SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

March 2024, Vol. 67, Iss. 3, 132101:1–132101:13
https://doi.org/10.1007/s11432-022-3507-7

Model architecture level privacy leakage in neural networks

Yan LI[1,2], Hongyang YAN[1*], Teng HUANG[1], Zijie PAN[1*], Jiewei LAI[1], Xiaoxue ZHANG[1], Kongyang CHEN[1,2] & Jin LI[1,2]

[1]*Institute of Artificial Intelligent and Blockchain, Guangzhou University, Guangzhou* 510555*, China;*
[2]*Pazhou Lab, Guangzhou* 510330*, China*

Received 5 January 2022/Revised 6 May 2022/Accepted 7 June 2022/Published online 17 October 2023

Abstract Privacy leakage is one of the most critical issues in machine learning and has attracted growing interest for tasks such as demonstrating potential threats in model attacks and creating model defenses. In recent years, numerous studies have revealed various privacy leakage risks (e.g., data reconstruction attack, membership inference attack, backdoor attack, and adversarial attack) and several targeted defense approaches (e.g., data denoising, differential privacy, and data encryption). However, existing solutions generally focus on model parameter levels to disclose (or repair) privacy threats during the model training and/or model interference process, which are rarely applied at the model architecture level. Thus, in this paper, we aim to exploit the potential privacy leakage at the model architecture level through a pioneer study on neural architecture search (NAS) paradigms which serves as a powerful tool to automate a neural network design. By investigating the NAS procedure, we discover two attack threats in the model architecture level called the architectural dataset reconstruction attack and the architectural membership inference attack. Our theoretical analysis and experimental evaluation reveal that an attacker may leverage the output architecture of an ongoing NAS paradigm to reconstruct its original training set, or accurately infer the memberships of its training set simply from the model architecture. In this work, we also propose several defense approaches related to these model architecture attacks. We hope our work can highlight the need for greater attention to privacy protection in model architecture levels (e.g., NAS paradigms).

Keywords neural architecture search, data reconstruction attack, membership inference attack

1 Introduction

Machine learning (ML) models based on deep neural networks (DNNs) have been proven to have superior performances in various tasks, ranging from image and speech recognition to generating realistic-looking data. Experts have designed many sophisticated network architectures to promote the accuracy of DNNs [1]. However, the process of designing advanced network architectures for a given task remains a trial-and-error process that requires extensive personal experiences and human efforts. To mitigate this issue, researchers in recent years have proposed a novel ML paradigm called neural architecture search (NAS), which could automate the neural network architecture design.

The process of both NAS and network parameter training is data-dependent. Usually, the training of a well-behaved neural network is supported by a massive amount of training data harvested from a wide variety of application scenarios. These data are highly sensitive in many cases, such as user speeches, facial images, and medical or financial records. Thus, ML models generated from these data raise many privacy concerns. The existing literature has shown that an adversary may infer the properties of training data if given access to the target model or the inferring/training process.

For example, by constructing a shadow model to imitate the target model, one study [2] succeed in inferring the membership of training sets. Attacks proposed in [3] intercepted model updates to infer the presence of exact data points under the scenario of collaborative learning. Similarly, Ref. [4] leveraged generative adversarial networks (GANs) to reconstruct private datasets. Authors in [5] also presented

*Corresponding author (email: hyang.yan@foxmail.com, zijipan@protonmail.com) a comprehensive analysis of privacy challenges in deep learning networks, including threat models in different settings and defense methods.

However, to the best of our knowledge, privacy threats in NAS methods remain unexplored. NAS paradigms aim at obtaining a well-performed network architecture for a specific task, which differs from model-parameter-oriented training procedures. In particular, NAS paradigms have achieved remarkable results in promoting the automatic neural networks design and gained increasing interest because of the need to carefully reevaluate privacy threats in this area. However, existing solutions only cover privacy issues at model parameter levels and rarely at model architecture levels (e.g., NAS).

To this end, we will formulate, evaluate and disclose privacy threats in NAS paradigms in this paper. The critical question we ask is: is an adversary able to infer any property of its training sets in a NAS paradigm, if given different levels of access permissions to its training procedure? In this case, what the adversary can do (or obtain) may also be various, e.g., modifying the training sets by injecting carefully crafted adversarial samples, or only observing the final output of the training algorithm. In this study, we aim to provide a pioneering answer to this question, starting with formulating the research problem of inferring the properties of training sets in NAS paradigms. We will also construct two major attack models for different application scenarios.

In particular, our first attack model is the architectural dataset reconstruction attack (ADRA), in which the attacker only obtains the final output of the target model during the training process, and tries to reconstruct its original training sets. Notably, the output of a NAS paradigm is a specific searched neural network architecture, instead of a perdition vector in traditional classification neural networks. Our second attack model is the architectural membership inference attack (AMIA). In this attack, the attacker has black-box visibility of a pre-trained target model and tries to determine whether a known sample belongs to the training sets of this model architecture. Briefly, a black-box typically refers to the ability to query the target model, without visibility of its decision-making process. Generally, our two attack models aim to infer training set information stored only in model architectures to illustrate the privacy leakage issues in NAS paradigms.

Our model analysis and performance evaluation indicate that the model architecture in NAS paradigms would indeed bring in critical privacy leakages. For ADRA, the attacker could precisely reconstruct training samples only from the model architecture exchange procedure. In comparison, for AMIA, the attacker reaches up to 87% inference accuracy in identifying whether a sample belongs to the original training sets used in the model architecture training procedure.

To better understand and counter these two attacks, we will also discuss and evaluate several defense methods that do not rely on any cryptographic techniques [6–9]. In particular, perturbations on gradients or training samples perform best in privacy leakage protections against our NAS attacks. Differential privacy (DP) based defense schemes can reduce the attack accuracy by half, whereas maintaining high classification accuracy for the target model.

Our contributions in this paper can be summarized as follows. First, to the best of our knowledge, we are the first to identify and formulate potential privacy threats in NAS paradigms. In particular, we provide a systematic study of two privacy threats in this area, namely ADRA and AMIA. Second, the attack performances of these threat models are evaluated in different aspects, which illustrate the significant information leakages in the training sets. Finally, we investigate several non-cryptographic defense methods against these two attacks.

To the best of our knowledge, this work is the first to identify and formulate potential privacy threats in NAS paradigms. Our contributions are summarized as follows.

- We provide a systematic study of two privacy threats in NAS paradigms, termed ADRA and AMIA, which reveal potential privacy threats in NAS paradigms.

- We demonstrate and compare attack performances of attacks under these threat models in different settings. Our empirical evaluation shows that an attacker may precisely reconstruct a training image, or infer memberships in training sets.

- To further enhance the understanding of the mechanism behind our attack, we investigate several non-cryptographic defense methods.

2 Preliminaries & attack models

In this section, we introduce the preliminaries and the background of our attacks. We first present the basic idea of gradient-based architecture search paradigms at a high level. Then, we introduce our two attack models, which correspond to two existing major categories of privacy attacks, i.e., the dataset reconstruction attack and the membership inference attack.

2.1 Gradient-based NAS

NAS paradigms aim at finding an optimal network architecture for given tasks, which can be used in object classification, image generation, or an optimization target. Generally, a NAS paradigm can be formalized as

$$\boldsymbol{F}(\mathcal{D};\mathcal{S};\mathcal{T}) = F_{\mathcal{A},\theta}(\cdot). \tag{1}$$

Here, $\boldsymbol{F}(\cdot)$ denotes the NAS paradigm and $\mathcal{D}$, $\mathcal{S}$, $\mathcal{T}$ denote the training set, search space, and optimization target (loss function), respectively. $F_{\mathcal{A},\theta}(\cdot)$ is the output of a NAS paradigm, i.e., a specific network with architecture $\mathcal{A}$ and weights θ. In this paper, the training set $\mathcal{D}$ is the inference target of an adversary attacker.

A neural network $F_{\mathcal{A},\theta}(\cdot)$ can typically be viewed as a direct acyclic graph (DAG) with computational cells as nodes and their connection patterns as edges. Each computational cell consists of several operations to its input, such as convolution, and pooling. In a neural network with P computational cells, the output of p-th computational cell can be represented as

$$\boldsymbol{f}_p = \theta_p \odot_p \bigvee_{j\in J_p} \boldsymbol{f}_j, \tag{2}$$

where θ_p are the parameters of p-th cell, $\odot_p$ is the operation in this cell, and $\bigvee_{j\in J_p} \boldsymbol{f}_j$ is the collection of previous outputs connected to p-th cell. The process of a NAS paradigm is to find an optimal operation $\odot$ for each cell and optimal connection patterns from the given search space $\mathcal{S}$.

Gradient-based NAS paradigms [10, 11] provide an effective solution to this optimization problem via relaxing the categorical candidate operations between these nodes to a continuous space. Specifically, in each cell a weight $\alpha_{\odot i}$ is assigned to i-th candidate operation in $\mathcal{S}$. In this way, a discrete architecture can be obtained by picking out the most likely operation: $\odot = \arg\max_{\alpha_\odot\in\mathcal{S}} \alpha_{\odot i} \odot_i$. Similarly, a weight α_{ij} is assigned to the connection between i-th and j-th cell, which defines whether these two cells should be connected or not. If α_{ij} reaches a certain threshold, a connection should be invoked and vice versa.

In this way, the architecture inside a neural network can be parameterized as a continuous and differentiable parameter: $\alpha = \{\alpha_\odot, \alpha_{IJ}\}$. Thus, gradient-based optimization methods such as stochastic gradient descent (SGD) or adaptive moment estimation (ADAM) can be used to solve the loss function $\mathcal{L}_\alpha(\cdot)$. In practice, some approximation schemes can be adopted to reduce the computation cost. The optimization of NAS can be formulated as a bilevel optimization problem, with α as the upper-level variable and θ as the lower-level variable:

$$\min_\alpha \mathcal{L}_\alpha(\theta^*(\alpha),\alpha) \quad \text{s.t.} \quad \theta^*(\alpha) \; = \; \arg\min_w \mathcal{L}_\theta(\theta,\alpha). \tag{3}$$

Notably, the loss function for model architecture α may be varied along with the model parameter θ. The training set for α and θ can also be independent.

2.2 Attack models

An adversary attempts to obtain the original information of $\mathcal{D}$ as much as possible, while defenders will endeavor to maintain its confidentiality. In this paper, we formulate and investigate privacy leakages in two typical attack models.

Attack model I: ADRA. In this attack, we assume that the attacker can obtain the final output $\mathcal{A}$ of an ongoing NAS paradigm (e.g., a specific model architecture). Note that this output does not include any internal weights or hyper-parameters. The attacker aims to reconstruct the training samples from this model architecture only. Figure 1 (top, right) shows the pipeline of this attack.

Attack model II: AMIA. In this attack, we assume that the attacker has black-box visibility to the pre-trained target model. The attacker may query the target model with its local samples $\mathcal{D}'$ multiple

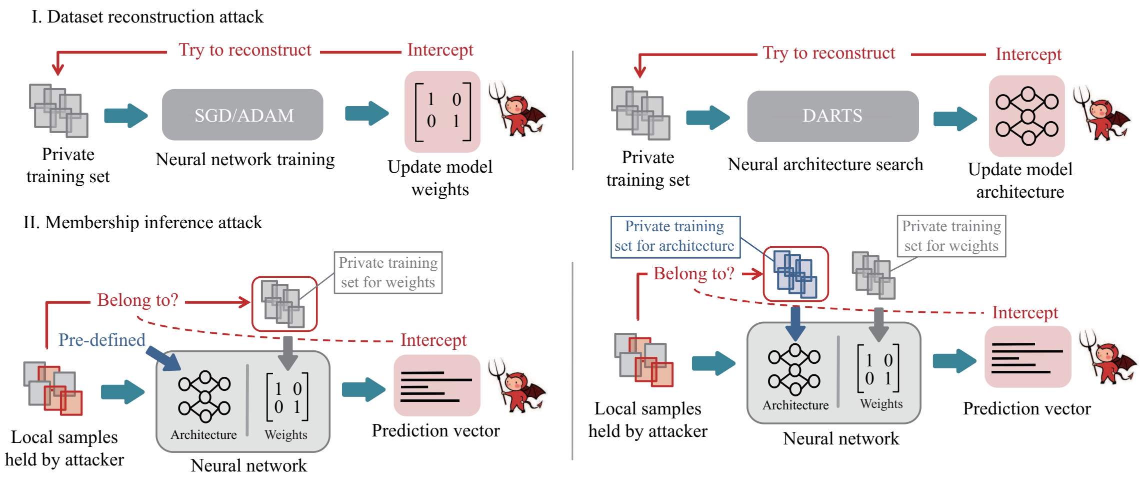

I. Dataset reconstruction attack
Try to reconstruct
Intercept
Private training set
SGD/ADAM
Neural network training
Update model weights
DARTS
Neural architecture search
Update model architecture
II. Membership inference attack
Belong to?
Private training set for weights
Pre-defined
Intercept
Local samples held by attacker
Architecture
Weights
Neural network
Prediction vector
Private training set for architecture

Figure 1 (Color online) Pipeline and comparison between the model-parameter-oriented attack (left) and the NAS attack (right).

times and tries to tell which part of $\mathcal{D}'$ belongs to the training set $\mathcal{D}$ used in the target model. Notably, in NAS paradigms, as mentioned in Subsection 2.1, the training set for model architecture is different from that for model parameters. Therefore, intuitively, sensitive information in the training sets for architecture will be only kept in the output model architecture. We aim to infer membership information of this part of the training sets, which implies that the model architecture will remember information from the training sets, as well as model weights. Figure 1 (bottom, right) shows the pipeline of this attack.

In this paper, we consider two typical application scenarios for our attack models. The first one is a NAS in a federated manner, wherein multiple data holders jointly train a shared model without reviewing their local private datasets. If an attacker manages to intercept the communication between clients and servers, it may leverage this information to reconstruct a certain client's dataset. The second one is ML as a service (MLaaS), where many large companies offer their computing resources as services to data holders (e.g., hospitals or banks), allowing data holders to perform predictions over their new data. Data holders only, they only upload data and receive prediction results from the server. In this scenario, if an attacker invades the access permission to query the offered prediction application programming interface (API), he may leverage this information to infer properties of the original dataset used to train this online prediction model.

3 Attack model I: ADRA

In this section, we present the attack model of AMIA and empirically show that an adversary is able to reconstruct the training sample if given model architecture gradient α.

3.1 Information leakage from gradients

Prior studies [3] have demonstrated that an adversary is able to reconstruct training samples if given update gradients during training.

The attacker first randomly creates a "dummy" sample $\mathcal{D}'$ with random noises, and then tries to optimize this dummy sample to make it close to the real training data as much as possible. This dummy sample however becomes the optimization target, with the optimization goal to minimize the difference (e.g., Euclidean distance) with the real sample. When the optimization process finishes, the private real training data will be disclosed. In each epoch, the attacker reconstructs training samples from the update architecture gradient α only.

Definition 1 (ADRA)**.** For a given architecture gradient $\Delta\alpha$, batch size $\mathcal{B}$, and randomly initialized dummy sample $\mathcal{D}'$, ADRA computes

$$\arg\min_{\mathcal{D}'} D\left(\Delta\alpha, \frac{1}{\mathcal{B}}\sum_{b\in\mathcal{B}}\alpha'_b\right), \tag{4}$$

where $D(\cdot)$ is a distance measurement function and α' is the dummy gradient produced by $\mathcal{D}'$.

Denote the output architecture of the target model by α_{t1} and α_{t2}. Thus, the updated architecture gradient $\Delta\alpha$ can simply be obtained by $\Delta\alpha = \alpha_{t2} - \alpha_{t1}$. If $D(\cdot)$ is implemented as a L_2 distance, the dummy sample $\mathcal{D}'$ can be literally optimized by

$$\begin{aligned}\mathcal{D}' &= \arg\min_{\mathcal{D}'} ||\Delta\alpha - \alpha'||_2 \\ &= \arg\min_{\mathcal{D}'} \left\| \Delta\alpha - \frac{\partial \mathcal{L}_\alpha(F(\boldsymbol{X}'|\alpha', \theta_r), \mathcal{Y}')}{\partial \alpha'} \right\|_2. \end{aligned} \tag{5}$$

where θ_r denotes model weights in the shadow model, and $\mathcal{Y}'$ is the dummy label that corresponds to the dummy sample, which is also an optimization target. Intuitively, the success of the dataset reconstruction depends on the solvability of (5), i.e., whether the second derivative $\frac{\partial^2 \mathcal{L}_\alpha(\cdot)}{\partial \boldsymbol{X} \partial \alpha}$ can be determined. In gradient-based NAS paradigms, architecture gradient α' can be rewritten as

$$\alpha' = \frac{\partial \mathcal{L}_\alpha(\boldsymbol{f}_P, y)}{\partial \boldsymbol{f}_P} \cdot \frac{\partial \boldsymbol{f}_P}{\partial \alpha}. \tag{6}$$

In (6), $\partial \mathcal{L}_\alpha(\cdot)/\partial \boldsymbol{f}_P$ can be easily determined by $(\partial \mathcal{L}_\alpha(\cdot)/\partial \boldsymbol{f}_0) \prod_{p=0}^{P-1} (\partial \boldsymbol{f}_p / \partial \boldsymbol{f}_{p+1})$ according to a chain rule, where $\boldsymbol{f}_0$ is the input $\boldsymbol{X}$ itself. The second term, $\partial \boldsymbol{f}_P / \partial \alpha$ can be simplified by expanding $\boldsymbol{f}_P$ as

$$\boldsymbol{f}_P = \sum_{m \in M} \cdots \sum_{n \in N} \alpha^m f_\theta^m(\cdots \alpha^n f_\theta^n(\boldsymbol{X})), \tag{7}$$

where $f_\theta^n(\cdot)$ denotes the operation to its input with parameter θ, m and n are the index of certain types of operation in the M-th and the N-th cell, respectively. In convolution cells, $f_\theta^n(\cdot) = \sigma(\theta x + b)$, where $\sigma(\cdot)$ is an activation function, and b is the bias. The decision process of a neural network can then be viewed as a weighted summation characterized by α. For a specific architecture parameter α^M, we have

$$\frac{\partial \boldsymbol{f}_P}{\partial \alpha^M} = f_\theta^M \left(\sum_{m \in M} \alpha^m f_\theta^m(\boldsymbol{X}^m) \right), \tag{8}$$

where m is the index of prior cells connected to the M-th cell.

Consider a simple case in most neural networks with an identity activation function, e.g., $f(x) = \theta x + b$. The derivative of (8) of the input $\boldsymbol{X}$ can be obtained by

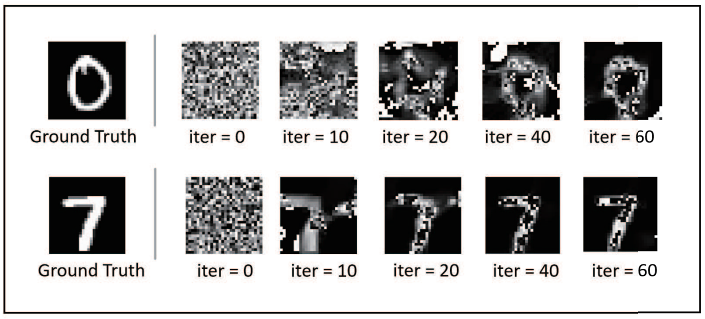
$$\frac{\partial^2 \boldsymbol{f}_P}{\partial \alpha^M \partial \boldsymbol{X}} = \sum_{m \in M} \left(\prod_{i \in I} \alpha_i \theta_i \right), \tag{9}$$

where α_i and θ_i denote the prior architecture parameter and weights that have a connection to the M-th cell, respectively.

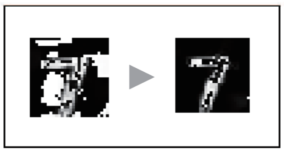
The above analysis implies that this reconstruction attack, or in other words the solvability of (5), mainly depends on the following aspects.

• **Sufficient architecture parameters.** Generally, in a neural network with P layers, we have $\alpha \in \mathbb{R}^{\frac{1}{2} \times P \times P}$. In practice, P is often chosen between 32 and 256, as the time complexity for optimizing α reaches $\mathcal{O}(n^2)$. In CNNs, weights $\theta \in \mathbb{R}^{q \times q \times c \times n_q}$, where q, c, and n_q denote the spatial extent of filters, input channels, and the number of filters, respectively. However, the input data has a significantly large size than α, i.e., $\boldsymbol{X} \in \mathbb{R}^{h \times w \times c}$, where h, w, and c denote the height, weight, and channel of an image, respectively. During the attacks, it may potentially result in multiple local minimums in solving equation (5), which associates the success of the dataset reconstruction with the initialization noises of the dummy samples. Such dependence can be relaxed by increasing architecture parameters α.

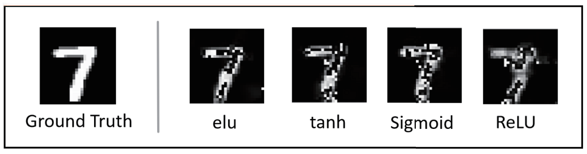
• **Continuous activation functions.** Optimizing $\boldsymbol{X}$ to guide the architecture search paradigm produces certain α^M requires a continuous function $f(\cdot)$ over $\boldsymbol{X}$ to make the optimization objective differentiable. Thus, non-continuous activation functions such as ReLU will decrease the reconstruction accuracy to a certain degree. Besides, to ensure the existence of each slight change of $\boldsymbol{X}$ causes non-negligible impact to α, an increasing activation function $\sigma(\cdot)$ is preferred, i.e., $\frac{\partial \sigma}{\partial x} > 0$ for any x, to maximize the attack efficiency.



(a)



(b)



(c)

Figure 2 Illustration of (a) the reconstruction process, (b) denoising function, and (c) reconstruction of different used activation functions.

3.2 ADRA

We show an adversary may reconstruct original training samples from architecture gradients, starting from a simple case: there is only one sample in a training batch $\mathcal{B}$. Batch size is also a key factor in this reconstruction attack. If $\mathcal{B} > 1$, the acquired architecture gradient is a mixture of several data points, thus identifying a component of the mixture requires extra effort. In particular, there exists $\mathcal{O}(\mathcal{B}!)$ permutations, which greatly obstruct gradient selection and slow the optimization process.

We use MNIST hand written-digits, a collection of 70000 grayscale images with the size of 28×28, to implement the architectural reconstruction attack. Our implementation is based on the open-sourced code from DARTS[1)], and their settings, with Python 3.7 and 4 NVIDIA Tesla V100 GPUs. Detailed settings of this attack are given in Appendix A.

In addition, a denoising function is adopted to make the reconstruction results more understandable in each iteration, as the size mismatch between $\boldsymbol{X}$ and α brings in some "uncontrollable pixels" that stay firmly around the maximum value. Specifically, we turn all pixels exceeding a certain threshold κ to 0. In our experiment, we set $\kappa = 245$. Figure 2(b) shows this denoising process.

Results. Figure 2(a) reports the reconstruction results. Although the reconstructed images are not exactly identical to the real training samples, the digits in the reconstructed images are still recognizable and the "outline" is consistent with the ground truth in general. It means that model architectures contain significant private information.

We also investigate the reconstruction attacks under different possible activation functions, such as elu, tanh, sigmoid, and ReLU. The reconstruction results are shown in Figure 2(c). It can be seen that the reconstructed quality significantly depends on the activation function, where elu and tanh could produce preferable reconstruction results.

After several iterations, the reconstructed image may also have background noises without clear noise patterns. It means that the optimization in equation (5) cannot reach a local optimum. In fact, the success rate (SR) (i.e., the reconstructed image converges to a recognizable one) is around 30% in our attack. Since our analysis implies that the reconstruction SR is associated with network sizes and activation functions, we report the SR and the mean square error (MSE) under different network sizes (P) and different activation functions, which are listed in Table 1. From Table 1 we can infer that, in most cases, a larger P (except ReLU) improves the reconstruction accuracy. Also, an activation function with a simple derivation form is beneficial to the reconstruction. Notably, in our results for ReLU, the attack effectiveness decreases with the increase of P. We speculate that the reason behind this is that, overlaying noncontinuous activation functions (in this case ReLU) would significantly make the perdition

1) https://github.com/quark0/darts.

Table 1 Attack SR and MSE under different network sizes (P) and different activation functions

		$P = 32$	$P = 64$	$P = 96$	$P = 128$	$P = 256$
elu	SR (%)	31.4	36.0	39.8	42.1	42.8
	MSE	2.98	2.87	2.44	2.07	1.87
tanh	SR (%)	27.6	32.9	36.4	38.0	38.5
	MSE	2.84	2.79	2.71	2.65	2.49
Sigmoid	SR (%)	26.1	32.1	37.6	35.3	36.2
	MSE	3.21	3.08	2.90	2.91	2.88
ReLU	SR (%)	23.4	22.5	22.9	22.2	21.7
	MSE	3.18	3.32	3.24	3.17	3.20

Table 2 Attack SR and MSE under different network sizes (P) and different batch sizes

		$P = 32$	$P = 64$	$P = 96$	$P = 128$	$P = 256$
$\mathcal{B} = 1$	SR (%)	31.4	36.0	39.8	42.1	42.8
	MSE	2.98	2.87	2.44	2.07	1.87
$\mathcal{B} = 2$	SR (%)	21.2	23.9	23.8	24.0	24.5
	MSE	3.20	3.24	3.12	3.05	2.92
$\mathcal{B} = 4$	SR (%)	14.4	14.0	14.1	14.2	14.5
	MSE	5.03	4.91	4.83	4.88	4.82
$\mathcal{B} = 8$	SR (%)	0	0	0	0	0
	MSE	–	–	–	–	–

into a nonlinear process. This confirms our analysis in Subsection 3.1: a continuous activation function would be helpful for the reconstruction attack.

Finally, we continue our attack on a large-batch training. We investigate the attack performance when $\mathcal{B} = 1, 2, 4$, and 8 with elu, which are listed in Table 2. From Table 2 we can find that, in multi-samples reconstruction, architecture gradient fails to help the reconstruction. For example, when $\mathcal{B} = 8$, the reconstruction does not converge.

4 Attack model II: AMIA

In this section, we present the attack model of AMIA and empirically show that an adversary is able to determine whether a data point belongs a particular training set used in the model architecture training.

4.1 Information leakage from architectures

Membership privacy was firstly formalized in [12], where adversaries may determine if a data sample $\mathcal{D}'$ belongs to the training set $\mathcal{D}$ of a black box target model $F_{\mathcal{A},\theta}(\cdot)$. This is achieved by training multiple shadow models $F^s_{\mathcal{A},\theta}(\cdot)$ to mimic the behavior of a target model $F_{\mathcal{A},\theta}(\cdot)$, and leveraging these shadow models' output pattern $\boldsymbol{V}$ to train an attack model $V(\cdot)$. These shadow models are initialized with a fixed architecture [13]. The attack model $V(\cdot)$ then identifies whether a data sample belongs to a model's training set by the model output prediction vector (i.e., posterior probability). MIA has proved that model weights will remember the information of the training set, which brings in information leakage. We argue that, however, information on the training set will be not only kept in model weights but also in model architectures.

The "inference ability" of an adversary is described by a parameter ϵ. Denote a sample and its label by x and y, if $\epsilon = 0$, the adversary has (x, y) sampled uniformly from the training set $\mathcal{D}$ to generate a neural network $F_{\mathcal{A},\theta}(\cdot)$. Otherwise, if $\epsilon = 1$, the adversary has (x, y) from the general population. During AMIA, the adversary exploits his additional knowledge (e.g., prediction vector in our attack model) to predict the correct value of ϵ. Notably, ϵ can also be considered as an indicator of the distance between (x, y) and the real training set $\mathcal{D}$. The Bayesian optimal membership inference rate in a model architecture is given in Theorem 1.

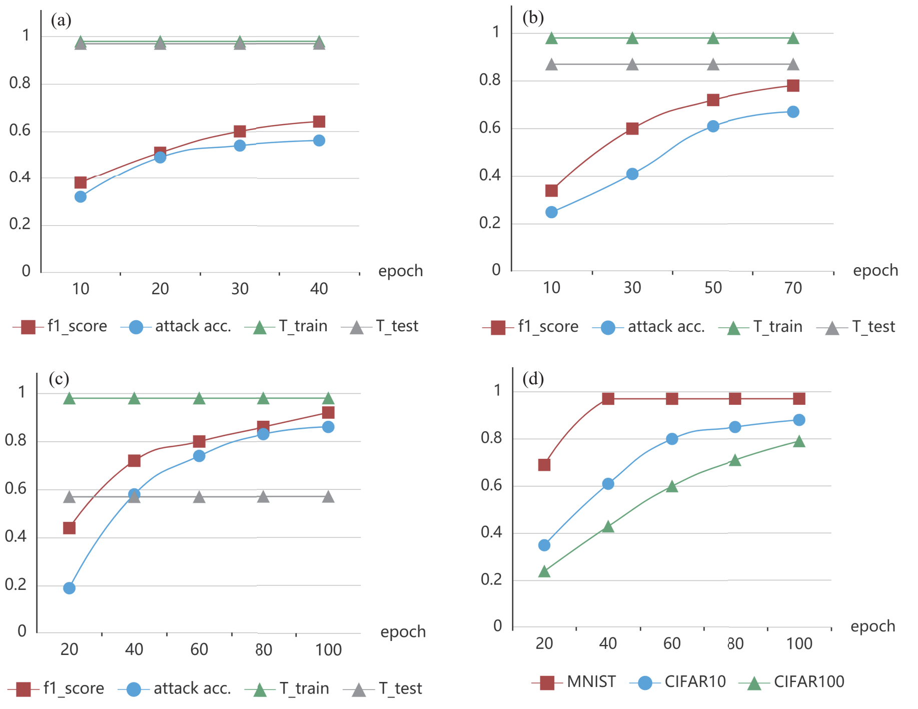


Figure 3 (Color online) Performances of the attack model and the target model for (a) MNIST, (b) CIFAR10, (c) CIFAR100, and (d) the f_1-score.

Theorem 1. Given θ and α, the optimal membership inference for a sample $\mathcal{D}_i$ is

$$\mathbb{P}(\epsilon_{\mathcal{D}_i}=1|\alpha,\theta)=\mathbb{E}_{\mathcal{D}_r}\left[\left(1+\left(\frac{1-\lambda}{\lambda}\right)\frac{\mathbb{P}(\alpha|\epsilon_{\mathcal{D}_i}=1,\theta,\mathcal{D}_r)}{\mathbb{P}(\alpha|\epsilon_{\mathcal{D}_i}=0,\theta,\mathcal{D}_r)}\right)^{-1}\right], \tag{10}$$

where $\lambda=\mathbb{P}(\epsilon_{\mathcal{D}_i}=1)$ and $\mathcal{D}_r=\mathcal{D}-\mathcal{D}_i$. The proof is given in Appendix B.

We consider the attack accuracy as the probability that his prediction equals ϵ. If the attack model $V(\cdot)$ is randomly initialized without proper training, it should maintain 50% accuracy. Thus, we formalize the attack accuracy of $V(\cdot)$ as

$$\text{attack\_acc.}(V)=2\mathbb{P}[V(\cdot)=\epsilon]-1. \tag{11}$$

We also consider f_1-score as our major evaluation matrix. It describes the general performance of the attack model:

$$f_1\text{-score}(V)=\frac{2\mathbb{P}[\epsilon=1|V(\cdot)]\cdot\mathbb{P}[|V(\cdot)|\epsilon=1]}{\mathbb{P}[\epsilon=1|V(\cdot)]+\mathbb{P}[|V(\cdot)|\epsilon=1]}, \tag{12}$$

where $\mathbb{P}[\epsilon=1|V(\cdot)]$ and $\mathbb{P}[V(\cdot)|\epsilon=1]$ are precision and recall, respectively.

4.2 AMIA

In this subsection, We evaluate the effectiveness of AMIA with attack accuracy and f_1-score in several datasets such as MNIST, CAFAR-10, and CAFAR-100. Each dataset is independently identically distributed (IID) and divided into four sub-datasets: $\mathcal{D}^\theta$, $\mathcal{D}^\alpha$, $\mathcal{D}^s_{\text{Train}}$, and $\mathcal{D}^s_{\text{Test}}$. In particular, $\mathcal{D}^\theta$ and $\mathcal{D}^\alpha$ are used to optimize the model parameter and architecture, respectively. The network consists $P=64$ computation cells. Detailed settings of this attack are presented in Appendix C.

Results. Figures 3(a)–(c) report the training performance of the attack model and the target model for MNIST, CIFAR10, and CIFAR100, respectively. T_train and T_test denote the training accuracy and test accuracy of the target model. Figure 3(d) reports the f_1-score of the shadow model. From Figure 3, we can infer the following solutions.

(1) The attack accuracy increases in each training epoch, and reaches 0.56, 0.67, and 0.86 for MNIST, CIFAR10, and CIFAR100, respectively. It proves that private information in the training set for model architecture is not safe, and can be directly leaked through AMIA.

Table 3 Attack results on different network scales

		$P = 32$	$P = 64$	$P = 128$	$P = 256$
MNIST	Attack acc.	0.56	0.56	0.57	0.57
	f_1-score	0.62	0.64	0.65	0.65
CIFAR10	Attack acc.	0.63	0.67	0.69	0.70
	f_1-score	0.74	0.78	0.79	0.79
CIFAR100	Attack acc.	0.83	0.86	0.89	0.91
	f_1-score	0.87	0.92	0.94	0.96

Table 4 Attack results on different NAS paradigms

		DARTS	ENAS	PDARTS	SMASH
MNIST	Attack acc.	0.56	0.51	0.53	0.53
	f_1-score	0.64	0.64	0.65	0.61
CIFAR10	Attack acc.	0.62	0.60	0.64	0.60
	f_1-score	0.75	0.72	0.75	0.74
CIFAR100	Attack acc.	0.86	0.79	0.87	0.81
	f_1-score	0.92	0.82	0.90	0.89

Table 5 Attack results of different distribution of attack dataset ($P = 64$)

		$\eta = 1.00$	$\eta = 0.75$	$\eta = 0.50$	$\eta = 0.25$
MNIST	Attack acc.	0.56	0.55	0.55	0.54
	f_1-score	0.64	0.62	0.61	0.61
CIFAR10	Attack acc.	0.62	0.61	0.58	0.56
	f_1-score	0.75	0.72	0.70	0.69
CIFAR100	Attack acc.	0.86	0.83	0.81	0.75
	f_1-score	0.92	0.84	0.82	0.76

(2) Dataset with complex patterns (e.g., CIFAR100) is much more vulnerable than the one with a relatively simple pattern (e.g., CIFAR10) when facing AMIA. This is consistent with prior experience [13]. In addition, different levels of overfitting can be observed in the target model. This implies that model generalization prevents privacy leakages to a certain degree, in both MIA and AMIA.

The AMIA is also evaluated in more practical scenarios. Table 3 further reports the attack performances with different network scales of the target model. We can find that the architecture of a deeper network tends to absorb information in training sets with complex patterns. In addition, we also implement AMIA on different NAS paradigms, as shown in Table 4. It can be seen that reinforcement-learning based (ENAS [14]) and one-shot (SMASH [15]) NAS paradigms have a better generalization ability against AMIA.

We also report the attack results ($P = 64$) in Table 5 when the attacker does not process an attack dataset that is completely independent and identically distributed to the target training set. Specifically, we use a parameter η to illustrate how much proportion of the attack dataset is overlaps the distribution of the target training set. If $\eta < 1$, $(1-\eta)\%$ of the attack dataset is chosen from another dataset irrelevant to the target training set. From Table 5 we can observe that, a dataset with a simple pattern would be more robust against AMIA. Besides, the distribution attack dataset also contributes to the attack result.

4.3 Improved AMIA

The above attack results highlighted privacy leakage in the model architecture of pretrained neural networks. Specifically, one key factor to deploy the MIA is the linearly separable degree between the member samples and the non-member samples, which requires the shadow model $F^s_{\mathcal{A},\theta}(\cdot)$ to overfit in its training sets. However, the shadow model in AMIA is initialized with a fixed architecture, which might limit its ability in imitating the target model.

We thus improve AMIA by making the architecture of shadow models searchable. As the attacker does not know the NAS paradigm of the target model, it will adopt a random one (e.g., DARTS). Figure 4 illustrates the decision boundaries and distributions of member/non-member samples for different shadow models with fixed/unfixed architecture. Compared with the fixed architecture in Figure 4(a), it can be seen that member/non-member samples are more separable in architecture-searchable shadow models as

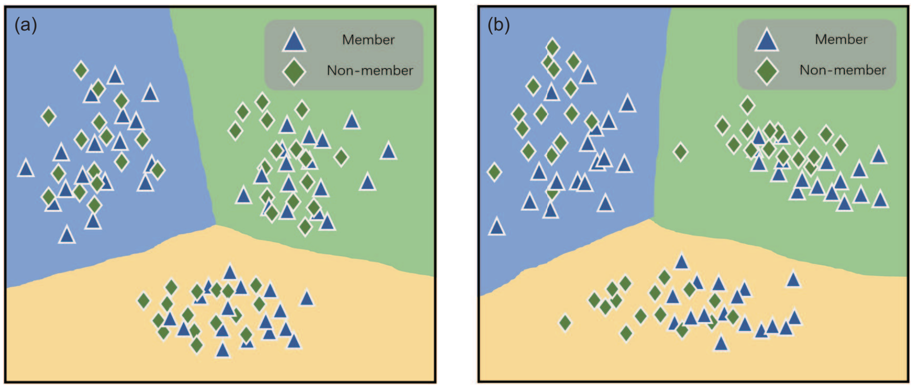


Figure 4 (Color online) Decision boundaries and distributions of member/non-member samples for different shadow models. (a) Fixed architecture; (b) unfixed architecture.

Table 6 Defense results in various scenarios

		DP_SGD					LR		DP		MD	
		$B=1,\ P=32$	$B=1,\ P=256$	$B=4,\ P=256$			$P=64$	$P=256$	$P=64$	$P=256$	$P=64$	$P=256$
elu	SR (%)	−18.2	−13.8	−11.2	MNIST	Attack acc.	−0.10	−0.11	−0.31	−0.35	−0.14	−0.16
	MSE	+1.44	+1.45	+6.89		f_1-score	−0.11	−0.13	−0.34	−0.36	−0.18	−0.20
tanh	SR (%)	−17.2	−13.1	−12.0	CIFAR10	Attack acc.	−0.14	−0.17	−0.39	−0.40	−0.21	−0.22
	MSE	+1.64	+1.81	+10.29		f_1-score	−0.15	−0.17	−0.43	−0.44	−0.25	−0.25
Sigmoid	SR (%)	−20.2	−18.2	−18.0	CIFAR100	Attack acc.	−0.23	−0.26	−0.55	−0.54	−0.30	−0.31
	MSE	+2.04	+3.25	+12.50		f_1-score	−0.27	−0.31	−0.52	−0.53	−0.35	−0.34

shown in Figure 4(b). Our evaluation shows that, under the setting of $P = 32$ and DARTS as the target NAS paradigm, the attack accuracy and f_1-score for three datasets increase by 0.13 and 0.17 on average, respectively.

5 Defense schemes

In this section, we evaluate several potential defense schemes against our two attacks. For ADRA, we adopt DP_SGD [16], where the updated architecture gradients are added with a random noise under Gaussian Distribution $\mathcal{N}(0, 0.01)$. For AMIA, L_2-norm regularization (LR), DP, and model distillation (MD) are evaluated. Particularly, in LR, the sum of the squared model architecture $\Sigma\alpha^2$ is added into the loss function as a penalty term to be minimized during its training. In DP, a random noise under $\mathcal{N}(0, 0.01)$ is added to each sample in the training set. In MD, it transfers the knowledge in the searched model to a concise student model. Detailed defense settings are given in Appendix D.

These defense results are reported in Table 6, where each result indicates the increment (or decrement) value caused by current defense settings, in comparison with a standard attack without any defense. Please note that the comparison between different defense schemes should consider multiple factors, especially the trade-off between its effectiveness and computing cost. In this paper, we mainly consider the upper limit of the defense performance, regardless of other evaluation metrics.

For ADRA, we can see that added noises in the architecture gradient will reduce the SR and obstruct the data reconstruction. With the increment of model complexities and training batches, meaningful features in training samples can be recovered easily. Generally, DP_SGD generates an acceptable accuracy loss (around 7%), which can be seen as a preliminary defense method against ADRA.

For AMIA, DP-based dense schemes also outperform LR and MD in general. Averaged accuracy losses for LR, DP, and MD are 3%, 7%, and 4%, respectively, which can also be considered acceptable. In addition, DP reduces the inference accuracy by half, while LR and MD reduce by 20% and 30%, respectively. Therefore, perturbations on gradients or training samples achieve the best defensive performances.

6 Related work

Privacy attacks on ML models have attracted growing interest [17–19] in recent years. Various attack methods have also been proposed to determine sensitive information within their training sets. However, gradient sharing in collaborative learning has been proven unsafe against the reconstruction attacks [3, 20]. While the above solutions formalize the reconstruction attack as an optimization problem, Ref. [21] leverages a multi-task GAN to generate samples by refining the recovered class representatives. More related solutions are also illustrated in [22], which explores different distance measurements in the reconstruction, and in [23] which improves the reconstruction accuracy with large training batches. Various gradient-based information leakage methods have also been studied in [5, 24].

DNNs have also been proven to be vulnerable against MIA [12]. A number of studies [5, 14] have studied MIA in different application scenarios with significant inference accuracies. One paper [25] studied membership inference against GANs, while MIA in a white-box setting has been studied in [26]. After that, recent attempts to reveal the underlying mechanisms [27, 28] have also promoted defenses against MIA [29]. Many experiences of MIA are summarized in [30].

Apart from privacy attacks, DNNs also suffer from adversarial attacks, where an imperceptible perturbation is added to mislead an ML-based classifier [31, 32]. Effective attacks include fast gradient sign method (FGSM) [33], momentum iterative FGSM (MI-FSGM) [34], and project gradient descent (PGD) [35]. A number of recent studies have focused on broadening the application of attacks in various scenarios [36–38] and in the physical world [39, 40]. Subsequently, some defense schemes have been proposed, including preprocessing techniques [41], feature denoising [42], regularization [32], and model ensambles [43]. Given that privacy attacks and adversarial attacks are built upon similar underlying mechanisms to a certain degree, studies on adversarial attacks may contribute to understanding privacy issues in neural networks. Particularly, both privacy and adversarial attacks depend on the complexity of the model decision boundary. Well-generalized neural networks often employ a smooth decision boundary, which is helpful in resisting various kinds of attacks.

NAS paradigms [44]. The current work is thus motivated to investigate existing privacy attacks [45–50] solutions that can be extended to NAS paradigms, which motivates this work.

7 Conclusion and future work

In this work, we studied privacy threats in NAS paradigms. ADRA and AMIA were also proposed in NAS paradigms. The results prove that model architecture might also cause potential privacy leakage in neural networks. Our empirical study shows that DP-based defense schemes might prevent such privacy leakage to some extent. This work can be considered a pioneering study on privacy leakages in terms of NAS. Future works may consider finding ways to exploit more potential threats in NAS with better attack accuracy and more practical adversarial defense schemes under more practical scenarios.

Acknowledgements This work was supported by National Natural Science Foundation of China (Grant Nos. U1936218, 62102107, 62072132, 62002074, 62072127, 62002076, 61802383, U20A20176), Science and Technology Program of Guangzhou of China (Grant No. 202002030131), Guangdong Basic and Applied Basic Research Fund Joint Fund Youth Fund (Grant No. 2019A1515110213), and Research Project of Pazhou Lab for Excellent Young Scholars (Grant No. PZL2021KF0024).

References

1 Huang G, Liu Z, van der Maaten L, et al. Densely connected convolutional networks. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2017. 4700–4708

2 Shokri R, Stronati M, Song C, et al. Membership inference attacks against machine learning models. In: Proceedings of IEEE Symposium on Security and Privacy (SP), 2017. 3–18

3 Zhu L, Liu Z, Han S. Deep leakage from gradients. In: Proceedings of Advances in Neural Information Processing Systems, 2019. 14774–14784

4 Salem A, Bhattacharya A, Backes M, et al. Updates-leak: data set inference and reconstruction attacks in online learning. In: Proceedings of the 29th USENIX Security Symposium (USENIX Security 20), 2020. 1291–1308

5 Nasr M, Shokri R, Houmansadr A. Comprehensive privacy analysis of deep learning: passive and active white-box inference attacks against centralized and federated learning. In: Proceedings of IEEE Symposium on Security and Privacy (SP), 2019. 739–753

6 Song Y, Li Z, Li Y, et al. Attribute-based signcryption scheme based on linear codes. Inf Sci, 2017, 417: 301–309

7 Zhang X, Zhu L, Wang X, et al. A packet-reordering covert channel over VoLTE voice and video traffics. J Netw Comput Appl, 2019, 126: 29–38

8 Zhang Q, Wang X, Yuan J, et al. A hierarchical group key agreement protocol using orientable attributes for cloud computing. Inf Sci, 2019, 480: 55–69

9 Li J, Wang X, Huang Z, et al. Multi-level multi-secret sharing scheme for decentralized e-voting in cloud computing. J Parallel Distr Comput, 2019, 130: 91–97

10 Chen X, Xie L, Wu J, et al. Progressive differentiable architecture search: bridging the depth gap between search and evaluation. In: Proceedings of IEEE/CVF International Conference on Computer Vision, 2019. 1294–1303

11 Liu H, Simonyan K, Yang Y. DARTS: differentiable architecture search. 2018. ArXiv:1806.09055

12 Long Y, Bindschaedler V, Gunter C A. Towards measuring membership privacy. 2017. ArXiv:1712.09136

13 Salem A, Zhang Y, Humbert M, et al. ML-leaks: model and data independent membership inference attacks and defenses on machine learning models. 2018. ArXiv:1806.01246

14 Pham H, Guan M, Zoph B, et al. Efficient neural architecture search via parameters sharing. In: Proceedings of International Conference on Machine Learning (PMLR), 2018. 4095–4104

15 Brock A, Lim T, Ritchie J M, et al. Smash: one-shot model architecture search through hypernetworks. 2017. ArXiv:1708.05344

16 Abadi M, Chu A, Goodfellow I, et al. Deep learning with differential privacy. In: Proceedings of ACM SIGSAC Conference on Computer and Communications Security, 2016. 308–318

17 Mo K, Huang T, Xiang X. Querying little is enough: model inversion attack via latent information. In: Proceedings of International Conference on Machine Learning for Cyber Security, 2020. 583–591

18 Lin W, Xu Y, Liu B, et al. Contribution-based federated learning client selection. Int J Intell Sys, 2022, 37: 7235–7260

19 Ma J, Naas S, Sigg S, et al. Privacy-preserving federated learning based on multi-key homomorphic encryption. Int J Intell Sys, 2022, 37: 5880–5901

20 Zhao B, Mopuri K R, Bilen H. IDLG: improved deep leakage from gradients. 2020. ArXiv:2001.02610

21 Wang Z, Song M, Zhang Z, et al. Beyond inferring class representatives: user-level privacy leakage from federated learning. In: Proceedings of IEEE INFOCOM 2019-IEEE Conference on Computer Communications, 2019. 2512–2520

22 Geiping J, Bauermeister H, Dröge H, et al. Inverting gradients — how easy is it to break privacy in federated learning? In: Proceedings of Advances in Neural Information Processing Systems, 2020. 16937–16947

23 Zhu J, Blaschko M. R-GAP: recursive gradient attack on privacy. 2020. ArXiv:2010.07733

24 Mo F, Borovykh A, Malekzadeh M, et al. Layer-wise characterization of latent information leakage in federated learning. 2020. ArXiv:2010.08762

25 Chen D, Yu N, Zhang Y, et al. GAN-leaks: a taxonomy of membership inference attacks against generative models. In: Proceedings of ACM SIGSAC Conference on Computer and Communications Security, 2020. 343–362

26 Leino K, Fredrikson M. Stolen memories: leveraging model memorization for calibrated white-box membership inference. In: Proceedings of the 29th USENIX Security Symposium (USENIX Security 20), 2020. 1605–1622

27 Rezaei S, Liu X. On the difficulty of membership inference attacks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021. 7892–7900

28 Truex S, Liu L, Gursoy M E, et al. Towards demystifying membership inference attacks. 2018. ArXiv:1807.09173

29 Jia J, Salem A, Backes M, et al. MemGuard: defending against black-box membership inference attacks via adversarial examples. In: Proceedings of ACM SIGSAC Conference on Computer and Communications Security, 2019. 259–274

30 Hu H, Salcic Z, Sun L, et al. Membership inference attacks on machine learning: a survey. 2021. ArXiv:2103.07853

31 Ai S, Koe A S V, Huang T. Adversarial perturbation in remote sensing image recognition. Appl Soft Comput, 2021, 105: 107252

32 Guo Y, Jiao B, Tan Y, et al. A transfer weighted extreme learning machine for imbalanced classification. Int J Intell Sys, 2022, 37: 7685–7705

33 Goodfellow I J, Shlens J, Szegedy C. Explaining and harnessing adversarial examples. 2014. ArXiv:1412.6572

34 Dong Y, Liao F, Pang T, et al. Boosting adversarial attacks with momentum. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2018. 9185–9193

35 Madry A, Makelov A, Schmidt L, et al. Towards deep learning models resistant to adversarial attacks. 2017. ArXiv:1706.06083

36 Liu J, Zhang Q, Mo K, et al. An efficient adversarial example generation algorithm based on an accelerated gradient iterative fast gradient. Comput Stand Interfaces, 2022, 82: 103612

37 Huang T, Zhang Q, Liu J, et al. Adversarial attacks on deep-learning-based SAR image target recognition. J Netw Comput Appl, 2020, 162: 102632

38 Wang J, Yin Z, Jiang J, et al. Attention-guided black-box adversarial attacks with large-scale multiobjective evolutionary optimization. Int J Intell Sys, 2022, 37: 7526–7547

39 Chen C, Huang T. Camdar-ADV: generating adversarial patches on 3D object. Int J Intell Syst, 2021, 36: 1441–1453

40 Ren H, Huang T, Yan H. Adversarial examples: attacks and defenses in the physical world. Int J Mach Learn Cyber, 2021, 12: 3325–3336

41 Buckman J, Roy A, Raffel C, et al. Thermometer encoding: one hot way to resist adversarial examples. In: Proceedings of International Conference on Learning Representations, 2018

42 Xie C, Wu Y, Maaten L, et al. Feature denoising for improving adversarial robustness. In: Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019. 501–509

43 Tramér F, Kurakin A, Papernot N, et al. Ensemble adversarial training: attacks and defenses. 2017. ArXiv:1705.07204

44 Wistuba M, Rawat A, Pedapati T. A survey on neural architecture search. 2019. ArXiv:1905.01392

45 Wang Y, Xue H, Liu Y, et al. Statistical network protocol identification with unknown pattern extraction. Ann Telecommun, 2019, 74: 473–482

46 Wang Y, Meng W, Li W, et al. Adaptive machine learning-based alarm reduction via edge computing for distributed intrusion detection systems. Concurr Comput Pract Exper, 2019, 31:

47 Jiang N, Tian F, Li J, et al. MAN: mutual attention neural networks model for aspect-level sentiment classification in SIoT. IEEE Int Things J, 2020, 7: 2901–2913

48 Guo X, Liu Z, Li J, et al. VeriFL: communication-efficient and fast verifiable aggregation for federated learning. IEEE Trans Inform Forensic Secur, 2020, 16: 1736–1751

49 Tang W, Li B, Barni M, et al. An automatic cost learning framework for image steganography using deep reinforcement learning. IEEE Trans Inform Forensic Secur, 2020, 16: 952–967

50 Huang Y, Lv S, Liu Z, et al. Cetus: an efficient symmetric searchable encryption against file-injection attack with SGX. Sci China Inf Sci, 2021, 64: 182314

Appendix A Attack settings in ADRA

The searchable operations are 3×3 and 5×5 separable convolutions, 3×3 and 5×5 dilated separable convolutions, 3×3 max pooling, 3×3 average pooling, identity, and zero. Internal model weight θ is not available to the adversary. Thus, in ADRA, θ is randomly initialized from a uniform distribution $\mathcal{U}(-0.25, 0.25)$. We use elu ($\sigma(x) = \max\{\mathrm{e}^x - 1, x\}$) as the activation function. The architecture search paradigm for both ADRA and AMIA blow is based on DARTS[2].

Appendix B Proof of Theorem 1

Considering the law of total expectation and Bayes' theorem, we have

$$\mathbb{P}\left(\epsilon_{\mathcal{D}_i} = 1 \mid \alpha, \theta\right) = \mathbb{E}_{\mathcal{D}_r}\left[\mathbb{P}\left(\epsilon_{\mathcal{D}_i} = 1 \mid \alpha, \theta, \mathcal{D}_r\right)\right] = \mathbb{E}_{\mathcal{D}_r}\left[\frac{\mathbb{P}\left(\epsilon_{\mathcal{D}_i}=1\right)\mathbb{P}\left(\alpha|\epsilon_{\mathcal{D}_i}=1,\theta,\mathcal{D}_r\right)}{\mathbb{P}(\alpha|\theta,\mathcal{D}_r)}\right].$$

Let $\lambda = \mathbb{P}(\epsilon_{\mathcal{D}_r} = 1)$, and substitute $\mathbb{P}(\alpha|\theta, \mathcal{D}_r) = \lambda\mathbb{P}(\alpha|\epsilon_{\mathcal{D}_i} = 1, \theta, \mathcal{D}_i) + (1-\lambda)\mathbb{P}(\alpha|\epsilon_{\mathcal{D}_i} = 0, \theta, \mathcal{D}_r)$, we have

$$\mathbb{P}\left(\epsilon_{\mathcal{D}_i} = 1 \mid \alpha, \theta\right) = \mathbb{E}_{\mathcal{D}_r}\left[\left(1 + \left(\frac{1-\lambda}{\lambda}\right)\frac{\mathbb{P}\left(\alpha|\epsilon_{\mathcal{D}_i}=1,\theta,\mathcal{D}_r\right)}{\mathbb{P}\left(\alpha|\epsilon_{\mathcal{D}_i}=0,\theta,\mathcal{D}_r\right)}\right)^{-1}\right].$$

Appendix C Attack settings in AMIA

The attack is implemented with Python 3.7 and 4 NVIDIA Tesla V100 GPUs. We adopt a publicized image classification model, Resnet-50, as the shadow model $F^s_{\mathcal{A},\theta}(\cdot)$, since the attacker is assumed to have black-box access only. Notably, only one shadow model is employed, as the number of shadow models does not contribute to the attack accuracy. The attack model $V(\cdot)$ is generated on a 3-layer feedforward neural network, trained with SGD.

Appendix D Defense settings

For Model Distillation, our implantation is based on DMP[3]. In particular, a private training dataset and an unlabeled reference dataset are required. Then, an unprotected teacher model is trained and leveraged to label data instances in the unlabeled reference dataset. DMP selects data instances in the labeled reference dataset that have low prediction entropy to train the target model. In this way, the protected classifier's direct access to the private training dataset is restricted, thus reducing the membership privacy leakage.

2) https://github.com/quark0/darts.
3) https://github.com/vrt1shjwlkr/AAAI21-MIA-Defense.