Active Attribute Inference Against Well-Generalized Models In Federated Learning

Catarina Gomes
CRACS/INESC TEC and Dept. of Computer Science
Faculty of Sciences, University of Porto
Porto, Portugal
up201804545@edu.fc.up.pt

Ricardo Mendes
CISUC and Dept. of Informatics Engineering
University of Coimbra
Coimbra, Portugal
rscmendes@dei.uc.pt

João P. Vilela
CRACS/INESC TEC, CISUC and Dept. of Computer Science
Faculty of Sciences, University of Porto
Porto, Portugal
jvilela@fc.up.pt

Abstract*—Federated Learning (FL), a distributed learning mechanism where data is decentralized across multiple devices and periodic gradient updates are shared, is an alternative to centralized training that aims to address privacy issues arising from raw data sharing. Despite the expected privacy benefits, prior research showcases the potential privacy leakage derived from overfitting, exploited by passive attacks. However, limited attention has been given to understanding and defending against active threats that increase model leakage by interfering with the training process, instead of relying on overfitting. This work addresses this gap by introducing Active Attribute Inference (AAI*), a novel active attack that encodes sensitive attribute information by making any targeted training sample leave a distinguishable footprint on the gradient of maliciously modified neurons [8]. Results, using two real-world datasets, show that it is possible to successfully encode sensitive information incurring a small error in terms of neuron activation. More importantly, on a practical scenario, AAI* can improve upon a state-of-the-art approach by achieving over 90% of restricted ROC AUC, therefore increasing model leakage. To defend against such active attacks, this work introduces several attack detection strategies tailored for different levels of the defender's knowledge. Including the novel White-box Attack Detection Mechanism (WADM*) that detects abnormal changes in weights distribution, and two black-box strategies based on the monitorization of model performance. Results show that the detection rate can be 100% on both datasets. Remarkably, WADM* reduces any attack to random guessing while preserving model utility, offering significant improvements over existing defenses, particularly when clients are non-IID. By proposing active attacks against well-generalized models and effective countermeasures, this research contributes to a better understanding of privacy in FL systems.

***Index Terms*—Federated Learning, Data Privacy, Membership Inference Attack, Attribute Inference Attack, Attack Detection**

1. Introduction

Artificial Intelligence (AI) is leading a technological paradigm shift that impacts several application fields. Sophisticated machine learning 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 [33]. Despite being especially pertinent in centralized learning, whereby all data is held by a single entity, privacy concerns may arise even in distributed frameworks 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, 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 [24]. 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, which can raise *severe privacy risks* to individuals. In addition to the potential violation of data regulations (e.g. General Data Protection Regulation [1]), highly accurate MIAs are useful to demonstrate privacy leakage [29]. In an Attribute Inference Attack (AIA), an attacker with access to the trained model attempts to infer the sensitive attribute value of a training sample [24]. The level of threat of AIAs ultimately depends on the destination of the inferred information, which can be leveraged, for example, to deliver personalized advertisements to users [13].

Past research primarily focused on passive attacks or unrealistic scenarios (e.g. required access to individual gradient updates), leaving a *gap in understanding and defending against active attacks*, which manipulate models to extract sensitive information, thus posing a significant threat to FL systems. These do not require a less accurate or overfitted model, 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, are ineffective. In fact, given that active privacy attacks are under-explored, so are defense mechanisms to mitigate them, namely defenses that balance the privacy-utility trade-off.

This work addresses this gap by focusing on active and realistic AIAs over FL models, that do not rely on overfitting but increase model leakage by encoding

attribute information on model parameters. With these attacks, the ultimate goal is to identify vulnerabilities of well-generalized distributed learning models, that often carry a false sense of security. Additionally, since these attacks modify the global model, a novel defense mechanism is developed to mitigate them while minimizing model utility loss, resorting to distance-based techniques to detect malicious modifications of model parameters. These novel attacks and defense mechanisms are applied on challenging scenarios, such as FL systems with non-IID clients and data imbalance problems, showcasing that these novelties improve upon prior work particularly on challenging setups. By understanding the risks posed by active attacks and developing effective countermeasures, this work aims to contribute to a better understanding of privacy in FL systems.

The remainder of this paper is structured as follows. Section 2 presents background concepts and a review of the state of the art on privacy attacks and defenses. Section 3 introduces and evaluates novel active attacks performed by a malicious central server against federated models trained on several datasets. Section 4 introduces and evaluates three attack detection strategies tailored for distinct levels of the defender's knowledge applied against attacks evaluated in Section 3. Section 5 discusses the main findings and presents suggestions for future work.

2. State of the art

Federated Learning (FL), proposed by Google in 2016 [18], is an alternative to centralized training that leaves private training data distributed on mobile devices and trains a shared model by aggregating locally computed updates. An FL system involves a central server (service provider) that aggregates each participant's (also referred to as client) contribution. In this work, only horizontal FL, referred to as FL, is used. This framework requires private datasets to have an identical feature space but different identity spaces, meaning distinct instances described by the same set of features at each client.

2.1. Privacy Attacks

Attacks on machine learning (ML) models encompass a range of strategies to exploit vulnerabilities and compromise data privacy. An adversary can have distinct goals [24], such as membership, attribute, or property inference, all of which constitute a violation of the privacy of data owners, leveraged upon a released model or the adversary's participation in a model training system. Adversaries can be described by their level of access and their effect on the target model (if any), which ultimately determines the adversarial approach.

2.1.1. Membership Inference Attacks. In an MIA, an attacker with access to the trained model attempts to infer whether a given sample was used in the training process [24] (member). Most MIAs exploit the fact that the prediction of members is directly optimized (up to a point) and the non-members' predictions only hopefully follow the same trend [4], [9], [21], [26], [31], [32], which usually leads to a performance gap. Hence, *overfitting is pointed out as the main reason for membership vulnerability* [32]. Although some MIAs look for high-risk samples, meaning those that lead to a significant different output when seen during training, in the lower tail of the loss distribution, a small loss can be achieved by feeding a point during training or by naturally easier-to-classify data points [4]. Analogously, the higher tail of the loss distribution can have non-members or members that are naturally harder to classify. Other MIAs look for high-risk samples among those that have a unique influence on the model's behavior regarding the target and related samples [16]. Such attacks do not require overfitting, as they exploit local vulnerability, pointing outliers, w.r.t. statistical properties of the dataset, as high-risk samples.

Even though overfitting is strongly related to membership risk, as it is a sufficient condition, Yeom et al. [32] claimed that it is not a necessary condition, by proposing an attack in which a sufficiently complex model leaks precise membership status by memorizing specific samples, increasing model leakage without significantly hurting the target model's performance. Similarly, Nguyen et al. [22] developed an attack that forced the model to leak precise membership status of a sample in a specific neuron's gradient. Its impressive performance raises awareness about how dangerous gradient sharing and privileged model access can be - conditions met in FL.

2.1.2. Attribute Inference Attacks. In an AIA, an attacker with access to the trained model attempts to infer the sensitive attribute value of a training sample [24]. These attacks usually *exploit the correlation between model performance and the sensitive attribute at the record level*, and can be further divided by the type of attribute being inferred. AIAs can target predictive features [12], [17], [27], [32], whose value can be somewhat encoded in the model's output or parameters, or non-predictive features [27], whose information may be intrinsically learned. Models implicitly learn to recognize sensitive features that are not part of the learning task. This phenomenon is referred to as overlearning and may be unavoidable [27], since it comes from the simplest model utility (predictions) and not even censoring techniques can fully prevent privacy leakage, which raises an urgent need for a balanced privacy-utility trade-off. Additionally, overlearning can be induced by active adversaries with privileged control over the target model (e.g. central server), for example by forcing the model to return an output vector with higher entropy for a sample with the positive attribute value [17].

Conceptually, AIAs are closely related to MIAs, as they can be viewed as an extra step of an MIA, i.e. to determine the sensitive attribute value of a member one can perform an MIA for every possible attribute value. In this case, the adversary infers the sensitive attribute value with which the target sample's membership probability was higher [32]. This relationship is theoretically supported by Salem et al. [25] who state that *MIAs and AIAs are mutually reducible*, meaning one can find an AIA that is as good as any MIA and vice-versa. Ultimately, the goal of studying this relationship is to improve defenses against these attacks, as *resilience against MIAs implies resilience against AIAs* [25].

2.1.3. Measuring attack performance. Attack evaluation should carefully *measure the extent to which model release helped adversaries infer private information*. In an MIA, since the adversary is most interested in determining members, controlling the number of misclassified non-members is far more important than controlling the amount of members classified as non-members, as reducing this error is trivial. Accuracy or precision are suitable but sensitive to class imbalance and depend on a cutting threshold (above which one classifies the target sample as a member) which must be optimized by the adversary [4]. The Area Under of Receiver Operating Characteristics curve (ROC AUC) is the probability that a classifier will be more confident predicting the positive label for a member than for a non-member, which is an adequate metric that overcomes the aforementioned limitations. Accounting for larger false positive rates (FPRs) can be a problem, so the curve can be restricted accordingly (restricted ROC AUC). Other metrics have been proposed, namely adversarial advantage [32] (TPR − FPR), which depends on the adversary's ability to find the threshold that maximizes the true positive rate (TPR) while minimizing FPR.

In its turn, an AIA succeeds if it correctly determines the sensitive value of training data records. Their evaluation is even more challenging, as these attacks often require stronger adversarial knowledge, namely prior knowledge about the sensitive attribute. As argued by Jayaraman et al. [12], AIA performance should be compared to data imputation[1] to determine the extent to which model release improved the adversary's ability to infer the sensitive attribute (attribute leakage).

2.2. Defenses

This overview of ML privacy highlights a trend regarding the literature on this topic: *there is a tendency to focus on developing attacks, leaving defenses under-explored*. Even though identifying new vulnerabilities is a relevant research topic, developing defense mechanisms to address those vulnerabilities is as important, particularly with adversaries that evolve faster than countermeasures.

2.2.1. Preventive Defenses. Most defenses found in the literature are preventive, as the goal of reducing adversarial advantage is part of the training process by default. Regularization techniques, such as dropout, L1 and L2-regularization, early stopping, and data augmentation, are among the most common against MIAs. These are frequently applied outside the scope of privacy protection to *decrease overfitting*. Albeit effective at improving generalization, these are frequently ineffective at protecting data privacy [9], [28], as they only slightly reduce attack performance. Additionally, Differential Privacy (DP) is a privacy-preserving technique that reduces a sample's contribution to the trained model by carefully adding noise while preserving statistical properties. Despite the powerful theoretical guarantees, DP is frequently associated with *high model utility loss* [9]. Furthermore, adversarial training is the most common defense against AIAs, which involves a combined loss function to reduce the main task error and the adversarial advantage simultaneously. The intuition behind this approach is to *penalize main task loss with adversarial advantage*. However, it can be ineffective [28] or incur high model utility loss [27].

All of these defenses are applied over the model's response/parameters regardless of the suspicion of any dishonest behavior. 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.

2.2.2. Reactive Defenses. Contrary to the aforementioned approaches, reactive defenses only mitigate attacks when an alarm for suspicious activity is triggered. Examples of this type of defense (attack detection) are common in the context of cybersecurity attacks [5], [11].

Regarding privacy attack detection, a very under-explored defense mechanism, a major distinction among these approaches is that they may target malicious users, by detecting malicious queries, or malicious participants/servers, by detecting malicious modifications to the target model. 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. Ko et al. [14] developed a passive attack detection mechanism to detect label-only MIAs and model inversion attacks through malicious query detection. A query is classified as malicious if its dissimilarity measure, against past queries, is below a preset threshold. The intuition is that *malicious queries tend to be semantically similar*, as attackers usually generate a sequence of queries that are slight variations of each other [6], [23], whereas benign queries are usually dissimilar since they typically represent a broader range of natural variations [14].

2.3. Limitations

This systematic review of prior work highlights that privacy attacks are often inadequately measured [4] and that the study of the connection between different attacks is mostly disregarded, as well as how realistic these attacks are, especially in FL settings (e.g. by requiring unrealistic access to individual gradients of all clients [21]). Moreover, attacks, namely MIAs, have been applied over extremely overfitted models [26], which are far away from the best models trained on those public datasets, which suggests that only overfitted models are vulnerable to attacks, or at least considerably more vulnerable. In other words, it is expected that *well-generalized models*[2] *may provide some level of security* against these threats. However, this may be a false sense of security as attacks that do not rely on overfitting (e.g. active MIA [22]) should be able to infer sensitive information regardless of the target model's generalization.

Furthermore, defense mechanisms are under-explored, often ineffective [22], or incur a high model utility loss to be effective [9]. Particularly, preventive defenses lead to a poor privacy-utility trade-off. Reactive defenses are far

1. Process of replacing missing features by leveraging prior knowledge about their relationship with the remaining known features.

2. Models that can generalize the learned relation between input features and main task label to unseen data (opposite of overfitted).

less studied, however, they may be the *key to reduce model utility loss*, as only suspicious behaviors are affected. In fact, there has been little research on active attacks due to their potentially strong impact on the target model, which would be easy to detect. However, there are no approaches to detect active privacy attacks (as far as the author's knowledge goes).

2.4. Contributions

To address the aforementioned limitations, the contributions of this paper are the development and evaluation of the following methods:

- A novel Active Attribute Inference attack (AAI*), an adaptation of AMI [22] to increase attribute leakage of well-generalized federated models.
- An adaptation of a state-of-the-art MIA to perform AIA (MIA2AIA), which is used as baseline for AAI*.
- A novel White-box Attack Detection and Mitigation (WADM*) based on abnormal changes in the weights distribution to reduce adversarial advantage while preserving model utility.
- Black-box Attack Detection methods based on Accuracy and ROC AUC (BADAcc and BADAUC) inspired by drift and anomaly detection, to serve as a baseline for WADM*.

Experiments are conducted on two real-world datasets, covering cases of non-IID and IID clients, showcasing the effectiveness and robustness of the novel attack and defense. By comprehensively exploiting under-explored vulnerabilities in FL with AAI* and developing WADM* to ensure an appropriate privacy-utility trade-off while defending against AAI*, this research contributes for a better understanding of privacy-preserving FL systems.

3. Novel Active Attacks

Privacy attacks are a significant threat, particularly relevant for systems thought to be privacy-preserving by default, such as FL systems. While it avoids data centralization for training, FL also introduces additional vulnerabilities compared to centralized learning. Participants are exposed to powerful active attacks that modify model parameters and exploit model complexity rather than relying on overfitting [22]. Precisely by not relying on overfitting, this threat model encompasses a larger range of realistic models. This section introduces and evaluates active AIAs aiming to demonstrate that well-generalized models carry a false sense of security, as they are exposed to these threats. The remainder of this section is structured as follows. Section 3.1 details adversarial goal, knowledge, and capabilities, Section 3.2 outlines the methodology of active attacks, and Sections 3.3 and 3.4 explain details of each proposed attack. Finally, Section 3.5 presents results of applying these attacks on two real-world datasets and Section 3.6 summarizes the contribution of this section.

3.1. Threat model

The proposed attacks are performed by the central server in an FL system composed of (presumably) honest participants, implying that the adversary can modify the global/target model, but cannot access individual gradient updates. Given incomplete knowledge about the target sample (i.e. the whole input vector is known except the sensitive attribute), the adversary aims to encode attribute information in model parameters. With these attacks, the adversary seeks to improve inference by forcing an increase of model leakage, instead of training data imputation models, that require a lot of data, to determine the sensitive (and missing) attribute.

For instance, if the adversary has access to the embedding representation of an image containing a person's face and additional information (e.g. if the person is young or has a big nose), then it can leverage its role on the FL system to infer the gender of the person. Besides inferring sensitive information, the adversary takes advantage of model leakage w.r.t the gender and determines the sample's membership status, as only target samples that are members will leave a footprint on gradients. Note that the assumption of having access to the embedding may seem unrealistic, but embeddings are commonly shared instead of real images, as they apparently protect privacy. The additional information can be obtained from other sources (e.g. social networks). Thus, this threat model poses a realistic privacy concern.

3.2. Attack Overview

Figure 1 illustrates the general methodology of these active attacks. On the top of this figure, a common FL system is represented in which several participants locally train a global model that 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 prevents the central server from receiving individual local models (also called gradient updates) by calculating an aggregate value through individual gradient contributions of each party with privacy guarantees [2]. These local models are all aggregated privately, such that the central server does not have access to individual updates, as illustrated by the Aggregation step node.

After aggregating, the newly updated global model is sent back to clients, thus completing a round. This process repeats until convergence or indefinitely (e.g. on streaming applications operating in periodic batch mode). At some point, the malicious central server performs an active attack in which it locally changes the global model to leak information, as illustrated at the bottom of this figure. Firstly, a neuron from the second fully connected (FC) 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 a shadow dataset, to leak membership or attribute information about a target sample. Then, it replaces benign weights/edges (from the first and second FC layer) by these maliciously trained weights/edges (in red). This global model is sent to clients which in turn send it back to the central server (after local training). Information leakage occurs in this step when the central server observes the difference (gradient update) between

the previous (malicious) and the current global model, particularly in the selected neuron (node filled in red). This type of attack exploits model complexity because if the dimension of the first two hidden layers is lower than the original input space, then the attack fails since these layers cannot encode more information than the original input space needed to encode membership/attribute information.

This attack can target several samples by selecting more than one neuron from the second FC layer. Due to the increase of the size of the adversarial network, time complexity scales quadratically with the number of target samples, but it is executed in powerful central servers, and since it does not require any additional communication among parties, there is no communication overhead.

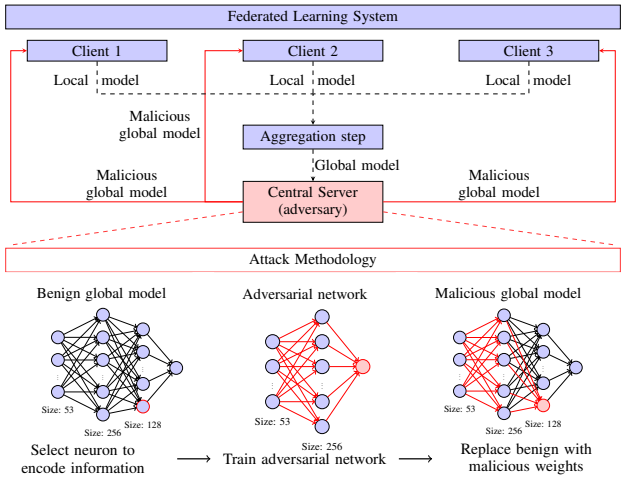

Figure 1. Overview of novel active attacks performed by the central server on a FL system.

3.3. MIA2AIA: Attribute Inference based on Membership Inference

Active Membership Inference (AMI) is an MIA, proposed by Nguyen et al. [22], that follows this methodology and classifies a target sample as a member if the maliciously modified neuron's gradient is not null. The key idea is to overfit the malicious neuron towards the target sample such that, if it passes through the model during training, then it will leave a distinguishable footprint on the gradient. AMI requires the target model to have at least 2 FC layers and use ReLU as the activation function, but since the adversary is the central server, model architecture can be tuned conveniently. Moreover, the adversary requires access to the target sample, although not to its label, and it must have a shadow dataset with which the neuron's weights are trained to be deactivated. Adversarial knowledge is limited to the shadow dataset, therefore any statistical information (e.g. distribution of an attribute) is calculated with this set, hopefully not significantly diverging from clients' data. However, the adversary *does not require access to individual gradient updates*, nor it imposes local training restrictions regarding the batch size and the number of local epochs. Due to these fair and attainable assumptions, AMI is a realistic attack, and, in contrast with some state-of-the-art approaches, it works regardless of the use of secure aggregation.

Building upon the state-of-the-art MIA proposed by Yeom et al. [22] (called AMI), this work proposes MIA2AIA, an active white-box AIA based on AMI, performed by the central server, which can be considered a metric-based attack[3], like AMI. In this case, the adversary has the whole input vector except the sensitive attribute, which it will try to infer by performing MIA2AIA. The intuition behind this attack is to *perform an 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. Although the MIA-based approach is from the state of the art, proposed by Yeom et al. [32], MIA2AIA is actually a new attack as it is based on a different - and much more powerful - MIA than previous AIAs sharing the same methodology.

The major difference between AMI and MIA2AIA methodology is in the inference phase, as 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, the attack itself fails, falling back to data imputation, i.e. the adversary uses its prior knowledge about the training data to estimate the attribute value. Therefore, the adversary predicts that the attribute value is the most frequent value in the shadow dataset. However, MIA2AIA performance is now based on AMI and also on *data imputation*, as the adversary may only accurately determine the sensitive value due to a highly imbalanced attribute distribution.

3.4. AAI*[4]: Active Attribute Inference

Following the same methodology, AAI* builds upon AMI to encode attribute information. 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 by analysing its sign. The selected neuron's weights are trained such that it takes a different value according to the sensitive attribute and such distinction must hold in the gradient update, enabling attribute inference. AAI* is a realistic attack for the same reasons as AMI, however instead of the ReLU activation function, the target model must use ELU with a negative parameter a. Additionally, the adversary needs the main task label of each target sample, contrary to MIA2AIA. If the label is not available, it could use the target model (ideally a sufficiently accurate one) to infer.

The major difference between AMI and AAI* is in the way this adversarial network is trained and, consequently, its inference phase. A modified neuron only leaks the

3. Attack in which the adversary uses the distribution of a metric (e.g. loss) to infer information based on a preset threshold.

4. The * superscript identifies novel methods proposed in this work.

target sample's attribute value if different logits/activations correspond to 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 values. Therefore, one must infer the target sample's attribute value based on the *sign* of gradient updates - positive, negative, and null.

The procedure of AAI* is an ML task that trains the malicious neuron to overfit the target sample such that it has a large and positive logit/activation when the target sample has a positive attribute value; negative and close to zero logit/activation when the target sample has a negative attribute value; and large but negative logit/activation when non-target samples pass through it. The next set of equations explains the reason behind these choices, where Eq. (1) show the derivative of ELU, and Eq. (2) how to encode a binary attribute through the gradient's sign.

$$\frac{\partial \text{ELU}}{\partial x} = \begin{cases} 1 & x > 0 \\ ae^x & x \leq 0 \end{cases} \tag{1}$$

Where x is the logit of the selected neuron i for sample s. Therefore, if $a < 0$

$$\frac{\partial \text{ELU}}{\partial x} = \begin{cases} > 0 & x > 0 \\ < 0 & x \leq 0 \\ \approx 0 & x \ll 0 \end{cases} \tag{2}$$

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 on the selected neuron's gradients does not harm attribute inference. However, they 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 the gradient computation from subsequent layers (which affect the malicious neurons' gradient) does not change the sign, by defining a condition on how to select the malicious neuron or how to change the weight connecting it to the output layer (Appendix A).

3.5. Evaluation and Analysis

To show that AAI* improves upon the most common state-of-the-art approach based on AMI (MIA2AIA), this section presents an evaluation of these attacks with two real-world datasets encompassing standard and challenging scenarios. The ultimate goal is to show that well-generalized models carry a false sense of security by being vulnerable to these novel active attacks. To ensure the robustness and reliability of findings, all experiments conducted in this study were repeated 32 times, and a confidence interval of 90% confidence was calculated, providing a rigorous statistical guarantee. Regarding the target sample selection, these experiments are balanced with respect to both membership and attribute value. The remainder of this section is structured as follows: setup details and evaluation metrics are explained in sections 3.5.1 and 3.5.2, respectively; empirical results are presented in sections 3.5.3 and 3.5.4, the impact of the sensitive attribute on these results is commented in Section 3.5.5 and the performance recovery after active attacks is presented in Section 3.5.6.

3.5.1. Setup. For generalization purposes, experiments are conducted on a mostly categorical and on another real-valued dataset. First, 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. CelebA is used for several tasks, namely attribute classification [15] and face recognition [14]. Among other well-known real-valued datasets, particularly in the image domain, that are frequently used to access attack performance, CelebA is the most suitable given that each image has a vector of 40 binary facial attributes (e.g. Attractive, Male, Young). Thus, the image paired with some of these annotations composes the input of federated models used throughout these experiments. Second, the dataset from the COP-MODE project [20] is comprised of approximately 65K permission requests, manually answered by 93 users which accepted 66% of them, captured from a runtime permission manager in Android [19]). To evaluate the robustness against non-IID cases, CelebA is randomly partitioned into IID clients and COP-MODE is partitioned by user leading to a case of non-IID clients.

To show that these attacks are effective against well-generalized models, models are trained on a simulated federated learning system in a single device (see Appendix B for more details). Specifically, CelebA models, trained to predict one of the annotations (Attractive), achieved an average ROC AUC around 88% and an accuracy of 79% on both training and testing sets. COP-MODE models, trained to predict user response [3], [19], achieved around 83% of ROC AUC and an accuracy of 77% on both training and testing sets. Furthermore, these attacks target several attributes to evaluate the effect of the sensitive attribute on attack performance. For this purpose, in each figure, the proportion of the most frequent value is indicated below (in the x-axis).

3.5.2. Evaluation Metrics. Active attacks are evaluated in two distinct ways: theoretical (Section 3.5.3) and practical (Section 3.5.4). TPR and FPR are used to measure the theoretical performance as these attacks are not threshold-dependent, since the nullity/sign of gradient updates does not depend on a threshold. Thus, TPR denotes correct neuron activation, meaning each target sample activates its corresponding neuron, and FPR denotes non-target samples that incorrectly activated the neuron. On a set of completely random samples, a dummy classifier would be accurate as many times as the expected number of samples with the majority value, but these experiments are balanced with respect to the attribute value. Therefore, TPR should be compared to 50% as a baseline.

Nevertheless, the comparison between this baseline and TPR unrealistically implies the assumption that FPR does not influence inference. However, an adversary may not be able to succeed due to the effect of false positives. An adversary conducting MIA2AIA must not expect the malicious neuron's gradient to be exactly zero every time the target sample does not have the corresponding attribute value or every time AAI* targets a non-training sample.

Instead, it must *allow for some variability*, since not only the target sample activates the malicious neuron, but also its neighborhood. Furthermore, since the adversary is looking to determine the missing value of a sensitive attribute, it is expecting that the target sample paired with the incorrect value belongs to the meaningless part of the feature space, thus it must be poorly represented in the training set, as well as its neighborhood. Therefore, the malicious neuron's gradient can be taken as the adversary's confidence in the target sample's membership status when paired with each possible attribute value, as illustrated in Appendix D. As a threshold-dependent inference phase (defining the allowed amount of variability), ROC AUC is used to measure the extent to which both gradient distributions are separated. To remove from the equation unacceptable FPR values, ROC AUC restricted to a maximum FPR of 25% is presented separately for members and non-members to bound attribute leakage of an ML model.

3.5.3. Theoretical evaluation. After proper hyperparameter tuning, Figures 2 and 3 illustrate TPR and FPR of single-target AAI⋆ and MIA2AIA, respectively for the CelebA and COP-MODE datasets, for several attributes (x-axis). Results from these figures show that both active attacks achieve great attack performance in terms of TPR. Particularly, TPR is nearly 100% for AAI⋆ and both versions of MIA2AIA[5] over COP-MODE models. Over CelebA models, AAI⋆ tends to outperform MIA2AIA, even more evidently when compared to MIA2AIA without data imputation, as the latter can be random guessing when targeting a few attributes.

Regarding FPR, conclusions differ according to the use-case but not to the target attribute. Over CelebA models, AAI⋆ significantly outperforms MIA2AIA, as FPR is nearly 0.0% for AAI⋆, versus an average close to 10% for MIA2AIA. Over COP-MODE models, FPR is always below 3.5% and MIA2AIA is statistically better than in previous experiments (CelebA) but not significantly better than AAI⋆ in COP-MODE experiments.

Regardless of the use-case, these experiments show that *AAI⋆ and MIA2AIA can infer sensitive information from well-generalized models* about their training data, always improving upon the baseline (blue dashed line) in terms of TPR.

3.5.4. Practical evaluation. An extensive set of experiments, to build a confidence interval of restricted ROC AUC, revealed that under gradient uncertainty (caused by false positives) these active attacks can pose a real threat to FL systems, namely through the AAI⋆ attack. Figures 4 and 5 present restricted ROC AUC of single-target AAI⋆ and MIA2AIA attacks over CelebA and COP-MODE models, respectively. Results from these figures show that AAI⋆ improves upon MIA2AIA regardless of the use-case and the degree of that improvement may vary according to the target attribute. On the members' set, differences are statistically significant when targeting every binary attribute from the CelebA dataset, as MIA2AIA is almost always random guessing (compared to the baseline of

5. The best version of MIA2AIA falls back to data imputation, while the other version (*MIA2AIA w/ imputation*) does not.

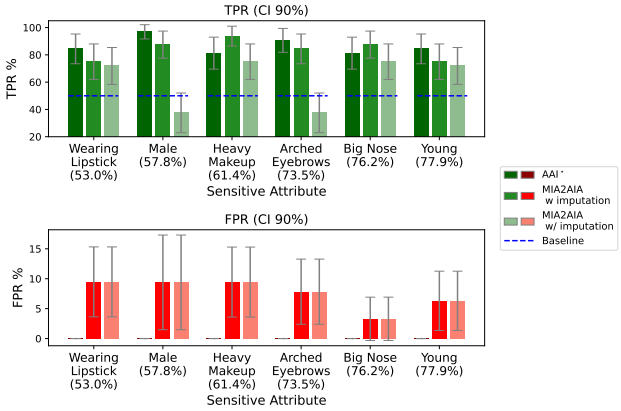

Figure 2. Single-target AAI⋆ and MIA2AIA TPR and FPR on CelebA models for several sensitive attributes.

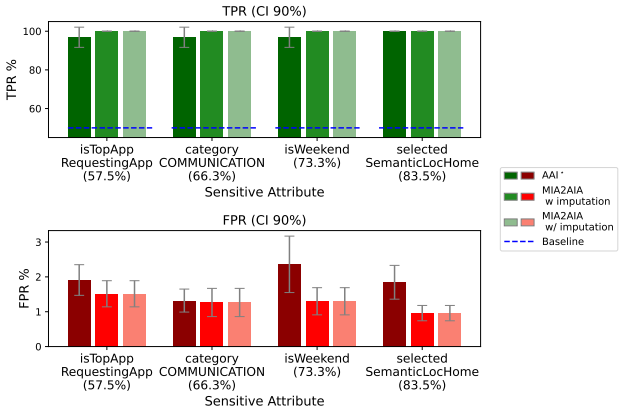

Figure 3. Single-target AAI⋆ and MIA2AIA TPR and FPR on COP-MODE models for several sensitive attributes.

50% in Figure 4). Remarkably, in this use-case, AAI⋆ can exceed an average of 90% of restricted ROC AUC, being always close to or above 80% for all attributes. On COP-MODE experiments, AAI⋆ tends to outperform MIA2AIA on the members' set, by reaching restricted ROC AUC above 70% but only for a few target attributes.

On the non-members' set, AAI⋆ is not significantly different from random guessing (the same holds for MIA2AIA), for all CelebA attributes. Compared to the respective performance on the members' set, *AAI⋆ can increase attribute leakage*, as it can leverage having access to the model to improve inference on the training set. However, in COP-MODE experiments, AAI⋆ on the members set can statistically improve on the non-members' set, but only slightly and for a single attribute (*isWeekend*).

The performance of AAI⋆ in this setting is attributed to its theoretical TPR (Figure 2), as its theoretical FPR is not significantly different from 0.0%, whereas MIA2AIA ineffectiveness is due to its excessive theoretical FPR which prevents the adversary from succeeding. But this error is not the only reason that explains the superiority of AAI⋆, as it is still able to improve upon MIA2AIA even when both register the same amount of error (e.g. over COP-MODE models). In fact, the effect of false positives differs per active attack, as in AAI⋆ some of these samples may cancel out the gradient of others, since false positives' gradient can have either a positive or a negative sign, as well as a smaller absolute value, according to Eq. 2.

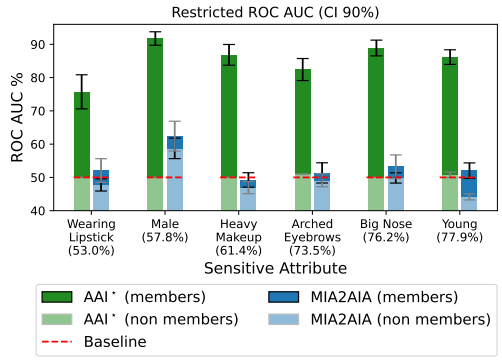

Figure 4. Single-target AAI* and MIA2AIA restricted ROC AUC on CelebA models for several sensitive attributes.

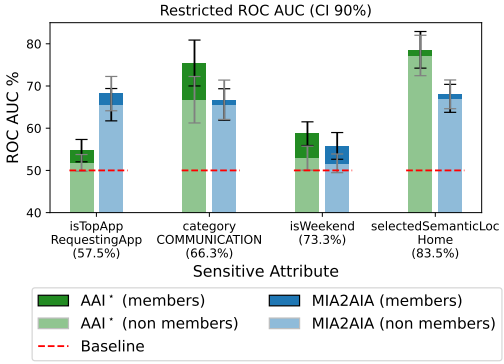

Figure 5. Single-target AAI* and MIA2AIA restricted ROC AUC on COP-MODE models for several sensitive attributes.

According to Yeom et al. [32] and judging by figures 4 and 5, MIA2AIA does not increase target model leakage regardless of its theoretical FPR, since it apparently cannot leverage having access to the target model to improve inference about training samples. Only AAI* can do so but depending on the amount of error. Additionally, the theoretical FPR registered on COP-MODE experiments coupled with the data augmentation technique used to train those models (explained in Appendix B) might be the reason why none of these attacks can increase model leakage about its training data, as both members and non-members' neighborhood is equally represented on the training set, besides the apparent similarity between training and testing distributions.

3.5.5. Impact of the sensitive attribute. Regarding the effect of the sensitive attribute, theoretical results suggest that, for a sufficiently small theoretical FPR (Figure 3), the distribution of the sensitive attribute does not play a role in theoretical TPR. Only for greater theoretical FPR values (Figure 2), the choice of the sensitive attribute may have an influence, judging by how MIA2AIA without data imputation behaves in some cases, as an attack that does not fall back to data imputation can be random guessing.

Nevertheless, the degree of imbalance of each attribute does not affect results monotonously, therefore other statistical characteristics, such as the relationship between the attribute and the rest of the input vector (assumed to be known by the adversary), may also have an impact. Practical results corroborate this insight, given that the same couple of theoretical TPR and FPR values yielded significantly different performances in terms of restricted ROC AUC. Particularly in COP-MODE experiments, AAI* that targets *isTopApp requestingApp* and *category COMMUNICATION* have the same theoretical FPR, however, ROC AUC is around 50% and 70%, respectively. Information gain, i.e. the reduction in entropy (degree of disorder), from knowing the input vector partially explains this discrepancy, as the greater the information gain the better the attack performance (correlation of 33%). The intuition behind this insight is that information gain measures how much the input vector tells about the target attribute. Thus, if information gain is large then the neighborhood of a given input vector will tend to have the same attribute value (which activates the malicious neuron correctly). Analogously, if information gain is small then there is some disorder in the target sample's neighborhood, hindering attribute inference.

3.5.6. Recovery from active attacks. Reduction of model utility loss is one of the pillars of the evaluation of defense mechanisms. However, if the target model cannot recover from active attacks, then there is no utility to preserve. This section aims to show that model performance is partially recoverable after a few training rounds, ultimately 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. This section also motivates the need for defense mechanisms that preserve model utility, as the latter is partially recoverable.

To evaluate model recovery, the attacked target model keeps training for a few training rounds. CelebA models fully recovered their performance on the train and test set, regardless of the targeted attribute, reaching close to 87% and 88% after AAI* and MIA2AIA, respectively, given the great performance and fast convergence of these models before any attack. However, COP-MODE models take longer to converge. Figure 6 shows their ROC AUC on the train set after training for 10 rounds after both attacks. Results show that ROC AUC approaches the baseline along the subsequent training rounds, converging significantly faster after MIA2AIA (in less than 3 rounds compared to at least 10 to fully recover from AAI*). Recovery on the test set, presented in Appendix E along with further explanations for this discrepancy, reveals that recovery on this set is significantly harder. Hence, models can only partially recover from active attacks.

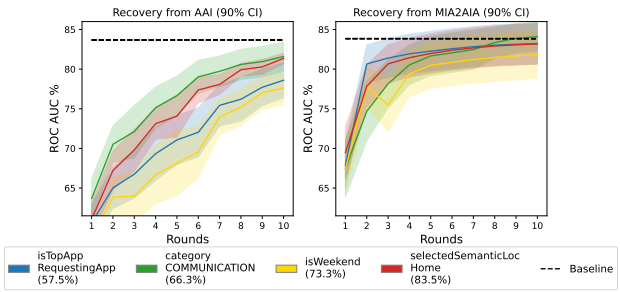

Figure 6. Recovery (ROC AUC evolution) from single-target AAI* and MIA2AIA on the train set over 10 training rounds. The baseline stands for the average ROC AUC before any attack.

3.6. Summary

This section introduced novel active attribute inference attacks, namely AAI* that outperforms state-of-the-art alternatives (MIA2AIA) by reducing FPR in terms of neuron activation, increasing its performance on a practical setting and improving attribute leakage, namely when the number of false positives is nearly 0%. Both attacks are equally strong in terms of TPR on both datasets, but MIA2AIA without data imputation (to deal with inconclusive attacks) can be ineffective. Particularly over CelebA models, TPR is often over 80% on average, and over COP-MODE models is not significantly different from 100%. The effectiveness on both use-cases shows that these attacks are effective on challenging settings (i.e. non-IID clients). Furthermore, results from Appendix C show that these novel active attacks are scalable, meaning they can target more than a single sample, but only up to point, with AAI* significantly improving on MIA2AIA.

Even though the relationship between the theoretical and practical evaluation is clear - the overlap between the gradient distribution of positive and negative samples (w.r.t. the sensitive attribute) depends on FPR. What is not so clear is the amount of FPR that still allows for a sufficiently accurate inference. In this setting, AAI* improves upon MIA2AIA, by achieving over 80% of restricted ROC AUC, due to incurring significantly less FPR. Even when FPR is slightly higher on AAI* (or statistically the same), the adversary can still improve upon MIA2AIA.

A study about the impact of active attacks on model performance and about recovery from this impact shows that it is possible to partially recover model performance after a few training rounds, ultimately demonstrating that *active attacks may occur in FL systems without major long-term model utility loss*.

These results suggest that traditional countermeasures (to improve generalization) do not mitigate these novel active attacks, as they do not rely on overfitting. Thus, effective defense mechanisms are needed, ideally reducing model utility loss, as it is partially recoverable.

4. Attack Detection Mechanisms

From the previous section, it is clear that these novel active attacks pose a real threat to FL systems. However, these are not being addressed by the community as no effective defense mechanism has been proposed. Particularly, Local Differential Privacy (LDP), a decentralized approach of adding noise to ensure standard differential privacy guarantees [30], is not effective against AMI [22], since the adversary can adapt the methodology to account for perturbed versions of the target sample, therefore evading the defense. Even LDP based on gradient clipping is ineffective against these attacks because the gradient of a neuron that is only activated by a single training sample, and its close neighbourhood, is small in absolute value, hence it is not clipped. Furthermore, strategies to improve model generalization are not expected to defend against attacks that do not exploit overfitting. Instead, detecting and responding to ongoing attacks presents a viable alternative, especially since the performance is partially recoverable after a few training rounds (Appendix E), therefore minimizing model utility loss. The remainder of this section is structured as follows: Section 4.1 outlines the methodology behind the proposed and baseline defenses, Sections 4.2 and 4.3 further detail each strategy, and their efficiency is commented in Section 4.4. Finally, Section 4.5 presents empirical results to validate the effectiveness of these approaches and Section 4.6 summarizes the contribution of this section.

4.1. Defense Overview

Figure 7 illustrates the general methodology of defense mechanisms applied by each client in a FL system. The top of this figure represents a common FL system in which the role of the central server is to join the aggregated contribution from each training round to the global model from the previous round, sending it back to clients to start another training round. At some point, the central server performs an active attack (Figure 1), illustrated by the red arrows connecting the central server to each client, which receives a malicious global model. However, each client must apply a defense mechanism locally, since it cannot trust the central server to collaboratively defend against these attacks. The bottom of this figure explains how these strategies work according to the knowledge of the defender (black-box or white-box). Each defense is divided into a detection phase (indicated by a yellow triangle) and a mitigation phase (indicated by a red triangle).

A defender with black-box access can detect active attacks by comparing train or test performance using the current and the previous model received from the central server. The most limited type of black-box access, i.e. access to the predicted class only, allows to compare accuracy, whereas access to the output vector allows to compare ROC AUC. On the other hand, if the defender has white-box access, i.e. global model parameters are accessible, then it can detect maliciously modified neurons (used to leak precise attribute information) by comparing the current and the previous weights distribution of each neuron. In this figure, the 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, as explained in Appendix G.1. Note that these detection strategies are employed before local training starts at the beginning of a new round, as locally training a malicious model can reduce the probability of detecting an ongoing attack by reducing the drop of evaluation metrics, or the distortion of weights distribution. 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.

4.2. Black-box Attack Detection Strategies

Black-box attack detection strategies are based on well-known drift detection and statistical control algorithms. Even though the approach is not new, these are, for the first time, applied in this context and are used as a baseline for the novel defense mechanism (proposed in Section 4.3). As strategies based on model performance, data heterogeneity, referred to as non-IID clients and main task imbalance play an important role, namely the latter

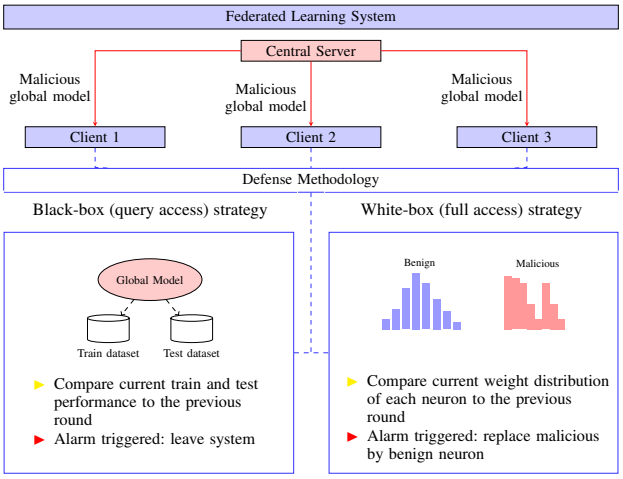

Figure 7. Overview of defense mechanisms applied by each client on a FL system.

determines the best set to monitor (i.e. train or test set). The way data imbalance affects attack detection varies per strategy, which is explained throughout this section, whereas the effect of data heterogeneity is empirically studied in Section 4.5.

4.2.1. BADAcc: Black-box Attack Detection based on Accuracy. Inspired by Statistical Process Control (SPC) [7], BADAcc is a statistical-based approach that uses the 3 standard deviations (3-sigma) rule and *detects an ongoing attack if local accuracy drops more than expected*, meaning 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. The intuition is that under an active attack, model behavior is perturbed and the error increases significantly, resulting in a significant accuracy drop.

Other than working under the most restricted model access (from the defender side), one of the advantages of this approach is that it guarantees statistical properties, by detecting abnormal drops in accuracy using confidence intervals. Moreover, its precision can be adjusted as needed, for example reducing the in-control margin from 3 standard deviations to 2 (approximately 95% confidence). However, this strategy is user-dependent, as the dataset size and local accuracy from previous rounds may influence attack detection, other than a possibly different precision level per user. Results presented in Section 4.2.2 are from monitoring train performance, as BADAcc can be less effective by monitoring test performance, given that this set can have class imbalance, thus leading to a higher baseline for accuracy (of a random classifier) without the model being necessarily accurate.

4.2.2. BADAUC: Black-box Attack Detection based on ROC AUC. Similarly to BADAcc, BADAUC is a distance-based approach that identifies a *significant perturbation to the expected convergence* of ROC AUC, specifically if the difference (in absolute value) between the current and previous round is greater than a preset threshold. The sensitivity of the baseline of accuracy to class imbalance and the fact that it is threshold-dependent are the reasons behind the choice to monitor ROC AUC. Particularly, accuracy can hide a significant drop in model performance because the baseline is too high (on class imbalance problems), or it can increase false alarms due to an inadequate threshold choice.

This approach works under restricted model access, but it is threshold-dependent. This hyperparameter may be tuned, previously or adaptively during training, according to the use-case. 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, 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. For comparison purposes, BADAUC monitors train performance, but it could be applied to the test set.

4.3. WADM*: White-box Attack Detection and Mitigation

Black-box attack detection mechanisms may result in significant false alarms, especially for non-IID clients. To avoid this user-dependency and improve robustness, an alternative is to use the global model's weights, as these are a product of the aggregation of the updates from all participants. Additionally, a user leaving the system after detecting an ongoing attack incurs the highest model utility loss, despite effectively protecting users' privacy. WADM* emerges as an alternative to BADAcc and BADAUC to overcome the aforementioned issues by identifying malicious neurons and changing their distribution instead of abandoning the training process.

The intuition behind this approach is to *detect significant distortions of the weights distribution* between consecutive training rounds. WADM* is divided into two phases: detection and mitigation. In the detection phase, KL-Divergence is used to measure the difference between the current and previous weights distribution of each neuron, as in Equation (3).

$$\text{KL-Div}(X_{prev}, X_{cur}) = \sum_i X_{prev} \log\left(\frac{X_{prev}}{X_{cur}}\right), \quad (3)$$

where i ranges across the number of bins used to represent both distributions. Note that to account for distribution shift (as illustrated in Figure 22 from Appendix G.1), 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, therefore being a distance-based approach. From equation (3), one can see that if a bin from the distribution X_{cur} has 0 frequency, then KL-Divergence is infinite. Besides taking the minimum between both directions, this can be avoided by choosing an adequate number of bins to represent the weights distribution, possibly reducing false alarms. Therefore, the detection phase depends both on the threshold and the number of bins. In the mitigation phase, the defender

changes the current weights to the previous weights of any neuron flagged as malicious, discarding the most recent ones. A benign neuron keeps its weights and the previous ones are updated.

This strategy depends on a threshold whose tuning depends on the use-case and, similarly to BADAUC, may be set a priori or change throughout the training process according to local information. Moreover, users may have different concepts of sensitive data, therefore WADM* can 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.

4.4. Efficiency of defense mechanisms

The application of these defenses on a practical scenario, and in every training round, raises concerns about the consumption of computational resources. The computational complexity of WADM* is linear with the size of the second FC layer, whereas black-box strategies are constant with respect to the same variable. Since they are performed locally, meaning that they do not depend on communication among parties, there is no additional communication overhead. Moreover, the spatial complexity of WADM* is linear with respect to the size of the second FC layer, since it requires the storage of this layer. When this requirement is not met, instead of storing the complete distribution of the weights, the defender can 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. In general, these defenses are efficient, even though WADM* requires a higher computational capacity which is compensated by the expected improvement, therefore balancing the efficiency-efficacy trade-off.

4.5. Evaluation and Analysis

To show that WADM* improves upon black-box strategies, commonly applied on predictive systems for a variety of purposes, this section presents results for BADAcc, BADAUC, and WADM* applied to the same couple of datasets with which attacks were previously tested. Similarly, to ensure the robustness and reliability of findings, all experiments conducted in this study were repeated 32 times, and a confidence interval of 90% was calculated, providing a rigorous statistical guarantee. Regarding the target sample selection, these experiments are balanced w.r.t both membership and attribute value. It is important to note that, contrary to black-box strategies, WADM* is evaluated per training round, as it does not force the client to leave the system. Thus, results presented below are from applying WADM* on the training round in which an attack occurs, latter WADM* is applied on early training rounds (Section 4.5.4). Results of BADAUC and WADM* assume an optimal threshold choice, meaning one that minimizes $\frac{\text{TPR}}{\text{FPR} + \text{Missed Attacks}}$.

The remainder of this section is organized as follows: Section 4.5.1 explains evaluation metrics, Section 4.5.2 and 4.5.6 presents detection and mitigation results, Section 4.5.3 comments about the user dependency, Section 4.5.6 and 4.5.5 presents false alarms triggered by WADM* on early training rounds and under a threshold-selection policy that minimizes FPR, respectively.

4.5.1. Evaluation metrics. To access the performance of the detection phase, one must use TPR (detection of an ongoing attack) and FPR (false alarm, i.e. a benign round classified as malign). Additionally, Missed Attacks stands for the probability of failing to detect an ongoing attack, and it is such that summing it with TPR and FPR yields 100% for black-box strategies. For this reason, it is omitted from figures but it is relevant to study user dependency in Section 4.5.3. For black-box strategies, these metrics are evaluated in the domain of users, i.e. each user will contribute exclusively to a single metric, either to TPR, FPR, or Missed Attacks.

Given that WADM* detects malicious neurons, its TPR, denoted as "WADM* (neuron)", stands for the probability of successfully detecting a malicious neuron and FPR stands for the rate of benign neurons that are misclassified. However, detection of MIA2AIA is successful if at least one neuron is detected[6], therefore "WADM* (attack)" is evaluated instead, and it stands for the probability of at least one neuron being detected (per attack). This metric allows a fair comparison between black and white-box strategies by evaluating all defenses in the domain of the attack. Furthermore, the theoretical attack performance (TPR and FPR) after the mitigation phase is used to evaluate the mitigation phase of WADM*. However, this evaluation is irrelevant for black-box strategies, since the client leaves the system as soon as it detects the attack, thus preventing leaky gradients from being shared with the central server.

4.5.2. Attack Detection. Figures 8 and 9 illustrate the detection rate (TPR) of all three defense mechanisms, applied against single-target AAI* and MIA2AIA, on CelebA and COP-MODE models, respectively. These results show that all attack detection mechanisms effectively detect both active attacks, as TPR is always significantly better than the baseline (50%). Additionally, attack detection does not depend on the targeted attribute, which suggests that these *active attacks greatly impact accuracy/ROC AUC and weights distribution regardless of the attack performance*.

Regarding black-box strategies, BADAUC is significantly better than BADAcc, particularly when applied to CelebA models, as TPR is always 100%. This discrepancy is a consequence of the threshold-dependency of accuracy. When applied to COP-MODE models, BADAUC detects both active attacks about 90% of the time. Nevertheless, close to 85%, or more, of the users detect AAI* with BADAcc, regardless of the use-case. However against MIA2AIA, the performance of BADAcc depends on the use-case, as it can detect close to 100% on CebelA models, but only up to 80% on COP-MODE models. This divergence is explained by the stability of model convergence on early training rounds, as explored in Section 4.5.3.

In its turn, WADM* detects almost all active attacks, as TPR is not significantly different from 100%, regardless of the attack and use-case. Particularly when applied

6. The adversary would notice an excessively high gradient, sign that FPR increased considerably, disabling any inference.

to COP-MODE models, WADM* is the best detection strategy, considering that it significantly improves upon the best black-box strategy, and, on CelebA models, it is as effective as BADAUC. However, WADM* can detect neurons used for AAI* significantly better than those used for MIA2AIA, meaning that the distortion of the weights distribution is greater when performing AAI*. However, this behavior might also be explained by the threshold choice, as it depends on the benign evolution of the weights distribution, which determines FPR and conditions the optimal threshold selection.

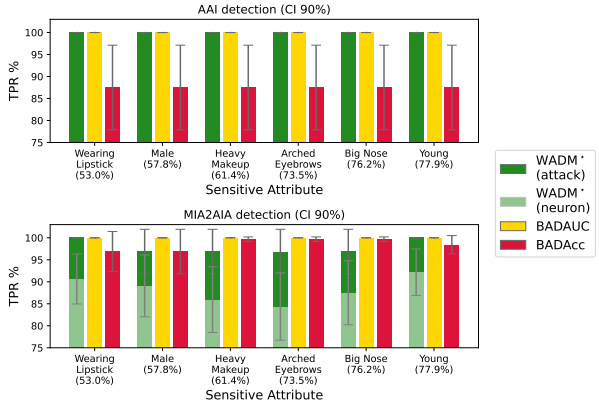

Figure 8. Detection of single-target AAI* and MIA2AIA over CelebA models (that target several attributes) with several attack detection strategies.

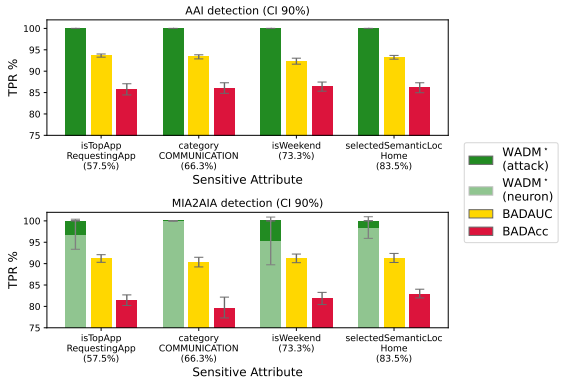

Figure 9. Detection of single-target AAI* and MIA2AIA over COP-MODE models (that target several attributes) with several attack detection strategies.

These results show that both WADM* and BADAUC are the best defense mechanisms in terms of detection rate. However, the overall conclusion must take into account false alarms (FPR) and their cost, which is explored throughout the rest of this section.

4.5.3. User dependency of BADAcc and BADAUC. Black-box strategies rely on the local monitorization of model performance to detect ongoing attacks, thus it is expected that local features play an important role in attack detection, especially local accuracy/ROC AUC (prior to the attack) and training dataset size.

Experiments on COP-MODE models show that the training dataset size is positively correlated with the probability of incurring a false alarm (FPR), specifically 12.0% and 10.6% depending on the type of model, as the safety margin (determined by standard deviation) becomes narrower as the local training dataset becomes larger. Therefore, any small decrease in accuracy may fall out and be incorrectly classified as an ongoing attack. Although, this correlation is small and does not fully explain BADAcc FPR. On the other hand, the dataset size is not correlated with the FPR of BADAUC. Instead, local ROC AUC before the attack is negatively correlated with BADAUC FPR, specifically -47.7% and -41.1%, given that the smaller the previous ROC AUC the lower the difference between a benign and malicious situation. In fact, the FPR of both strategies is mainly caused by a few users that are considerably more prone to that error, and the reason is attributed to an abnormally unstable evolution of the evaluation metric on early training rounds (hopefully before any attack).

As for the probability of missing an ongoing attack, local accuracy/ROC AUC before the attack is strongly and negatively correlated with this error (over -50%). Given that accuracy/ROC AUC drops to a random guessing level, it becomes natural that the greater the local accuracy/ROC AUC the greater the drop, therefore the easier the detection. Experiments on CelebA models revealed that FPR is 0.0% for both black-box strategies, but Missed Attacks, particularly AAI*, can be close to 13% on average with BADAcc. This observation is explained by the cutting threshold (to compute accuracy) and its inadequacy in certain situations, hiding an abnormal drop in accuracy.

In general, average model performance tends to converge and have a stable evolution. However, in a practical scenario, some users may have a sufficiently different data distribution to make their local performance less stable or accurate, i.e. clients are non-IID (e.g. COP-MODE). Hence, *user dependency is expected to impact black-box strategies in a real-world case*. This insight is empirically supported by these results, as users play an important role in COP-MODE experiments but not in CelebA, as in the latter case users are IID.

4.5.4. False alarms on early training rounds. Along the training process, and particularly on early training rounds, model parameters are expected to change considerably. Thus, the defender must distinguish benign from malicious neurons while the model converges to reduce the impact of WADM* on the training process. As a threshold-dependent strategy, WADM* triggers an alarm every time KL-DIVERGENCE exceeds this threshold. Therefore, the key to balancing FPR and TPR relies on fine-tuning this parameter. To show that *it is possible to adapt the threshold along the training process without affecting TPR significantly*, Figure 10 shows WADM* FPR on early training rounds (before any attack)[7]. This figure shows that an adequate threshold choice yields a small amount of FPR. Particularly, a selection policy that gradually decreases the threshold and yields 0.1556 as the one to use after 5 training rounds (when the attack occurs) simultaneously reduces FPR and matches the optimal selection policy as presented in the next section 4.5.5.

Furthermore, as training evolves so does FPR to the point of it becoming 0.0%, or within a negligible range,

7. Note that these experiments are restricted to COP-MODE, as CelebA models only trained for 2 rounds (before the attack), therefore not allowing for a proper study of the same evolution.

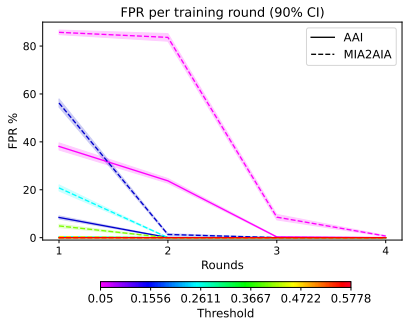

Figure 10. WADM⋆ FPR evolution along the first benign training rounds of COP-MODE models used for AAI⋆ and MIA2AIA.

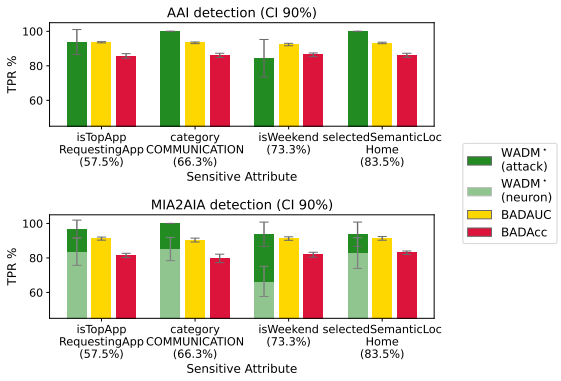

Figure 11. Detection of single-target AAI⋆ and MIA2AIA over COP-MODE models with several attack detection strategies (0.1556 was the threshold used for both cases by WADM⋆).

after a few rounds. Additionally, WADM⋆ incurs a significantly lower number of false alarms when the model is used for AAI⋆ compared to models used for MIA2AIA, at least for smaller thresholds. This observation is explained by the fact that the former type of models (used for AAI⋆) converged faster than the latter, which was observed during the experimental phase of this work.

Tuning this threshold depends on the use-case, as well as on the target model's architecture, therefore a methodology to fine-tune this threshold for several use-cases is left as future work.

4.5.5. Attack detection under FPR minimization. Previous results were obtained assuming an optimal threshold choice to maximize $\frac{\text{TPR}}{\text{FPR} + \text{Missed Attacks}}$, knowing that FPR was under control on the training round in which the attack occurred, as the model converged to a stable state. A suitable solution to reduce FPR on early training rounds is to adapt the threshold along the training process, as explained in Section 4.5.4[8]. However, a good selection policy not only reduces false alarms on early training rounds (Figure 10) but also must match the TPR of other selection policies, such as the optimal one (Figure 9).

Figure 11 illustrates attack detection for a fixed threshold (0.1556) according to the supra-mentioned selection policy (in Section 4.5.4). The comparison between this figure and Figure 9 show that both policies match in terms of attack detection. Although, regarding the detection of malicious neurons from MIA2AIA, WADM⋆ can be significantly worse, with this threshold, against attacks that target *isWeekend* (AAI⋆ detection is also statistically worst in this case but for a small margin). However, to mitigate MIA2AIA, one only needs to detect and mitigate one of the couple of neurons, which is measured by the "WADM⋆ (attack)" metric. Thus, besides the fact that significantly more neurons from MIA2AIA pass by undetected, attack detection results still match those obtained with the optimal threshold. All in all, these experiments show that *even under false alarm minimization on early training rounds, WADM⋆ effectively detects both active attacks*, while improving upon black-box strategies by improving FPR and TPR (in some cases).

4.5.6. Attack Mitigation. Given that WADM⋆ successfully detects malicious neurons used for these active attacks (Figures 8 and 9), one must expect that both AAI⋆ and MIA2AIA are reduced to random guessing. Figures 12 and 13 illustrate attack performance after the application of WADM⋆ over AAI⋆ and MIA2AIA targeting CelebA and COP-MODE models, respectively, meaning after detecting and switching the malicious neuron's weights. These figures show that both AAI⋆ and MIA2AIA are reduced to random guessing (or even less than that), in terms of theoretical TPR, regardless of the use-case. Besides that, *FPR increases considerably* (namely for AAI⋆ which is over 90%). As expected, with this amount of FPR the adversary is no longer able to overcome its effect on the gradient. For example, AAI⋆ and MIA2AIA over COP-MODE models achieve $[49.73\%, 55.08\%]$ and $[48.81\%, 52.86\%]$ of restricted ROC AUC on members when trying to target *selectedSemanticLoc Home*.

4.6. Summary

This section introduced three attack detection mechanisms, namely WADM⋆ that outperforms standard monitorization of model performance, i.e. black-box strategies, by improving attack detection and successfully mitigating both attacks while preserving model utility loss. Particularly, it can detect malicious neurons perfectly (100%), namely the ones used for AAI⋆, and the detection of an ongoing MIA2AIA is also nearly perfect. Regarding black-box strategies, BADAUC significantly outperforms BADAcc, which is clearer in the detection of AAI⋆ on CelebA models, for which BADAcc missed close to 13% of attacks on average. But the greatest improvement upon prior work is in terms of FPR, as WADM⋆ incurs nearly 0% whereas BADAcc and BADAUC incurred close to 8% and 3% on COP-MODE models respectively, which showcases the superiority of WADM⋆ on challenging settings (i.e. non-IID clients). Particularly, false alarms of black-box strategies are mainly due to user-specific characteristics,i.e. training dataset size and local performance before any attack. Remarkably, WADM⋆ FPR is not user-dependent and, on early training rounds, can be reduced to a negligible range if the threshold is properly tuned to the use-case and adapted along the training process, without significantly affecting TPR.

8. Similarly to Section 4.5.4, this study is restricted to COP-MODE, as CelebA models converged too fast.

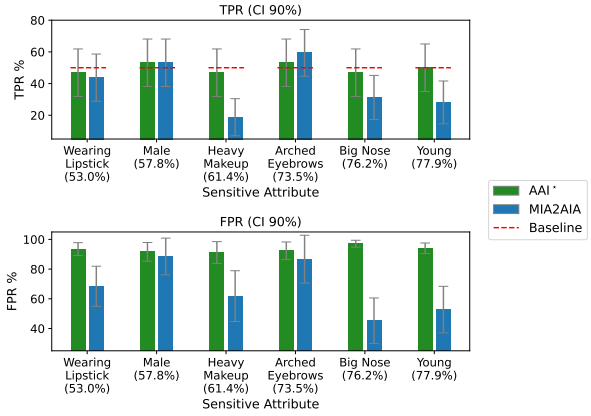

Figure 12. Single-target AAI and MIA2AIA attack performance after WADM* mitigation over Celeba models, for several attributes.

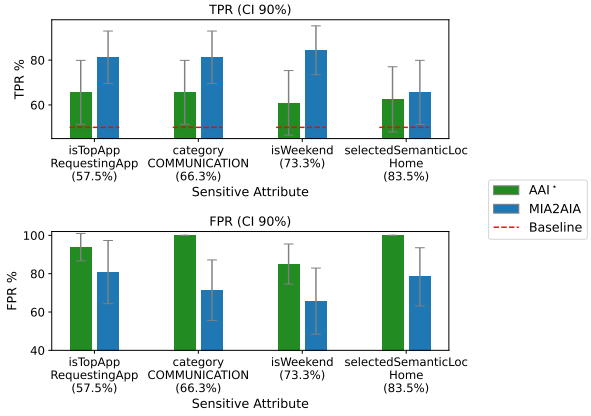

Figure 13. Single-target AAI and MIA2AIA attack performance after WADM* mitigation over COP-MODE models, for several attributes.

Regarding attack mitigation, black-box strategies force the client to leave the system, whereas WADM* effectively reduces any attack to random guessing without leaving the system. Although with a black-box strategy, a client may use the previous *box*/model (for predictions only) until it has the guarantee that the newly received *box* is benign, therefore reducing the excessive cost of FPR. However, besides preserving the ability to train, WADM* improves upon this baseline by reducing user dependency, as every client receives the same global model, even though each user can set its own threshold.

5. Conclusion

This work introduced novel active attribute inference attacks, namely AAI* that outperforms state-of-the-art alternatives based on membership inference (MIA2AIA) by reducing FPR (in terms of neuron activation). More importantly, AAI* improves upon prior work particularly in a practical setting, where it overcomes the effect of false positives and increases attribute leakage. Both theoretical (neuron activation) and practical (gradient uncertainty) evaluation methodologies target a limitation of prior work by adequately measuring attack performance and bounding model leakage. This study shows that *well-generalized models are vulnerable to these active attacks*, contrasting with past research that mainly targets extremely overfitted models [26]. These attacks pose a real threat to FL systems, as the adversary does not require unrealistic conditions to succeed and model performance is partially recoverable in a few training rounds. This study suggests that traditional countermeasures (to improve generalization) do not mitigate them, as they do not rely on overfitting.

Furthermore, this work introduced attack detection mechanisms that effectively, and efficiently, detect and mitigate both active attacks. The effectiveness of black-box strategies showcases that these attacks are easily detected while monitoring evaluation metrics on the training set. However, some users may systematically be forced to leave the system in a benign situation, while others may constantly be exposed to both attacks, as a consequence of false alarms and missed attacks, respectively. A novel defense mechanism based on the detection of abnormal distortions of the weights distribution (WADM*) significantly reduces false alarms (both in probability and respective cost), while matching the TPR of other defense mechanisms. Notably, WADM* reduces user dependency by monitoring model parameters, instead of model performance, which is critical when clients are non-IID. Most importantly, WADM* overcomes a major state-of-the-art limitation - *reduced model utility loss*. By switching malicious neurons' weights by the previous set of weights, a client can reduce any active attack to a random guess.

5.1. Limitations and Future Work

From this work, some relevant future directions arise, namely improving the scalability of these attacks and how to balance attack performance and interference on model parameters through subtle parameter changes. This could lead to an adversary that can evade detection by reducing the distortion of the weights distribution and the drop in model performance. Moreover, performance monitorization is a common practice, outside the scope of attack detection, and abnormal drops may not necessarily be attributed to active privacy attacks. Thus, BADAcc and BADAUC would most likely require additional steps to distinguish benign model performance reduction from a malign one. Similarly, further research is needed to fine-tune the threshold of WADM* to minimize FPR while preserving efficacy, for several use-cases. Furthermore, according to this research, these defenses may be considered attack agnostic, as they are successful against several active attacks, although, it needs to be validated in future work. Moreover, as these are applied in every training round, further research may be needed to reduce the consumption of computational resources.

Acknowledgment

This work is financed by National Funds through the Portuguese funding agency, FCT - Fundação para a Ciência e a Tecnologia, within project LA/P/0063/2020.DOI 10.54499/LA/P/0063/2020 | https://doi.org/10.54499/LA/P/0063/2020

References

[1] General data protection regulation GDPR. Accessed: April 2024.

[2] 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.

[3] 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.

[4] 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.

[5] Congcong Chen, Lifei Wei, Lei Zhang, Ya Peng, and Jianting Ning. Deepguard: Backdoor attack detection and identification schemes in privacy-preserving deep neural networks. *Security and Communication Networks*, 2022(1):2985308, 2022.

[6] Christopher A. Choquette-Choo, Florian Tramer, 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.

[7] João Gama, Pedro Medas, Gladys Castillo, and Pedro Pereira Rodrigues. Learning with drift detection. In *Brazilian Symposium on Artificial Intelligence*, 2004.

[8] Catarina Gomes. Code for Active Attribute Inference Against Well-Generalized Models In Federated Learning, April 2025. https://github.com/Ana-Catarina-Gomes/Active-Attribute-Inference-Against-Well-Generalized-Models-In-Federated-Learning.

[9] 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.

[10] 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.

[11] 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.

[12] 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.

[13] Jinyuan Jia and Neil Zhenqiang Gong. AttriGuard: A practical defense against attribute inference attacks via adversarial machine learning. In *USENIX Security Symposium*, 2018.

[14] 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.

[15] 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.

[16] Yunhui Long, Vincent Bindschaedler, Lei Wang, Diyue Bu, Xiaofeng Wang, Haixu Tang, Carl A. Gunter, and Kai Chen. Understanding membership inferences on well-generalized learning models. *arXiv preprint arXiv:1802.04889*, abs/1802.04889, 2018.

[17] Mohammad Malekzadeh, Anastasia Borovykh, and Deniz Gündüz. Honest-but-curious nets: Sensitive attributes of private inputs can be secretly coded into the classifiers' outputs. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, CCS '21, page 825–844, New York, NY, USA, 2021. Association for Computing Machinery.

[18] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera 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.

[19] 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.

[20] Ricardo Mendes, Mariana Cunha, João P. Vilela, and Alastair Beresford. The project cop-mode, 2020. Accessed: April 2024.

[21] 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.

[22] 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.

[23] 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.

[24] Maria Rigaki and Sebastian Garcia. A survey of privacy attacks in machine learning. *ACM Comput. Surv.*, 56(4), nov 2023.

[25] 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.

[26] 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.

[27] Congzheng Song and Vitaly Shmatikov. Overlearning reveals sensitive attributes. In *Proceedings of the 8th International Conference on Learning Representations (ICLR '20)*, 2020.

[28] 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.

[29] Jeffrey G. Wang, Jason Wang, Marvin Li, and Seth Neel. Pandora's white-box: Precise training data detection and extraction in large language models. *arXiv preprint arXiv:2402.17012*, 2024.

[30] 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.

[31] 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.

[32] 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.

[33] 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.

Appendix A.
Theory behind AAI*

Section 3.4 explained how the adversary modifies the malicious neuron, however no guarantees were given regarding whether the gradient computation preserves the information encoded by the activation. To establish some notation, Figure A 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 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 far away as possible from 0) otherwise. Additionally, Equation (4) shows how to compute the gradient (g_k) of the malicious neuron, selected from the second fully connected layer, as illustrated in Figure A, using the chain rule. This expression assumes the network has the same architecture as the one used for COP-MODE experiments. Equations (6), (8), and (10) present each part of the expression from Equation (4). Finally, Equation (11) shows the gradient computation, obtained by replacing each fraction of Equation (4) by its respective expression.

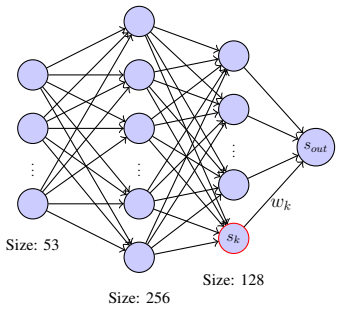

Figure 14. Illustration of the target model with annotations.

$$\frac{\partial \text{BCELoss}(\hat{y})}{\partial s_k} = \frac{\partial \text{BCELoss}(\hat{y})}{\partial \hat{y}} * \frac{\partial \hat{y}}{\partial s_{out}} * \frac{\partial s_{out}}{\partial s_k} \quad (4)$$

$$\text{BCELoss}(\hat{y}) = \begin{cases} log(\hat{y}) & y = 1 \\ log(1-\hat{y}) & y = 0 \end{cases} \quad (5)$$

$$\frac{\partial \text{BCELoss}(\hat{y})}{\partial \hat{y}} = \begin{cases} \frac{1}{\hat{y}} & y = 1 \\ \frac{1}{1-\hat{y}} & y = 0 \end{cases} \quad (6)$$

$$\hat{y} = \text{SIGMOID}(s_{out}) = \frac{1}{1 + e^{-s_{out}}} \quad (7)$$

$$\frac{\partial \hat{y}}{\partial s_{out}} = \hat{y} * (1 - \hat{y}) \quad (8)$$

$$s_{out} = \sum_i^{64} \text{ELU}(s_i) * w_i \quad (9)$$

$$\frac{\partial s_{out}}{\partial s_k} = w_k * \begin{cases} 1 & s_k > 0 \\ \alpha * \exp(s_k) & s_k \leq 0 \\ 0 & s_k \ll 0 \end{cases} \quad (10)$$

$$g_k = \begin{cases} \frac{\partial \text{BCELoss}(\hat{y})}{\partial \hat{y}} \hat{y}(1-\hat{y}) w_k & \text{positive member} \\ \frac{\partial \text{BCELoss}(\hat{y})}{\partial \hat{y}} \hat{y}(1-\hat{y}) \alpha e^{s_k} w_k & \text{negative member} \\ 0 & \text{non-member} \end{cases} \quad (11)$$

From Equation (11), 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. 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, as shown in Equation (6). It is important to remark that the derivative of the model's confidence with respect to s_{out} is never negative and is most likely not null (given that the target model is reduced to random guessing right after both active attacks), which does not change the adversary's inference.

Appendix B.
Data and experiment details

This section presents more details about each dataset (COP-MODE and CelebA) and how they are used in these experiments, as well as a clear statement on the data and code availability.

B.1. Data and code availability

Attack and defense mechanisms performance was accessed with two real-world datasets. The CelebA dataset, often used in the literature, is fully available at https://mmlab.ie.cuhk.edu.hk/projects/CelebA.html. The COP-MODE dataset [19], collected in the NGI Trust project COP-MODE[9], is made available to researchers, particularly an anonymized version of the collected dataset. Code to reproduce experiments with the CelebA dataset is available at [8].

B.2. Experiments with COP-MODE: details

Regarding the COP-MODE dataset [19], recall that it has approximately 65K samples of manually answered permission requests from which 66% were accepted and 33% denied. This dataset is used to train models to predict user responses to permission requests. To feed this dataset into a neural network (NN), categorical features are converted to their 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 the absence of that attribute).

To train an FL model with this dataset, each client represents a user whose data is composed of the permission requests he answered. However, some users have small datasets, few of which with a single label (practically), meaning they accepted or denied almost all requests. As a result, most clients' data, as well as the centralized version of the dataset, is imbalanced with respect to the target

9. https://cop-mode.dei.uc.pt/cm-npm

variable. To overcome such limitations, one can apply local *random oversampling*[10].

Furthermore, the question of how to measure the model's ability to predict users' answers may be addressed 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. Since there is no preferred label, 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.

B.3. Experiments with CelebA: details

Regarding the CelebA dataset [15], experiments of these active AIAs, 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, and to obtain them a pre-trained Resnet-18 model from Img2Vec2[11] was used to get a vector of 512 entries. Ideally, the adversary must also have access to all binary attributes used as predictors except for the sensitive attribute, of the target sample(s). To perform AAI*, it must also have the main task label, or be able to estimate it using the target model. The FL system was composed of 10 randomly sampled clients, therefore their local datasets may be considered independent and identically distributed (IID).

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 research topic as leakage from these can raise severe privacy risks, e.g. they can be leveraged to reconstruct the original face image [10]. Additionally, one could argue that if the adversary has access to those embeddings then it could train a classification network to predict the binary sensitive attribute. However, the adversary would need a large labelled dataset 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 an ML model would only improve the adversary's knowledge about training samples, that is ideally it would increase target model leakage.

Appendix C. Scalability of attacks

Given great results of single-target attacks, in terms of theoretical performance, it is relevant to see how attack performance behaves with an increase of the number of target samples, meaning if the attack model can overfit several target samples to the point of leaking attribute information. Figures 15 and 16 show multi-target AAI* and MIA2AIA, after proper hyperparameter tuning, on CelebA and COP-MODE models, respectively. From Figure 15, it is clear that the scalability of MIA2AIA over CelebA models is limited, as TPR is not significantly better than random guessing, even though FPR significantly decreases, which means that the neuron activation rate decreased. This observation can be explained by the challenging nature of these target samples (real-valued input vectors of size 518) paired with insufficient model complexity (first two FC layers of size 1024 and 512, respectively) to overfit several target samples. Notably, AAI* TPR only slightly decreases (on average) with an increase in the number of target samples but significantly improves upon its alternative. On a less challenging type of input, Figure 16 shows that AAI* and MIA2AIA can effectively target an increasing number of samples coming from the COP-MODE dataset, although TPR can decrease on average. Neither the number of target samples nor the target attribute strongly affects FPR. However, FPR significantly increases in AAI* experiments, as it was already larger on average compared to MIA2AIA on single-target attacks.

These results are particularly relevant to show how threatening these novel active attacks could be by revealing that an *ambitious adversary, with multiple target samples, succeeds at his malicious inference*, naturally depending on the combination of the type of input and model complexity (parameter assumed to be controlled by the adversary).

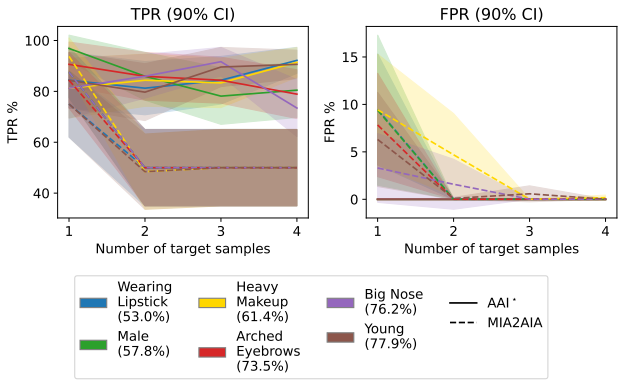

Figure 15. Multi-target AAI* and MIA2AIA theoretical TPR and FPR on CelebA models for several sensitive attributes.

Appendix D. Inference under gradient uncertainty

To show how gradients leak membership and attribute information, Figure 17 illustrates the gradient distribution of 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. Recall that MIA2AIA is based on AMI, thus this figure shows what to expect from the gradient of a single neuron used for MIA2AIA. From this

10. This 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.

11. https://github.com/christiansafka/img2vec

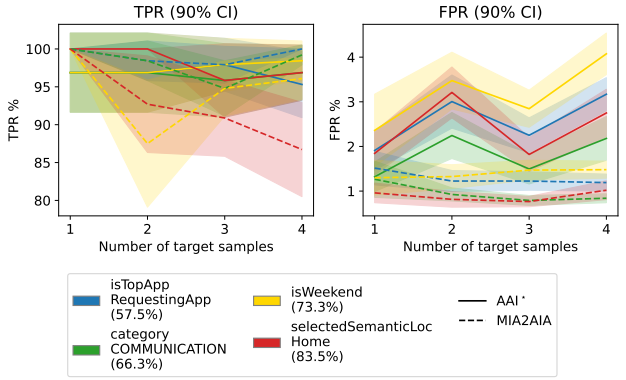

Figure 16. Multi-target AAI⋆ and MIA2AIA theoretical TPR and FPR on COP-MODE models for several sensitive attributes.

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 evident in this figure, the adversary must relax that assumption. Intuitively, an increase of FPR, in terms of neuron activation, will lead to an increase in 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) in 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.

Besides decreasing FPR, through an increase of the number of epochs to train malicious weights (as shown by Nguyen et al. [22] with AMI over CelebA models), *the degree to which an adversary can distinguish a member from a non-member also depends on the target sample's local density*. When local density is measured by Local Reachability Density[12] (LRD), results of membership inference, with AMI, greatly improve just by assuming that members have a higher local density. In fact, local density is moderately, and positively, correlated with the absolute value of gradients (34% on average). However, to improve model generalization, model trainers may increase the training sample's local density through data augmentation techniques (by oversampling it), thus increasing membership risk. Given the relation between membership and attribute inference [25], [32], also exploited in this work, one must expect that attribute inference risk also increases as a side-effect of some data augmentation techniques, which is a statement that needs further research.

12. LRD of a point is the inverse of the average reachability distance of its k nearest neighbors, where reachability distance is the maximum distance between the point and its neighbours and that point's k-distance. A greater average reachability distance (equivalent to a small LRD) means that even the nearest neighbors are far away

Appendix E.
Recovery from active attacks

One of the promises of WADM⋆ is that it can reduce model utility loss comparing to black-box strategies by enabling users to keep training while mitigating the attack. But active attacks are expected to greatly impact model performance, hence the satisfactory results of black-box strategies in terms of attack detection (Figures 8 and 9). However, if the model cannot recover from that interference, one may argue that keep training is no longer worth it. Perhaps keep using the previous state of the system (*box*) for predictions only (therefore only loosing the ability to train) is a better option if the model cannot recover its performance. This section aims to show that model performance is partially recoverable after a few training rounds, ultimately 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. This section also motivates the need for defense mechanisms that preserve model utility, as it is partially recoverable.

To evaluate the target model's ability to recover from single-target AAI⋆ and MIA2AIA after a few training rounds, the same system keeps training after the malicious attack. Note that the ability to recover is measured with ROC AUC, as the initial target model performance. Regarding CelebA models, these were able to fully recover their performance both on the train and test set, regardless of the targeted attribute, reaching close to 87% and 88% after recovering from AAI⋆ and MIA2AIA, respectively, and on average. This observation is explained by the great performance and fast convergence of these models, before any attack, as these models only trained for 2 training rounds, and the improvement from the first to the second round was negligible.

Regarding COP-MODE models, for which attacks occur after the first 5 training rounds, as their convergence required a higher number of rounds, it is also reasonable to expect that these models require more rounds to recover. Thus, Figures 18 and 19 show ROC AUC of these models in the following 10 training rounds (after AAI⋆ and MIA2AIA, respectively). The baseline (dashed black line) stands for the average ROC AUC before any attack. Results show that ROC AUC approaches the performance before any attack along with an increase in the number of training rounds that proceeded the attack. Moreover, there is a significant difference between recovery from AAI⋆ and MIA2AIA. Particularly on the training set, models can fully recover from MIA2AIA in less than 3 rounds, but recovery from AAI⋆ only happens after 10 rounds. Additionally, model recovery does not depend on the targeted attribute.

The observed difference between the recovery process from both active attacks can either be due to a distinct

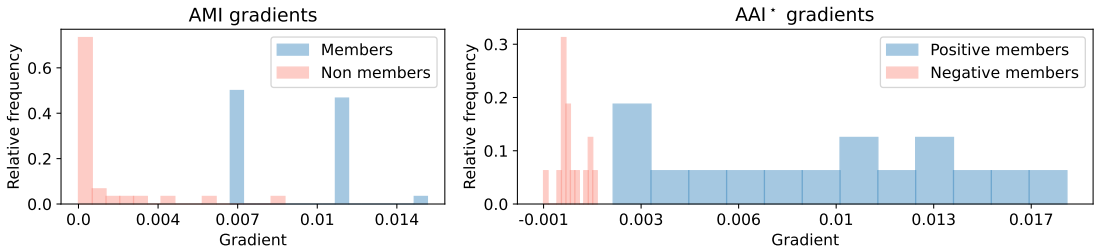

Figure 17. Gradient uncertainty illustration.

impact right after the attack or a different activation function used in each of these models. In fact, ELU activation function (used for AAI⋆) is known to prevent the vanishing gradients problem, meaning exponentially small gradients that halt the training process. By allowing small updates even when the logit is negative (cases for which RELU would return null gradients), ELU allows the training process to continue instead of stopping at an upper bound, as suggested by Figure 18.

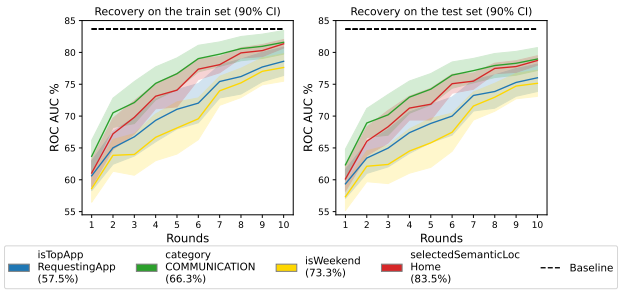

Figure 18. Recovery (ROC AUC evolution) from single-target AAI⋆ on train and test set over 10 training rounds.

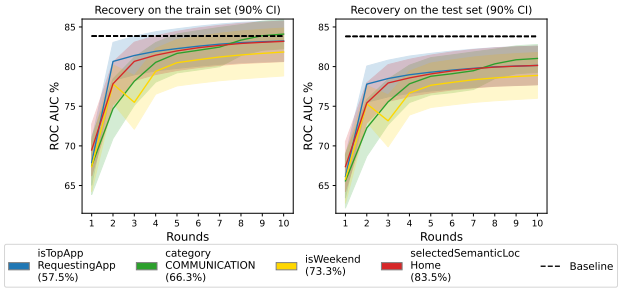

Figure 19. Recovery (ROC AUC evolution) from single-target MIA2AIA on train and test set over 10 training rounds.

However, these results reveal that after any attack, COP-MODE models present some train test gap, as convergence on the test set is significantly more difficult. As a result, these models might be more exposed to passive attacks that exploit this gap. Also note that these experiments included all clients but the capacity to recover from active attacks could change significantly if not every client is training. Particularly, if clients are free to choose a defense mechanism, and, assuming some applied black-box, they most likely left the training process in the meantime.

Appendix F. Black-box Attack Detection based on Accuracy

As explained in Section 4, BADAcc works by monitoring accuracy, or equivalently $0-1$ loss. In this section, BADAcc is presented in Algorithm 20 and explained in detail below.

Given that the $0-1$ loss of a sample follows a Bernoulli distribution, 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 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 20, where these variables are updated at the beginning of each training round (lines 10-11). At every training round, a defender receives the current error and number of samples and calculates p (expected error probability) and s (respective standard deviation) which are compared to the lower and narrower error interval (lines 12-16). This interval is updated at the end of this function (lines 17-20). Then the defender calls the function DETECT (lines 22-29) to detect whether there is an ongoing attack.

Appendix G. WADM⋆: White-box Attack Detection and Mitigation

This section presents details about the intuition behind WADM⋆ and its algorithm, as well as the WADM⋆ detection rate under FPR minimization, respectively in sections G.1, G.2, and 4.5.5.

G.1. Motivation

The difference between a benign and a malicious neuron relies on the evolution (between training rounds) of their weights distribution. Figures 21 and 22 illustrate the evolution of the weights distribution of a standard neuron and the change in weights distribution of a neuron selected to leak membership/attribute information, respectively. From Figure 21, one can see that the weights distribution of a standard neuron is expected to vary

BADAcc algorithm: Monitor accuracy over time.

1: **Parameters:**
2: n: Total number of samples observed
3: e: Number of incorrectly predicted samples
4: p_{min}: Minimum error probability observed (initialized to < 1)
5: s_{min}: Minimum standard deviation of the error probability (initialized to $< \infty$)
6: **Initial State:**
7: $p_{min} = 1$
8: $s_{min} = \infty$
9: **function** UPDATE(n, e)
10: $p \leftarrow e/n$
11: $s \leftarrow \sqrt{p(1-p)/n}$
12: **if** $p + s \geq p_{min} + 3 * s_{min}$ **then**
13: $status \leftarrow alarm$
14: **else**
15: $status \leftarrow normal$
16: **end if**
17: **if** $p + s < p_{min} + s_{min}$ **then**
18: $p_{min} \leftarrow min(p, p_{min})$
19: $s_{min} \leftarrow \sqrt{p_{min}(1-p_{min})/n}$
20: **end if**
return $status$
21: **end function**
22: **procedure** DETECT(n, e)
23: $status \leftarrow$ UPDATE(n, e)
24: **if** $status$ is $alarm$ **then**
25: Leave system
26: **else**
27: Continue
28: **end if**
29: **end procedure**

Figure 20. BADAcc algorithm

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 22. Particularly, both attacks extend the range in which weights fall in, going from $[-0.2, 0.35]$ and $[-0.25, 0.15]$ to $[-0.7, 1.3]$ and $[-7.0, 2.75]$, 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 21, to remarkably biased, in Figure 22. 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.

G.2. Algorithm

The procedure of WADM* in Algorithm 23 (lines 23-26) is divided into 2 phases: detection (lines 6-15) and mitigation (lines 17-23). In the detection phase, KL-Divergence (line 9) is used to measure the difference between the previous (line 7) and current (line 8) weights distribution of each neuron, as shown in equation (3). Note that to account for distribution shift, one must represent

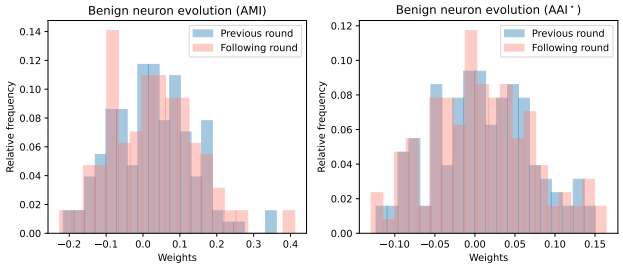

Figure 21. Evolution of the weights distribution of a standard neuron, computed from the fifth and sixth training round, once the model has converged. Each plot represents a different type of model (trained with ReLU and ELU respectively).

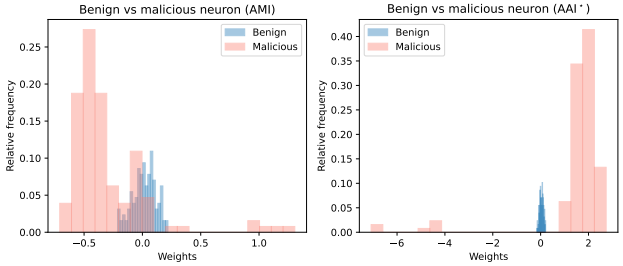

Figure 22. Change in weights distribution of a malicious neuron (after single-target attacks), computed from the fifth and sixth training round, once the model has converged. Each plot represents a different type of model (trained with ReLU and ELU respectively).

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, and, for the same reason, weights distributions are represented on a space of much smaller dimension (n_{bins}). Results were obtained by setting this parameter to 5 for robustness. A neuron is flagged as malicious if this value exceeds a preset threshold (lines 10-14), therefore being a distance-based approach. In the mitigation phase, the defender changes the current weights to the previous weights of any neuron flagged as malicious, discarding the most recent ones (line 18). A benign neuron keeps its weights and the previous weights are updated (line 20). The procedure (lines 23-26) shows how a defender combines both phases.

WADM* algorithm: Monitor neuron weights over time.

Parameters:
  W_{prev}: Previous set of neuron weights
  W_{cur}: Current set of neuron weights
  n_{bins}: Number of bins to represent weight distributions
  $threshold$: Threshold for KL-DIVERGENCE to trigger an alarm

function DETECT(W_{prev}, W_{cur}, n_{bins}, $threshold$)
  $Dist_{prev} = histogram(W_{prev}, n_{bins})$
  $Dist_{cur} = histogram(W_{cur}, n_{bins})$
  $div = \min(\ KLDiv(Dist_{prev}, Dist_{cur}),$
  $KLDiv(Dist_{cur}, Dist_{prev})\ )$
  if $div \geq threshold$ **then**
    $status \leftarrow alarm$
  else
    $status \leftarrow normal$
  end if
  return $status$
end function

function MITIGATE(W_{prev}, W_{cur}, $status$)
  if status is alarm **then**
    $W_{cur} \leftarrow W_{prev}$
  else
    $W_{prev} \leftarrow W_{cur}$
  end if
  return W_{prev}, W_{cur}
end function

procedure WADM(W_{prev}, W_{cur}, $threshold$)
  $status \leftarrow$ DETECT$(W_{prev}, W_{cur}, 5, threshold)$
  $W_{prev}, W_{cur} \leftarrow$ MITIGATE$(W_{prev}, W_{cur}, status)$
end procedure

Figure 23. WADM* algorithm