augmentation strategy is used may increase the real threat, as previously discussed in Chapter 2. Regarding defense mechanisms evaluation, this study assumed that any abnormal/significant drop in model performance is due to a privacy attack. However, current predictive systems already monitor model performance and are capable of reporting it to users, or to the service provider, if an abnormal reduction on evaluation metrics is observed. What they might not do is attribute that reduction to a privacy attack, as those changes may be caused by benign events, such as changes in the hidden process that generates data. Moreover, when the white-box detection and mitigation strategy is successfully applied, the adversary may be aware as the malicious neuron's gradient is much higher than expected for a successful attack (meaning one with reduced False Positive Rate (FPR)). Thus, it might be tempted to experiment other types of attacks, since those are expected to be mitigated, even though an attacker can always try different types of attacks.

Following these limitations, future research should measure FL model's vulnerability to these attacks when applied to other use-cases/datasets, as well as study how to attribute model performance drop to a privacy attack or to any other kind of benign event. It would also be interesting to see how can an adversary reduce active attacks impact on target model performance while balancing attack performance. Moreover, further research about the novel white-box attack detection strategy (WADM*) should focus on developing a methodology to fine tune the cutting threshold (above which any neuron is classified as malicious) for several use-cases, as the target model convergence may determine the success of that defense. Future research should also focus on a better understanding of model's vulnerability to these attacks and on developing proactive defense strategies where one makes the model itself more robust. In general, future research should focus on active attacks and countermeasures, preserving realistic attack setup, relationship between distinct privacy attacks, adequate risk measures, and effective solutions to reduce the vulnerability of models while preserving their utility.

To better understand a model's vulnerability to these type of attacks, one should delve into the effect of several data augmentation strategies on their real threat. In this study, oversampling was used to balance main task target distribution per client. However, in other use-cases, data augmentation techniques are used to exploit a sample's neighbourhood, which is expected to decrease the overlap between gradients' distribution of a malicious neuron that targets a member from one that targets a non member, therefore increasing membership risk. The community would benefit from knowing that some data augmentation techniques increase model vulnerability. To better understand a model's vulnerability to active attribute inference attacks, one should see if any statistical characteristic of the sensitive attribute has an impact on attack performance. That is, if there is some type of attribute whose value is easier to encode on a neuron while deactivating that neuron for any sample other than the target one and regardless of their attribute

value. Note that studying the side effect of data augmentation techniques is also relevant to prevent passive attacks that exploit how confident the target model is on the neighbourhood of target samples (Choquette-Choo et al. [12]). Related to passive attacks, another relevant future research direction is how to defend against AIAs that exploit the relationship between neuron’s activations and attribute values (Jayaraman et al. [28]) when there is a moderate relationship between the sensitive attribute and the main task label, as neural networks aim to precisely account for different contributions (activations) regarding distinct attribute values into the final output.

Proactive defense strategies, like input or gradient perturbation, are a possible solution to discourage an attacker from even attempting an active attack (if it knows that clients apply this strategy). Regarding gradient perturbation, pruning to 0 gradients with small magnitudes would deceive malicious inference, as a successfully trained malicious neuron is only activated by the target sample and its small neighbourhood. Although these defenses are expected to impact model performance significantly, judging from previous attempts [21].

Regarding input perturbation, a priori and contained in terms of noise’s variance, input perturbation does not defend against active membership inference, as stated by Nguyen et al. [42], because the malicious neuron can be trained to be only activated by the target sample’s neighbourhood. In fact, it is already activated by a small neighbourhood, as shown in Section 3.2.6. Instead of simply adding noise, one could perturb the input such that a transformation of it, and hopefully its neighbourhood, is no longer linearly separable. That is, one could train an autoencoder with the goal of minimizing reconstruction error, therefore preserving main task performance, and linear separability of the input’s representation after being transformed by the first hidden layer of the global model. This defense would be applied before every training round, after receiving the newest global model whose weights, from the first hidden layer, would be used to compute the penalty to add to the reconstruction loss. From this task should result an input transformation to apply on every training sample before feeding it to the global model. Ideally, every training sample should be used to train this autoencoder, possibly using a pre-trained set of weights. Questions related to the consumption of computational resources may arise from this approach.

About the application of defense mechanisms proposed in this work on a practical scenario, some questions related to the consumption of computational resources may arise, particularly regarding WADM*, since it requires the storage of the second fully connected layer. To account for this potential issue, instead of storing the complete weights distribution of neurons from this layer, the defender could store only what is necessary to compute the KL-Divergence, usually a histogram of much smaller size, then replace the malicious weights by a sample from this histogram. Also from the application of these defenses on a practical scenario, one may require additional details about how does a user applying some black-box strategy disconnects from the system, or if it is able to keep using the previous box (presumably a benign one), and if users that left the training process affect the ability of the model to recover from active attacks. Additionally, one of the major improvements of WADM* with respect to BADAcc and BADAUC

was the reduced user dependency, given that all users see the same global model and attack detection does not depend on its local performance. But users may have different concepts of sensitive data, therefore WADM* could apply a threshold selection policy that takes into account user's preferences, as some of them may value model utility more than they value their privacy. The same principle may be applied to both black-box strategies.

Appendix A

Subjective evaluation of SOTA

An extensive review of the state of the art (Chapter 2) allowed the author to build a critical view over the reviewed papers, which was summarized by an assessment of attacks and defenses through a graded evaluation according to several parameters (Figure 2.5). In addition to paving the way to develop novel approaches, that critical view was important to overcome some of the main limitations that have been disregarded, namely poorly measured attack threat/model leakage and unrealistic adversarial assumptions, as well as effective and with low model utility loss defense mechanisms. Table A.1 presents the final score attributed to each paper (whose type of privacy attack is indicated in the end of its row), as well as the respective score per parameter (explain in Section 2.5).

Paper	A. Effectiveness	A. Efficiency	A. Realistic	A. Accurately Measured	A. Connection Between Privacy Attacks	D. Effectiveness	D. Efficiency	D. Low Model Utility Loss	Final Score	Attack
[49]	1	0	1	1	0	1	1	1	0.80	MIA
[61]	1	1	1	1	0	-	-	-	0.80	MIA
[7]	1	0	1	1	0	-	-	-	0.60	MIA
[48]	0	1	1	1	0	1	1	0	0.63	MIA
[45]	1	1	1	1	0	1	1	0	0.73	MIA
[12]	1	1	1	0	0	0	1	0	0.47	MIA
[62]	1	1	0	1	1	-	-	-	0.80	MIA
[26]	1	1	1	1	0	1	1	0	0.73	MIA
[52]	1	1	1	0	0	1	1	1	0.80	MIA
[30]	1	1	1	1	0	1	1	1	0.90	MIA
[44]	1	0	0	0	0	-	-	-	0.20	MIA
[64]	1	0	1	0	0	-	-	-	0.40	MIA
[58]	1	0	1	0	0	-	-	-	0.40	MIA
[40]	1	1	1	0	0	-	-	-	0.60	MIA
[20]	1	1	1	0	0	0	0	0	0.30	MIA
[10]	1	0	0	0	0	0	1	1	0.43	MIA
[60]	1	1	1	1	0	-	-	-	0.80	MIA
[42]	1	1	1	1	0	0	1	0	0.57	MIA
[24]	1	1	0	0	1	-	-	-	0.60	REPR
[57]	1	1	0	1	1	-	-	-	0.80	REPR
[15]	1	1	1	0	0	1	1	1	0.80	MI
[65]	1	1	1	1	0	-	-	-	0.80	MI
[54]	1	1	1	1	0	-	-	-	0.80	MI
[68]	1	1	0	0	0	1	1	1	0.70	MI
[66]	1	1	0	0	0	-	-	-	0.40	MI
[41]	1	1	1	0	0	-	-	-	0.60	MI
[50]	1	1	1	1	1	1	1	1	1.00	MI
[21]	1	1	0	1	0	1	1	0	0.63	MI
[18]	1	1	0	1	0	-	-	-	0.60	MI
[67]	1	1	1	1	0	1	1	0	0.73	MI
[30]	1	1	1	1	0	1	1	1	0.90	MI
[17]	1	1	1	0	0	0	1	0	0.47	PIA
[43]	1	1	1	0	1	-	-	-	0.80	PIA
[38]	1	1	1	1	1	0	1	0	0.67	PIA
[35]	1	0	1	0	1	-	-	-	0.60	PIA
[8]	1	1	1	0	1	-	-	-	0.80	PIA
[53]	1	1	1	1	0	-	-	-	0.80	PIA
[51]	1	1	1	0	0	0	1	0	0.47	PIA
[29]	-	-	-	-	-	1	1	1	1.00	AIA
[62]	1	1	0	1	1	-	-	-	0.80	AIA
[28]	1	1	1	1	0	0	1	0	0.57	AIA
[36]	1	1	1	1	0	1	1	1	0.90	AIA
Score per metric	0.9756	0.8293	0.7561	0.561	0.2195	0.6364	0.9545	0.4545	0.6738	

Table A.1: Evaluation of SOTA

Appendix B

Theory behind novel active attacks

Novel active attacks explored throughout this thesis are based on maliciously modified neurons whose gradient leaks membership or attribute information. Sections 3.1.3 and 3.1.4 explained how the adversary modifies the neuron, however no guarantees were given regarding whether the gradient computation preserves the information encoded by the activation.

To establish some notation, that will be helpful in this Section, Figure B shows an example of a network. Let the red node denote the malicious neuron, s_k denote its activation ($\sum_i W_{ki} * s_i + b_k$), and w_k denote the weight of the link to the output node (s_{out}). Recall that an adversary performing AMI trains s_k to be positive whenever the target sample passes through the model and negative otherwise. Also recall that an adversary performing AAI* trains s_k to be positive whenever the target sample passes through the model with a positive attribute value, negative (closer to 0) with a negative attribute value and negative (as further away as possible from 0) otherwise. Additionally, Equation (B.1) shows how to compute the gradient of the malicious neuron (selected from the second fully connected layer, as illustrated in Figure B). This expression assumes that the network has the same architecture as the standard one used along this work, meaning two hidden layers and a final output node whose output function is SIGMOID (most common function for binary classification tasks).

$$gradient_k = \frac{\partial BCELoss(\hat{y})}{\partial s_k} = \frac{\partial BCELoss(\hat{y})}{\partial \hat{y}} * \frac{\partial \hat{y}}{\partial s_{out}} * \frac{\partial s_{out}}{\partial s_k} \quad (B.1)$$

Equation B.1 shows the chain rule to compute gradients. The first fraction of this expression refers to the derivative of the loss, Equation (B.2), with respect to the model's confidence, which is detailed in Equation (B.3), where cw_i denotes class weights. The second fraction of the chain rule refers to the derivative of the model's confidence, Equation (B.4), with respect to the activation of the output node (s_{out}) as shown is Equation (B.5). Finally, the last fraction refers to the derivative of the activation of the output node, Equation B.6, with respect to the activation of the malicious node (s_k), as detailed in Equation (B.7).

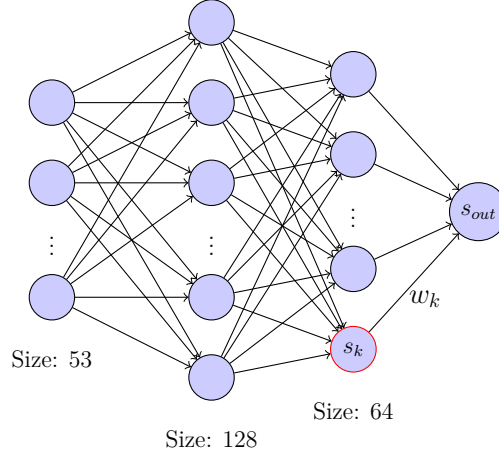


Figure B.1: Illustration of the target model (of standard complexity) with annotations.

$$\text{BCELoss}(\hat{y}) = \begin{cases} -cw_1 * \log(\hat{y}) & y = 1 \\ -cw_0 * \log(1 - \hat{y}) & y = 0 \end{cases} \quad \frac{\partial \text{BCELoss}(\hat{y})}{\partial \hat{y}} = \begin{cases} -cw_1 * \frac{1}{\hat{y}} & y = 1 \\ cw_0 * \frac{1}{1-\hat{y}} & y = 0 \end{cases} \quad (\text{B.2}) \quad (\text{B.3})$$

$$\hat{y} = \text{SIGMOID}(s_{out}) = \frac{1}{1 + \exp - s_{out}} \quad (\text{B.4})$$

$$\frac{\partial \hat{y}}{\partial s_{out}} = \hat{y} * (1 - \hat{y}) \quad (\text{B.5})$$

$$s_{out} = \sum_i^{64} \text{ReLU}(s_i) * w_i \quad (\text{B.6}) \quad \frac{\partial s_{out}}{\partial s_k} = w_k * \begin{cases} 1 & s_k > 0 \\ 0 & s_k \leq 0 \end{cases} \quad (\text{B.7})$$

An adversary conducting AMI over models trained with Rectified Linear Units (**ReLU**) activation function has the guarantee that the gradient is not null only if the malicious neuron is activated (presumably by the target sample only), as long as w_k is not null, as shown in Equation (B.8). To obtain this expression, one only needs to replace each fraction, of Equation (B.1), by its respective expression. Note that the derivative of the loss with respect to the model's confidence (Equation B.3) depends on the target sample's main task label. However, that never affects the nullity of the gradient, therefore it does not change the adversary's inference. It is also important to remark that the derivative of the model's confidence with respect to s_{out} is most likely not null (given that the target model is reduced to random guessing right after both active attacks, as shown in Figures 4.8 and 4.13) which does not change the adversary's inference.

$$\text{gradient}_k = \begin{cases} \frac{\partial \text{BCELoss}(\hat{y})}{\partial \hat{y}} * \hat{y} * (1 - \hat{y}) * w_k & \text{target sample is a member} \\ 0 & \text{target sample is a non-member} \end{cases} \quad (\text{B.8})$$

In its turn, models used to perform AAI* are trained with Exponential Linear Units (ELU) activation function (with $\alpha < 0$), as introduced in Equation (3.1.4.2) and used in Equations (B.9) and (B.10) to compute the respective fraction of the chain rule. An adversary conducting AAI* has the guarantee that the gradient sign leaks the desired information only if the rest of the components (of the gradient computation) do not change its sign, as shown in Equation (B.11). This guarantee is given by an alignment of the sign of w_k with the sign of the derivative of the loss with respect to the model's confidence, which depends on the target sample's main task label, Equation (B.3). It is important to remark that the derivative of the model's confidence with respect to s_{out} is never negative, which does not change the adversary's inference.

$$s_{out} = \sum_i^{64} \text{ELU}(s_i) * w_i \quad (\text{B.9}) \quad \frac{\partial s_{out}}{\partial s_k} = w_k * \begin{cases} 1 & s_k > 0 \\ \alpha * \exp(s_k) & s_k \leq 0 \\ 0 & s_k \ll 0 \end{cases} \quad (\text{B.10})$$

$$\text{gradient}_k = \begin{cases} \frac{\partial \text{BCELoss}(\hat{y})}{\partial \hat{y}} * \hat{y} * (1 - \hat{y}) * w_k & \text{target sample is a positive member} \\ \frac{\partial \text{BCELoss}(\hat{y})}{\partial \hat{y}} * \hat{y} * (1 - \hat{y}) * \alpha * \exp(s_k) * w_k & \text{target sample is a negative member} \\ 0 & \text{target sample is a non-member} \end{cases} \quad (\text{B.11})$$

Appendix C

Target model recovery after AMI

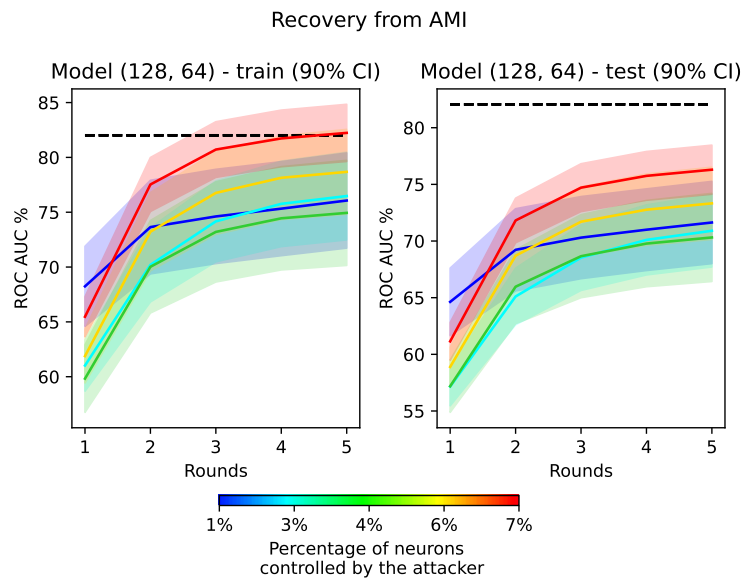


Figure C.1: Recovery (ROC AUC evolution) from multi-target AMI on train and test set of models of standard complexities over 5 training rounds

Figure 3.14 from Section 3.2.5 illustrates the target model's ability to recover from AMI, yet it showed that, for a smaller percentage of controlled neurons, target models (particularly of standard complexity) had a reduced capacity to recover from AMI, as well as higher variance. However, recovery from AMI improves with an increase of the percentage of controlled neurons, as suggested by Figure 3.14, and now observable by a progressive reduction of variance and increase of average in Figure C.1 for models of standard complexity. From this figure, one can see that when the adversary controls 7% of the neurons, the target model is able to fully recover from AMI on the train set, as with for a higher percentage of controlled neurons. Similarly to Section 3.2.5, the target model is unable to fully recover on the test set.

Appendix D

Experiments with other datasets

For generalization purposes, it is common to evaluate novel approaches on several datasets. Since the main dataset used for throughout this thesis (COP-MODE [1], detailed in Section 3.2.1.1) contains mostly binary input features, it is relevant to evaluate active attacks on a dataset containing real-valued features. Publicly accessible datasets from the computer vision domain are the most common within the community of privacy attacks, namely Cifar-10/100 [32], MNIST [13], and CelebA [34]. For reasons further detailed in Section D.1, only CelebA is used to evaluate these novel attacks.

D.1 CelebA

CelebFaces Attributes Dataset (CelebA) is a large-scale face attributes dataset with more than 200K celebrity images of over 10K distinct identities, each with 40 binary attribute annotations. These images cover large pose variations and background clutter. CelebA can be used for a variety of machine learning tasks, namely attribute classification [34] and face recognition [30]. Among several well-known real-valued datasets, particularly in the image domain, that are frequently used to assess attack performance, CelebA was the most suitable for attribute inference purposes given that each image has a vector of 40 binary attributes associated. Thus, one could use the image and some of these annotations for a classification task.

Experiments of AAI* and MIA2AIA, assume that the adversary has access to a vector representation of the image obtained after passing the matrix representation for a series of interpolated convolutional and max-pooling layers (common practice on this type of learning task). This representation is also called feature embeddings. Ideally, the adversary must also have access to all binary attributes used as predictors except for the sensitive attribute, even though previous experiments (Section 3.2.4.5) suggested that an adversary can still be successful even when it does not know a few predictive features. To perform AAI*, it must also have the main task label, or be able to estimate it using the target model. The attack pipeline is the same, only this time the input vector has some real-valued entries that represent the image.

Note that the assumption that the adversary has access to target feature embeddings may seem unrealistic at first, but these embeddings may be shared (instead of images) as they seem to protect users privacy. In fact, privacy-preserving face embeddings have been a relevant topic of research as leakage from these can raise severe privacy risks, e.g they can be leveraged to reconstruct the original face image (Han et al. [22]). Additionally, one could argue that if the adversary has access to those embeddings than it could train a classification network to predict the binary sensitive attribute. However, the adversary would need a large amount of labeled data for that purpose. Thus, it is best to use the target model, instead of training a network from scratch. Moreover, one must not forget that, more than inferring about the sensitive attribute, the adversary would be, ideally, inferring about the target sample’s membership status, as a perfect attribute inference over a machine learning model would only improve the adversary’s knowledge with respect to training samples, that is ideally it would increase target model leakage.

A brief exploratory data analysis, particularly about the binary features, revealed that the following attributes (for which the proportion of the most frequent label is indicated) were moderately correlated with one the most balanced binary attributes (here used as target variable) - *Attractive* (51.25%).

- *Wearing lipstick* (53.0%)
- *Male* (57.8%)
- *Heavy makeup* (61.4%)
- *Arched eyebrows* (73.5%)
- *Big nose* (76.2%)
- *Young* (77.9%)

These features were selected as predictors and included in the input. To obtain the feature embeddings of images, a pre-trained Resnet-18 model from Img2Vec2¹ was used to get a vector of 512 entries. Then, 32 federated learning target models (assuming random division into 10 clients) were trained over which several iterations of active attacks yielded confidence intervals for attack performance with 90% of confidence, similarly to the main use-case of this thesis. Federated learning target models that were used for these experiments had 1024 and 518 neurons on the first and second fully connected layer, respectively. These target models are also evaluated with ROC AUC and achieved a train performance of [88.13%, 88.17%] and test performance on [87.96%, 88.0%] when trained with ELU activation function (to perform AAI*) and [88.47%, 88.5%] and [88.24%, 88.27%] on the train and test set when trained with ReLU activation function (to perform MIA2AIA). Note that there is a very small train test gap, thus one can assume passive MIAs, or AIAs based on those passive attacks, would most likely be random guessing.

¹<https://github.com/christiansafka/img2vec>

D.2 AMI

Regarding the generalization of AMI to other datasets, Nguyen et al. [42] already presented results of AMI on the most known datasets. CelebA results were especially great, as FPR approached 0 faster than the rest. Similarly to COP-MODE, TPR is also always 100% and FPR tends to approach 0 with an increase of both training epochs and number of neurons of the first fully connected layer as shown in Figure D.1. This figure shows that AMI is effective on several well-known datasets from the vision domain, by showing that TPR is always 100%, FPR tends to 0%, and equivalently membership advantage (given by $TPR - FPR$ tends to 100%) with an increase of the number of epochs (to train the adversarial network).

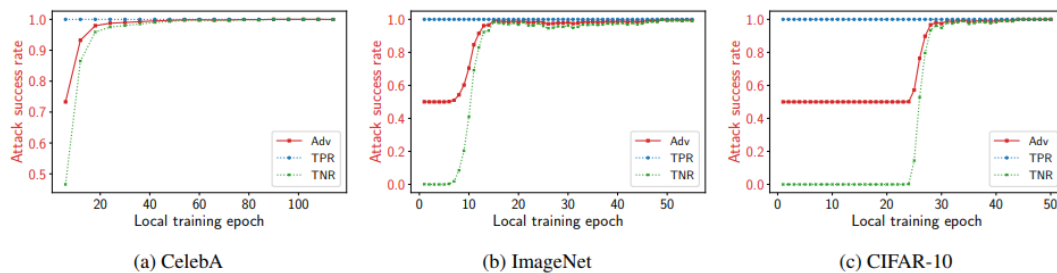


Figure D.1: AMI performance on several well-known datasets from the vision domain (Figure 9, Appendix D [42]).

D.3 AAI^{*} and MIA2AIA

Since AAI^{*} is first proposed in this work, additional experiments are needed to show that AAI^{*} generalizes to real-valued input vectors. Despite the fact that MIA-based approaches to perform attribute inference are not new, given that AMI was recently proposed (2023), there is still no published results of using AMI for attribute inference purposes. Thus, additional experiments are needed to show that MIA2AIA generalizes to real-valued input vectors.

After proper hyperparameter tuning, Figure D.2 illustrates TPR and FPR of single-target AAI^{*} attacks on CelebA for several sensitive attributes. Note that these experiments are balanced with respect to the target sample’s real attribute value and membership. From this figure, one can see that AAI tends to outperform (on average) both versions of MIA2AIA, in terms of TPR, particularly when the proportion of the most frequent value tends to 50%, that is on increasingly balanced attribute distributions (*Wearing Lipstick* and *Male*). Although for some imbalanced attributes, AAI^{*} also outperforms both versions of MIA2AIA TPR on average (*Arched Eyebrows* and *Young*). Moreover, the difference between MIA2AIA version with and without data imputation does not present a clear tendency to increase nor decrease depending on the sensitive attribute distribution. The same difference is also not properly explained by FPR. Regarding FPR, AAI^{*} improves even more as it is not significantly different from 0 (average is

very close to 0 for all cases), whereas MIA2AIA FPR is only not significantly different from 0 on one of the most imbalanced attributes (*Big nose*). Furthermore, MIA2AIA FPR presents a decreasing trend with respect to the imbalance nature of the sensitive attribute distribution (as the decreasing tendency towards the right, in Figure D.2, suggests). It is worth mentioning that MIA2AIA FPR increases considerably comparing to COP-MODE experiments (at most 2% for models of standard complexity, see Figure 3.9), this time reaching up to 17.5%, whereas AAI FPR decreases significantly (at least higher than 0.5% on all experiments from Figure 3.9).

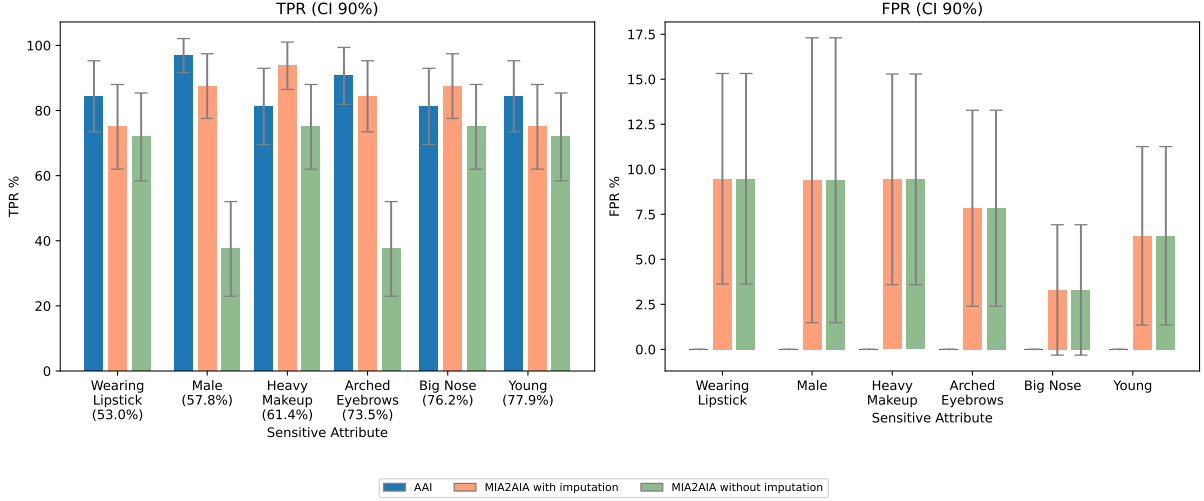


Figure D.2: Single-target AAI* and MIA2AIA TPR and FPR on federated models (on CelebA dataset) for several sensitive attributes. The baseline for each attribute is inside parenthesis.

From theory to practice some performance may be lost, that is the adversary may not be able to succeed at its inference due to the effect of false positives. Thus, it is important to see if the superiority of AAI* still holds, similarly to the evaluation in Section 3.2.6. An extensive set of experiments, to build a confidence interval of ROC AUC (restricted to a maximum FPR of 25%), revealed that even under gradient uncertainty (caused by false positives) AAI* outperforms MIA2AIA significantly. As one can see in Figure D.3 where restricted ROC AUC under gradient uncertainty of single-target AAI* and MIA2AIA attacks over federated models (on CelebA) is presented. When the sensitive attribute is *Male*, AAI* achieves [89.71%, 93.77%] comparing to MIA2AIA that achieves [56.1%, 62.51%], when target samples are members. When the sensitive attribute is *Arched eyebrows*, AAI* achieves [79.11%, 85.73%] comparing to MIA2AIA that achieves [48.76%, 55.1%], when target samples are members. On the other hand, when target samples are non-members, AAI* is not significantly different from random guessing, for both attributes. However, MIA2AIA is significantly different than random guessing when targeting *Male*. Moreover, comparing MIA2AIA performance on members and non members, one can see that these intervals are not disjoint, therefore MIA2AIA is not significantly better at attribute inference of members. According to Yeom et al. [62], this attribute inference adversary does not increase target model leakage, since it apparently cannot leverage having access to the target model to improve its inference on training samples.

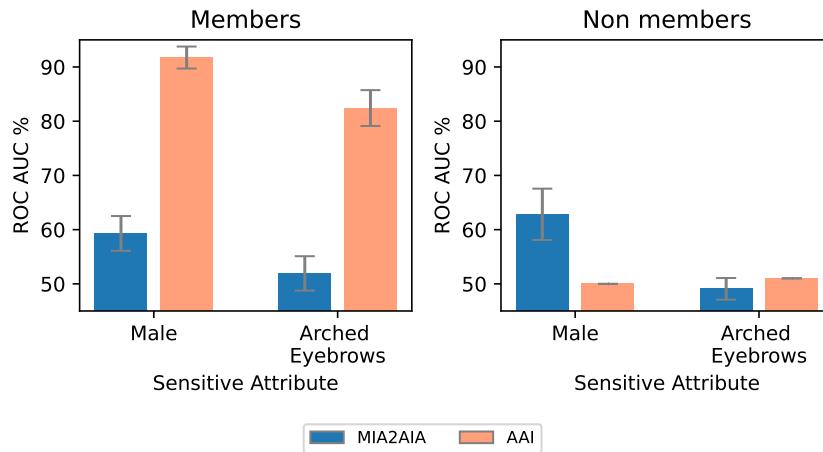


Figure D.3: Single-target AAI* and MIA2AIA ROC AUC % (restricted to a maximum FPR of 25%) under gradient uncertainty, over federated models on celebA dataset. Only two sensitive attribute values are presented, one with a balanced and another one with an imbalanced distribution.

D.4 Conclusion

Evaluating active attacks on different datasets revealed that these attacks generalize well for real-valued input features. More than showcasing their effectiveness regardless of the input type, these experiments enable a comparison with state-of-the-art results. Even though additional experiments would be needed to do a fair comparison, that is applying previously proposed attacks on the same target model. Nevertheless, AMI results, on several well known datasets, show that AMI improves upon prior work (Nguyen et al. [42]). Particularly since it does not require the target model to have a huge train test gap (overfitting), as it is a common practice on several papers (Shokri et al. [40], for example). Regarding AAI* results on CelebA, these show that AAI* improve upon the best state-of-the-art approaches (MIA2AIA), especially because it can increase target model leakage (by improving attribute inference of training samples when comparing to the performance on non-training samples), besides improving significantly upon its alternative in terms of restricted ROC AUC on training samples. Note that AIAs are under-explored in the literature and their performance has been poorly measured, according to Yeom et al. [62] and Jayaraman et al. [28], as explained in Section 2.4.2. Therefore, any direct comparisons with prior work would not be conclusive, nor fair.

Bibliography

- [1] [O projeto cop-mode](#). (consulted in April 2024).
- [2] [General data protection regulation](#).
- [3] [The difference between l1 and l2 regularization](#). (consulted in August 2024).
- [4] Giuseppe Ateniese, Luigi V. Mancini, Angelo Spognardi, Antonio Villani, Domenico Vitali, and Giovanni Felici. [Hacking smart machines with smarter ones: How to extract meaningful data from machine learning classifiers](#). *Int. J. Secur. Netw.*, 10(3):137–150, sep 2015. ISSN: 1747-8405. doi:10.1504/IJSN.2015.071829.
- [5] K. A. Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H. Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. [Practical secure aggregation for federated learning on user-held data](#). In *NIPS Workshop on Private Multi-Party Machine Learning*, 2016.
- [6] André Brandão, Ricardo Mendes, and João P. Vilela. [Prediction of mobile app privacy preferences with user profiles via federated learning](#). *Proceedings of the Twelfth ACM Conference on Data and Application Security and Privacy*, 2022.
- [7] Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis, and Florian Tramèr. [Membership inference attacks from first principles](#). In *2022 IEEE Symposium on Security and Privacy (SP)*, pages 1897–1914, 2022. doi:10.1109/SP46214.2022.9833649.
- [8] Harsh Chaudhari, John Abascal, Alina Oprea, Matthew Jagielski, Florian Tramèr, and Jonathan Ullman. [Snap: Efficient extraction of private properties with poisoning](#). In *2023 IEEE Symposium on Security and Privacy (SP)*, pages 400–417, 2023. doi:10.1109/SP46215.2023.10179334.
- [9] Congcong Chen, Lifei Wei, Lei Zhang, Ya Peng, and Jianting Ning. [Deep-guard: Backdoor attack detection and identification schemes in privacy-preserving deep neural networks](#). *Security and Communication Networks*, 2022(1):2985308, 2022. doi:https://doi.org/10.1155/2022/2985308.
- [10] Jiale Chen, Jiale Zhang, Yanchao Zhao, Hao Han, Kun Zhu, and Bing Chen. [Beyond model-level membership privacy leakage: an adversarial approach in federated learning](#). In

- 2020 29th International Conference on Computer Communications and Networks (ICCCN), pages 1–9, 2020. doi:10.1109/ICCCN49398.2020.9209744.
- [11] Steven Chen, Nicholas Carlini, and David Wagner. [Stateful detection of black-box adversarial attacks](#). In *Proceedings of the 1st ACM Workshop on Security and Privacy on Artificial Intelligence*, SPAI '20, page 30–39, New York, NY, USA, 2020. Association for Computing Machinery. ISBN: 9781450376112. doi:10.1145/3385003.3410925.
- [12] Christopher A. Choquette-Choo, Florian Tramèr, Nicholas Carlini, and Nicolas Papernot. [Label-only membership inference attacks](#). In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 1964–1974. PMLR, 18–24 Jul 2021.
- [13] Li Deng. [The mnist database of handwritten digit images for machine learning research \[best of the web\]](#). *IEEE Signal Processing Magazine*, 29(6):141–142, 2012. doi:10.1109/MSP.2012.2211477.
- [14] Cynthia Dwork. *Differential Privacy*, pages 338–340. Springer US, Boston, MA, 2011. ISBN: 978-1-4419-5906-5. doi:10.1007/978-1-4419-5906-5_752.
- [15] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. [Model inversion attacks that exploit confidence information and basic countermeasures](#). *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, 2015.
- [16] João Gama, Pedro Medas, Gladys Castillo, and Pedro Pereira Rodrigues. [Learning with drift detection](#). In *Brazilian Symposium on Artificial Intelligence*, 2004.
- [17] Karan Ganju, Qi Wang, Wei Yang, Carl A. Gunter, and Nikita 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*, CCS '18, page 619–633, New York, NY, USA, 2018. Association for Computing Machinery. ISBN: 9781450356930. doi:10.1145/3243734.3243834.
- [18] Jonas Geiping, Hartmut Bauermeister, Hannah Dröge, and Michael Moeller. [Inverting gradients - how easy is it to break privacy in federated learning?](#) In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN: 9781713829546.
- [19] Jianping Gou, Baosheng Yu, Stephen John Maybank, and Dacheng Tao. [Knowledge distillation: A survey](#). *arXiv preprint arXiv:2006.05525*, abs/2006.05525, 2020.
- [20] Yuhao Gu, Yuebin Bai, and Shubin Xu. [Cs-mia: Membership inference attack based on prediction confidence series in federated learning](#). *Journal of Information Security and Applications*, 67:103201, 2022. ISSN: 2214-2126. doi:https://doi.org/10.1016/j.jisa.2022.103201.
- [21] Samyak Gupta, Yangsibo Huang, Zexuan Zhong, Tianyu Gao, Kai Li, and Danqi Chen. [Recovering private text in federated learning of language models](#). In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

- [22] Dong Han, Yufan Jiang, Yong Li, Ricardo Mendes, and Joachim Denzler. [Robust skin color driven privacy preserving face recognition via function secret sharing](#). *arXiv preprint arXiv:2407.05045*, 2024.
- [23] Andrew Hard, Kanishka Rao, Rajiv Mathews, Françoise Beaufays, Sean Augenstein, Hubert Eichner, Chloé Kiddon, and Daniel Ramage. [Federated learning for mobile keyboard prediction](#). *CoRR*, abs/1811.03604, 2018.
- [24] Briland Hitaj, Giuseppe Ateniese, and Fernando 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*, CCS '17, page 603–618, New York, NY, USA, 2017. Association for Computing Machinery. ISBN: 9781450349468. doi:10.1145/3133956.3134012.
- [25] Hongsheng Hu, Zoran Salcic, Lichao Sun, Gillian Dobbie, Philip S. Yu, and Xuyun Zhang. [Membership inference attacks on machine learning: A survey](#). *ACM Comput. Surv.*, 54(11s), sep 2022. ISSN: 0360-0300. doi:10.1145/3523273.
- [26] Bo Hui, Yuchen Yang, Haolin Yuan, Philippe Burlina, Neil Zhenqiang Gong, and Yinzhi Cao. [Practical blind membership inference attack via differential comparisons](#). In *Proceedings of the Network and Distributed Systems Security (NDSS) Symposium 2021*, Virtual, February 21–25 2021. Internet Society. ISBN: 1-891562-66-5. doi:10.14722/ndss.2021.24293.
- [27] D R Janardhana, V Pavan Kumar, S R Lavanya, and A P Manu. [Detecting security and privacy attacks in iot network using deep learning algorithms](#). In *2021 IEEE International Conference on Distributed Computing, VLSI, Electrical Circuits and Robotics (DISCOVER)*, pages 35–40, 2021. doi:10.1109/DISCOVER52564.2021.9663586.
- [28] Bargav Jayaraman and David Evans. [Are attribute inference attacks just imputation?](#) In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, CCS '22, page 1569–1582, New York, NY, USA, 2022. Association for Computing Machinery. ISBN: 9781450394505. doi:10.1145/3548606.3560663.
- [29] Jinyuan Jia and Neil Zhenqiang Gong. [AttriGuard: A practical defense against attribute inference attacks via adversarial machine learning](#). In *USENIX Security Symposium*, 2018.
- [30] Myeongseob Ko, Xinyu Yang, Zhengjie Ji, Hoang Anh Just, Peng Gao, Anoop Kumar, and Ruoxi Jia. [Privmon: A stream-based system for real-time privacy attack detection for machine learning models](#). In *Proceedings of the 26th International Symposium on Research in Attacks, Intrusions and Defenses (RAID '23)*, pages 1–18, New York, NY, USA, 2023. ACM. ISBN: 979-8-4007-0765-0/23/10. doi:10.1145/3607199.3607232.
- [31] Zico Kolter and Aleksander Madry. [Adversarial robustness: Theory and practice](#). NeurIPS 2018 tutorial.
- [32] Alex Krizhevsky. [Learning multiple layers of features from tiny images](#). 2009.

- [33] Huiying Li, Shawn Shan, Emily Wenger, Jiayun Zhang, Haitao Zheng, and Ben Y. Zhao. [Blacklight: Scalable defense for neural networks against Query-Based Black-Box attacks](#). In *31st USENIX Security Symposium (USENIX Security 22)*, pages 2117–2134, Boston, MA, August 2022. USENIX Association. ISBN: 978-1-939133-31-1.
- [34] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. [Deep learning face attributes in the wild](#). In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.
- [35] Saeed Mahloujifar, Esha Ghosh, and Melissa Chase. [Property inference from poisoning](#). In *2022 IEEE Symposium on Security and Privacy (SP)*, pages 1120–1137, 2022. doi:10.1109/SP46214.2022.9833623.
- [36] Mohammad Malekzadeh, Anastasia Borovykh, and Deniz Gündüz. [Honest-but-curious nets: Sensitive attributes of private inputs can be secretly coded into the entropy of classifiers’ outputs](#). *CoRR*, abs/2105.12049, 2021.
- [37] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. [Communication-Efficient Learning of Deep Networks from Decentralized Data](#). In Aarti Singh and Jerry Zhu, editors, *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pages 1273–1282. PMLR, 20–22 Apr 2017.
- [38] Luca Melis, Congzheng Song, Emiliano De Cristofaro, and Vitaly Shmatikov. [Exploiting unintended feature leakage in collaborative learning](#). In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 691–706, 2019. doi:10.1109/SP.2019.00029.
- [39] Ricardo Mendes, André Brandão, João P. Vilela, and Alastair R. Beresford. [Effect of user expectation on mobile app privacy: A field study](#). In *2022 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pages 207–214, 2022. doi:10.1109/PerCom53586.2022.9762379.
- [40] Milad Nasr, Reza Shokri, and Amir Houmansadr. [Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning](#). In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 739–753, 2019. doi:10.1109/SP.2019.00065.
- [41] Ngoc-Bao Nguyen, Keshigeyan Chandrasegaran, Milad Abdollahzadeh, and Ngai-Man Cheung. [Re-thinking model inversion attacks against deep neural networks](#). In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16384–16393, 2023. doi:10.1109/CVPR52729.2023.01572.
- [42] Truc Nguyen, Phung Lai, Khang Tran, NhatHai Phan, and My T Thai. [Active membership inference attack under local differential privacy in federated learning](#). In *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, volume 206 of *Proceedings of Machine Learning Research*, pages 5714–5730. PMLR, 2023.

- [43] Mathias P. M. Parisot, Balazs Pejo, and Dayana Spagnuolo. [Property inference attacks on convolutional neural networks: Influence and implications of target model’s complexity](#). *CoRR*, abs/2104.13061, 2021.
- [44] Anastasia Pustozeroova and Rudolf Mayer. [Information leaks in federated learning](#). *Proceedings 2020 Workshop on Decentralized IoT Systems and Security*, 2020.
- [45] Shadi Rahimian, Tribhuvanesh Orekondy, and Mario Fritz. [Sampling attacks: Amplification of membership inference attacks by repeated queries](#). *arXiv preprint arXiv:2009.00395*, abs/2009.00395, 2020.
- [46] Maria Rigaki and Sebastian Garcia. [A survey of privacy attacks in machine learning](#). *ACM Comput. Surv.*, 56(4), nov 2023. ISSN: 0360-0300. doi:10.1145/3624010.
- [47] Ahmed Salem, Giovanni Cherubin, David Evans, Boris Köpf, Andrew Paverd, Anshuman Suri, Shruti Tople, and Santiago Zanella-Béguelin. [Sok: Let the privacy games begin! a unified treatment of data inference privacy in machine learning](#). *Proceedings IEEE Symposium on Security and Privacy*, 2023.
- [48] Ahmed Salem, Yang Zhang, Mathias Humbert, Mario Fritz, and Michael Backes. [ML-leaks: Model and data independent membership inference attacks and defenses on machine learning models](#). *CoRR*, abs/1806.01246, 2018.
- [49] 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)*, pages 3–18, Los Alamitos, CA, USA, may 2017. IEEE Computer Society. doi:10.1109/SP.2017.41.
- [50] Congzheng Song and Ananth Raghunathan. [Information leakage in embedding models](#). In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security, CCS ’20*, page 377–390, New York, NY, USA, 2020. Association for Computing Machinery. ISBN: 9781450370899. doi:10.1145/3372297.3417270.
- [51] Congzheng Song and Vitaly Shmatikov. [Overlearning reveals sensitive attributes](#). In *Proceedings of the 8th International Conference on Learning Representations (ICLR ’20)*, 2020.
- [52] Liwei Song and Prateek Mittal. [Systematic evaluation of privacy risks of machine learning models](#). In *30th USENIX Security Symposium (USENIX Security 21)*, pages 2615–2632. USENIX Association, August 2021. ISBN: 978-1-939133-24-3.
- [53] Anshuman Suri and David Evans. [Formalizing and estimating distribution inference risks](#). In *22nd Privacy Enhancing Technologies Symposium (PETS)*, 2022.
- [54] Hiroaki Tanaka, Wataru Yamada, Keiichi Ochiai, Rina Okada, Satoshi Hasegawa, and Daizo Ikeda. [Evaluation indicator for model inversion attack](#). In *AdvML ’20: Workshop on Adversarial Learning Methods for Machine Learning and Data Mining*, page 5, New York, NY, USA, August 24 2020. ACM.

- [55] Jeffrey G. Wang, Jason Wang, Marvin Li, and Seth Neel. [Pandora’s white-box: Precise training data detection and extraction in large language models](#), 2024.
- [56] Tianhao Wang. [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 (CCS)*, New York, NY, USA, October 2018. ACM. doi:10.1145/3243734.3243834.
- [57] Zhibo Wang, Mengkai Song, Zhifei Zhang, Yang Song, Qian Wang, and Hairong Qi. [Beyond inferring class representatives: User-level privacy leakage from federated learning](#). In *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*, page 2512–2520. IEEE Press, 2019. doi:10.1109/INFOCOM.2019.8737416.
- [58] Yuanyuan Xie, Bing Chen, Jiale Zhang, and Wenjuan Li. [Algans: Enhancing membership inference attacks in federated learning with gans and active learning](#). In *2022 IEEE International Symposium on Product Compliance Engineering - Asia (ISPCE-ASIA)*, pages 1–6, 2022. doi:10.1109/ISPCE-ASIA57917.2022.9971068.
- [59] Xingxing Xiong, Shubo Liu, Dan Li, Zhaohui Cai, and Xiaoguang Niu. [A comprehensive survey on local differential privacy](#). *Security and Communication Networks*, 2020:29, 2020. doi:10.1155/2020/8829523.
- [60] Xiaoyun Xu, Jingzheng Wu, Mutian Yang, Tianyue Luo, Xu Duan, Weiheng Li, Yanjun Wu, and Bin Wu. [Information leakage by model weights on federated learning](#). In *Proceedings of the 2020 Workshop on Privacy-Preserving Machine Learning in Practice*, PPMLP’20, page 31–36, New York, NY, USA, 2020. Association for Computing Machinery. ISBN: 9781450380881. doi:10.1145/3411501.3419423.
- [61] Jiayuan Ye, Aadyaa Maddi, Sasi Kumar Murakonda, Vincent Bindschaedler, and Reza Shokri. [Enhanced membership inference attacks against machine learning models](#). CCS ’22, page 3093–3106, New York, NY, USA, 2022. Association for Computing Machinery. ISBN: 9781450394505. doi:10.1145/3548606.3560675.
- [62] S. Yeom, I. Giacomelli, M. Fredrikson, and S. Jha. [Privacy risk in machine learning: Analyzing the connection to overfitting](#). In *2018 IEEE 31st Computer Security Foundations Symposium (CSF)*, pages 268–282, Los Alamitos, CA, USA, jul 2018. IEEE Computer Society. doi:10.1109/CSF.2018.00027.
- [63] Xuefei Yin, Yanming Zhu, and Jiankun Hu. [A comprehensive survey of privacy-preserving federated learning: A taxonomy, review, and future directions](#). *ACM Comput. Surv.*, 54(6), jul 2021. ISSN: 0360-0300. doi:10.1145/3460427.
- [64] Jingwen Zhang, Jiale Zhang, Junjun Chen, and Shui Yu. [Gan enhanced membership inference: A passive local attack in federated learning](#). In *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, pages 1–6, 2020. doi:10.1109/ICC40277.2020.9148790.

-
- [65] Y. Zhang, R. Jia, H. Pei, W. Wang, B. Li, and D. Song. [The secret revealer: Generative model-inversion attacks against deep neural networks](#). In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 250–258, Los Alamitos, CA, USA, jun 2020. IEEE Computer Society. doi:10.1109/CVPR42600.2020.00033.
- [66] Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. [iDLG: Improved Deep Leakage from Gradients](#). *arXiv preprint arXiv:2001.02610*, 2020.
- [67] J. Zhao, A. Sharma, A. Elkordy, Y. H. Ezzeldin, S. Avestimehr, and S. Bagchi. [Loki: Large-scale data reconstruction attack against federated learning through model manipulation](#). In *2024 IEEE Symposium on Security and Privacy (SP)*, pages 30–30, Los Alamitos, CA, USA, may 2024. IEEE Computer Society. doi:10.1109/SP54263.2024.00030.
- [68] Ligeng Zhu, Zhijian Liu, and Song Han. [Deep leakage from gradients](#). In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.