Defending Data Inference Attacks Against Machine Learning Models by Mitigating Prediction Distinguishability

Ziqi Yang, Yiran Zhu, Jie Wan, ChuXiao Xiang, Tong Tang, Yilin Wang, Ruite Xu, Lijin Wang, Fan Zhang, *Member, IEEE*, Jiarong Xu, and Zhan Qin

***Abstract*—Neural networks are vulnerable to data inference attacks, including the membership inference attack, the model inversion attack, and the attribute inference attack. In this paper, we propose PURIFIER to defend against membership inference attacks by quantifying the differences between dataset members and non-members in three dimensions: individual shape, statistical distribution, and prediction label. PURIFIER involves transforming the confidence scores produced by the target classifier, resulting in purified confidence scores that are indistinguishable across the dimensions above. We conduct experiments on widely-used datasets and models. The results show that PURIFIER offers robust defense against membership inference attacks with superior efficacy compared to prior defense techniques while maintaining minimal utility degradation (e.g., less than 0.7% classification accuracy drop of most datasets). Additionally, our extended experiments explore the effectiveness of PURIFIER in defending against the model inversion attack and the attribute inference attack.**

***Index Terms*—Data privacy, machine learning security, membership inference.**

I. INTRODUCTION

MACHINE learning offers significant convenience in daily life. However, its use of private information also poses potential risks of data breaches. Typically, users are granted access to the APIs of service providers, which return a confidence score vector or a label from the output of the machine learning model. However, many studies indicate that the predicted information can be exploited by adversaries to launch a data inference attack. These attacks aim to deduce private information about the data involved in the workflow of the target model [1], [2], [3]. Data inference attacks can be broadly divided into three categories: the Membership Inference Attack (MIA) [1], [4], [5], [6], [7], [8], [9], [10], [11], the model inversion attack [12] and the attribute inference attack [13]. In the MIA, the adversary is tasked with ascertaining whether a specific data sample belongs to the training data of the target model. The model inversion attack seeks to reconstruct the input data from the confidence scores generated by the target model. The attribute inference attack aims to deduce sensitive attributes hidden in training samples. The shared characteristic among them is to deduce data privacy information from publicly accessible outputs generated by the target model (such as confidence vectors).

In this paper, we utilize the MIA as a primary case study to explore the mitigation of data inference attacks. This is because extensive research has been conducted on the MIA in comparison to the other two attacks. It is widely acknowledged that the main factor contributing to the success of MIAs lies in the differences between prediction results for members and non-members. For example, a model that overfits the training dataset behaves more confidently when processing inputs from members than non-members. In general, the prediction differences between members and non-members can be observed in three primary aspects. (1) *Individual shape*. The confidence scores of members and non-members differ in their individual shapes, i.e., the distribution of confidence scores in an individual confidence vector. This is because the target model often assigns a higher probability to the predicted result when given a member than a non-member [4], [5]. (2) *Statistical distribution*. The statistical distribution of confidence scores for members and non-members within the same class is distinctly different. We find that confidence scores on the members are more concentrated in the encoded latent space, while those on non-members are more dispersed. BlindMI [9] exploits such statistical difference to infer membership by comparing the distance variation of the confidence scores of two generated datasets. (3) *Prediction label*. The confidence score differences between members and non-members can result in prediction label discrepancies. Member samples have a higher probability of being correctly predicted than the non-member samples, which leads to a difference in

Received 5 June 2023; revised 29 October 2024; accepted 12 December 2024. Date of publication 23 December 2024; date of current version 15 May 2025. This work was supported in part by National Natural Science Foundation of China under Grant 62102353 and in part by the National Natural Science Foundation of China under Grant 62072398. *(Corresponding author: Fan Zhang.)*

Ziqi Yang and Zhan Qin are with the State Key Laboratory of Blockchain and Data Security, Zhejiang University, Hangzhou 310027, China, also with the Hangzhou High-Tech Zone (Binjiang) Institute of Blockchain and Data Security, Hangzhou 310027, China (e-mail: yangziqi@zju.edu.cn; qinzhan@zju.edu.cn).

Yiran Zhu, Jie Wan, ChuXiao Xiang, Tong Tang, Yilin Wang, Ruite Xu, and Lijin Wang are with the State Key Laboratory of Blockchain and Data Security, Zhejiang University, Hangzhou 310027, China (e-mail: zhuyiran@zju.edu.cn; wanjie@zju.edu.cn; 22221058@zju.edu.cn; tong.tang@zju.edu.cn; 22221052@zju.edu.cn; xuruite@zju.edu.cn; wanglijin@zju.edu.cn).

Fan Zhang is with the College of Computer Science and Technology, Zhejiang University, Hangzhou 310027, China (e-mail: fanzhang@zju.edu.cn).

Jiarong Xu is with the Department of Information Management and Business Intelligence, Fudan University, Shanghai 200437, China (e-mail: jiarongxu@fudan.edu.cn).

Code is available at https://github.com/zyyrrr/purifier.

Digital Object Identifier 10.1109/TDSC.2024.3518698

classification accuracy. Various label-only attacks exploit this distinguishability to perform their attacks [6], [7], [8].

Several methods have been proposed to defend against the MIA while preserving the confidence scores, such as L_2 regularizer [1], dropout [5], model-stacking [5], min-max regularization [4] and differential privacy [14], [15], [16]. MemGuard [17] further enhances the preservation by enforcing the confidence score vectors to stay within a small distortion around the original confidence score vectors generated by classifiers without any defense. Relax Loss [18] reduces the difference between members and non-members by relaxing the loss of member samples. On the other hand, some studies believe that removing the confidence information in the prediction result is a way of defending against the membership inference attack. However, these defenses are compromised by label-only inference attacks [6], [7], [8], where membership is inferred only from the predicted label.

In this paper, we propose a defense mechanism, namely PURIFIER, to defend against MIAs. This method utilizes the confidence scores generated by the target model as its input and subsequently produces transformed confidence scores. These transformed scores exhibit indistinguishable characteristics among members and non-members in terms of *individual shape*, *statistical distribution*, and *prediction label*. More specifically, (1) to purify individual shapes, we propose a strategy to train a module named *confidence reformer*. The confidence reformer is trained on the confidence scores predicted by the target model on non-members, which allows the model to learn the individual shape of confidence scores on non-members. The confidence scores of members become indistinguishable from those of non-members after being transformed by the confidence reformer. (2) To purify the statistical distribution, we introduce a Conditional Variational Auto-Encoder (CVAE) into the confidence reformer to add Gaussian noises to confidence scores. The Gaussian noises disperse the initially statistically clustered confidence scores, thereby obscuring the distinction between members and non-members within a statistical distribution. (3) To purify prediction labels, we propose a mechanism named *label swapper*. To counteract the label-only attack that exploits the classification accuracy gap between members and non-members, the label swapper alters the prediction labels on members to another class at a specially designed rate. To enhance the robustness of PURIFIER, we introduce the k-Nearest Neighbor (kNN) method in the label swapper to tolerate small perturbations added by the attacker.

Experimental results underscore the efficacy of PURIFIER in defending against MIAs, the model inversion attack, and the enhanced attribute reasoning attack. The results are validated on 7 widely used datasets, including CIFAR10, Purchase100, Facescrub530, CIFAR100, Texas, Location, and UTKFace. Compared with existing defense methods, PURIFIER exhibits superior overall defensive efficacy against data inference attacks. For instance, when utilizing PURIFIER, the member inference accuracy of the NSH attack on FaceScrub530 is reduced by 0.79% to 14.67% compared to other defense methods. The reconstruction error of the model inversion attack reaches the highest when employing the PURIFIER defense. Furthermore, under the protection of PURIFIER, the enhanced attribute inference attack is completely ineffective. By comparing the individual shape and statistical distribution visualization results before and after applying PURIFIER, we observe that PURIFIER transforms the confidence vector features of members towards those of non-members, consequently reducing the distinguishability between members and non-members.

Contributions: In summary, the contributions of this paper are as follows.

1) To the best of our knowledge, we are the first to study MIAs from three distinct aspects: *individual shape*, *statistical contribution*, and *prediction label*.
2) We design PURIFIER to defend against data inference attacks. This system comprises two main components: the label swapper and the confidence reformer. By transforming confidence scores, PURIFIER achieves indistinguishability between members and non-members in the above three aspects.
3) Based on our comprehensive experiments, PURIFIER demonstrates superior performance in effectiveness when defending against data inference attacks compared to other defense techniques.

II. INFERENCE ATTACKS ON MACHINE LEARNING

Machine learning has been demonstrated to be susceptible to a variety of inference attacks [1], [6], [12], [19], which allow adversaries to extract valuable information about the target model from prediction APIs alone. Depending on the inference goals, these inference attacks can generally be categorized into two types: *model inference* and *data inference*. Model inference seeks to acquire knowledge about the target model itself, such as its parameters and architecture [20], [21], [22], [23]. In contrast, data inference focuses on extracting information about the data that the target model processes [1], [2], [6], [12], [19], [24], [25], [26]. In this paper, we concentrate on three significant and representative data inference attacks: membership inference attack, model inversion attack, and attribute inference attack. In this section, we first introduce these three data inference attacks, followed by a discussion of existing defense strategies. Finally, we analyze the limitations of current defense mechanisms.

A. Data Inference Attacks

Membership inference, model inversion, and attribute inference attacks are three widely-studied types of data inference attacks that threaten the security and privacy of machine learning. They differ in their inference goals.

Membership Inference Attack: In the MIA, the attacker is asked to determine whether a specific data record constitutes part of the target model's training dataset.

Confidence-Based Attack [1], [5]: Shokri et al. [1] present the MIA on black-box models, where the attacker has access only to the confidence scores $F(z)$ of the target model F for a given data sample z. To infer the membership, the attacker trains a binary classifier A (also referred to as an attack model) that takes $F(z)$ as input on z and predicts whether z is a member (Label 1) or non-member (Label 0) of the training dataset D_{train} of F, i.e.,

$A(F(z)) \rightarrow \{0, 1\}$. Before training A, the attacker trains a set of shadow models $\widetilde{F}$ on an auxiliary dataset drawn from the same data distribution as the D_{train} to replicate F. A is then trained on the confidence scores $\widetilde{F}(z)$ predicted by the shadow models instead of the target model on the members and non-members of the shadow models' training data. Salem [5] introduces the Mlleaks attack, demonstrating that only one shadow model $\widetilde{F}$ is required to train the attack model.

Label-Only Attack [7]. The label-only attack is a form of black-box MIA that operates solely on the output label of the target model, rather than its confidence scores. Li et al. [7] propose three distinct types of label-only attacks. In the Gap Attack, it is assumed that the attacker possesses the ground truth of the data sample and predicts membership based solely on whether or not F correctly labels the sample. The Transfer attack involves relabeling an auxiliary dataset by querying F, thereby enabling the adversary to train a shadow model $\widetilde{F}$ for launching a score-based MIA locally. However, in the Boundary attack, the auxiliary dataset is not accessible. The adversary utilizes adversarial noises to perturb the input to mislead F and identifies samples that exceed a certain threshold as members.

Confidence & Label-Based Attack [4]. Nasr et al. propose the NSH attack, which trains three distinct attack models (A_1, A_2, A_3) to infer the membership of the data samples. Specifically, A_1 and A_2 utilize the confidence score and label information as their respective inputs. A_3 accepts the outputs from both A_1 and A_2 to predict the membership of a given data sample. The attacker is assumed to possess knowledge of the membership labels, enabling them to query the target classifier for confidence scores of members and non-members without training the shadow models $\widetilde{F}$.

Distribution Difference-Based Attack: The BlindMI attack [9] probes the target model and extracts membership semantics via differential comparison, eliminating the need for shadow models. Specifically, BlindMI initially generates a dataset of non-members, S_{nonmem}, by transforming existing samples into new ones. It then differentially moves samples from a target dataset, S_{target}, to S_{nonmem} in an iterative manner. If the differential move of a sample results in an increase in the set distance, BlindMI classifies the sample as a non-member; conversely, if the set distance decreases, the sample is considered a member.

Confidence & Distribution-Based Attack: Carlini et al. [10] present MIA from First Principles and introduce an alternative evaluation metric, suggesting that MIAs should be assessed by computing their true-positive rate at low false-positive rates (e.g., $<0.1\%$). They discover that the majority of previous attacks perform poorly under this criterion. To meet this evaluation standard, they propose a method that trains N shadow models $\widetilde{F}$ on random samples from an auxiliary dataset. Half of these models are trained on the target sample, while the other half are not (referred to as IN and OUT models). Membership is inferred by comparing confidence scores on the target sample from the target model with those from both the IN and OUT models. Ye et al. [11] introduce Enhanced MIA, a comprehensive hypothesis-testing framework that not only allows for the consistent formalization of prior work but also supports the design of novel MIAs. These attacks employ reference models to achieve a significantly higher power (true positive rate) for any error (false positive rate). They propose four attacks, namely S, P, R, and D, each characterized by varying settings on the boundary between members and non-members.

Model Inversion Attack: Model inversion aims to reconstruct the original input data from the confidence scores predicted by the target model. Fredrikson et al. [2] introduce a method to infer a representative sample from a training class against a white-box target model. It casts the inversion task as an optimization problem in the input domain to identify the most suitable representative for a given class. Yang et al. [12] propose a model inversion attack in the black-box setting. Specifically, they train an inversion model G_θ on an auxiliary dataset that is the inverse of the target model T. G_θ takes the confidence scores of T as input and its goal is to reconstruct the original input data.

Attribute Inference Attack: Attribute inference aims to infer sensitive attributes [19], [26], [27], [28] or statistical information [24] about the training dataset. For example, the race attribute of the samples can be inferred with an API for the gender classifier C_{gender} on the UTKFace dataset [13]. The adversary has an auxiliary dataset and queries the target model to obtain the confidence scores. Subsequently, the adversary trains a classifier C_{race} on these confidence scores, using the race attribute as the label. Thus the race of a sample can be predicted with the output of C_{gender}.

B. Defenses Against Data Inference Attacks

Previous defenses against data inference attacks have mostly focused on mitigating MIAs. However, there has been a notable lack of research addressing the simultaneous defense against the model inversion attack and the attribute inference attack on classification models. Consequently, we present existing defenses against MIAs as representative examples in the literature that defend against data inference attacks.

Min-Max Game [4]: Nasr et al. suggest the incorporation of an adversarial regularizer into the loss function of the target model. This is designed to ensure that the model not only minimizes prediction loss but also maximizes membership privacy. The training procedure is conceptualized as a min-max optimization problem.

$$\min_f \left(L(f) + \lambda \max_h G_f(h) \right) \tag{1}$$

where f indicates the target model, h represents the inference model, and the regularization factor λ controls the balance between the classification loss function L and the regularizer G.

MemGaurd [17]: Jia et al. propose to post-process the confidence score vector by converting it into an adversarial example to fool the membership classification of the attack model A. In particular, the defender injects carefully designed noise into the confidence score vector predicted by the target model T to form an adversarial example, controlled by several hyper-parameters c_1, c_2, c_3. To achieve this goal, the defender first trains its attack model A'. Thus, T can generate the adversarial example against A' in a white-box manner.

Model Stacking [5]: Model stacking fundamentally employs an ensemble method, which combines multiple simple classifiers with a complicated one to generate the final prediction. This technique is frequently utilized to reduce overfitting and can be employed to mitigate MIAs.

MMD defense [8]: Li et al. introduce a defense strategy designed to bridge the gap between members and non-members by deliberately diminishing the training accuracy. The training procedure strives to align the training and validation accuracies, employing a new set regularizer derived from the Maximum Mean Discrepancy (MMD) between the softmax output empirical distributions of the training and validation datasets. MMD is defined as:

$$Distance(X, Y) = \left\| \frac{1}{n}\sum_{i=1}^{n}\Phi(x_i) - \frac{1}{m}\sum_{j=1}^{m}\Phi(y_j) \right\|_H \quad (2)$$

where H is a universal RKHS [29], $\Phi : X \to H$ is a Gaussian kernel [30] mapping function, each x_i (y_j) is the softmax output of the i-th training (validation) instance.

SELENA [31]: Tang et al. introduce a novel framework for training privacy-preserving models that generate similar behaviors on member and non-member inputs, thereby mitigating MIAs. This framework, SELENA, comprises two primary components. The first is an ensemble architecture for training, called Split-AI. This architecture partitions the training data into random subsets and trains a model on each subset. An adaptive inference strategy is employed during testing. The ensemble architecture then aggregates the outputs of those models that do not include the input sample in their training data. Let F denote Split-AI, it can be calculated as $F = \frac{1}{L}\sum_{i \in Id_{non}(x)} F_i(x)$, where x is a training sample, L is the number of sub-models that are not trained with x, F_i is a sub-model, and Id_{non} represents the set of L non-model indices for each x. The second component, Self-Distillation, uses the same training set as well as the predicted confidence vectors from Split-AI as soft labels to train a distillation model that serves as the defense model.

Relax Loss [18]. Current research underscores a significant correlation between the distinguishability of training (i.e., member) and testing (i.e., non-member) loss distributions and the model's susceptibility to MIAs. Drawing inspiration from these findings, Chen et al. introduce a training framework that employs a relaxed loss with a more attainable learning objective. The framework mainly makes two changes during the target model training. First, the loss is relaxed using gradient ascent to elevate the loss of member data to a level that is attainable by non-member data:

$$\theta \leftarrow \theta + \tau \cdot \nabla\mathcal{L}(\theta) \quad (3)$$

where θ is model parameters, τ denotes the learning rate. Then, to mitigate the side-effect of the utility caused by relaxed loss, they flatten the target posterior scores for non-ground-truth classes and compute the cross entropy loss based on the new soft label. The posterior flattening is defined as:

$$t_i^c = \begin{cases} p_i^c & \text{if } y_i^c = 1 \\ (1 - p_i^c)/(C-1) & \text{otherwise} \end{cases} \quad (4)$$

where C is the number of output classes, p_i is the original model softmax label output of the sample and t_i stands for the flattened soft label.

C. Limitations of Existing Defenses

Previous research on defense mechanisms against the MIA has not discussed their implications for the model inversion attack and attribute inference attack. The latter two attacks represent a significant threat to the security and privacy of machine learning data. To our knowledge, there are currently no known defense methods that effectively counteract membership inference, model inversion, and attribute inference attacks.

Overfitting is not the sole cause of MIAs [1]. Even when various machine learning models are similarly overfitted, they may reveal differing amounts of membership information. This discrepancy can be attributed to their unique structures, which may "recall" varying degrees of information about their training datasets. The attacker exploits the target model's confidence score distinctions between members and non-members to launch an MIA [1]. Current defense mechanisms, such as those aimed at reducing overfitting, contribute to this reduction in distinguishability. However, these defense strategies could be more effective if the distinguishability itself were directly reduced. However, certain existing methods exhibit suboptimal performance. For instance, Relax Loss requires retraining the target model, and SELENA needs to train multiple shadow models, which means greater computational overhead.

III. Problem Overview

In this section, we first introduce three roles and their capabilities involved in the data inference attack and defense problem. Then, we present our motivation for defense.

A. Roles & Capabilities

We consider the classification models of neural networks. A machine learning classifier F is trained on its training dataset D_{train} to map a sample $\mathbf{x}$ to a class based on the confidence vectors $F(\mathbf{x})$. In our problem, there are three parties: *model owner*, *attacker* and *defender*.

1) Model Owner: The model owner trains a machine learning classifier F on its training dataset D_{train} which is drawn from some underlying data distribution $p_x(\mathbf{x})$. The classifier F is trained with the goal of making predictions on unseen data which we refer to as test dataset D_{test}. Let $\mathbf{x}$ represent the data drawn from p_x, and $\mathbf{y}$ be the vectorized class of $\mathbf{x}$. The training objective is to find a function F to well approximate the relation between each data point $(\mathbf{x}, \mathbf{y})$. Formally, we have $F : \mathbf{x} \mapsto \mathbf{y}$. The training process is to optimize an objective function $L(F)$. The model owner releases the trained classifier F as a black box, for example, as a cloud service, and provides prediction APIs to users. The users can query F with their own data sample $\mathbf{x} \in D_{test}$ through the prediction APIs. The classifier F returns a confidence score vector $F(\mathbf{x})$ to the users. The confidence score vector is a probability distribution of the classifier's confidence over all the possible classes. For example, the i-th element $F(\mathbf{x})_i$

is the probability of the data $\mathbf{x}$ belonging to class i. We usually take the class with the maximum probability to be the predicted label y of the data $\mathbf{x}$.

2) Attacker: The attacker aims at performing data inference attacks against the target classifier F. We assume that the classifier F works as a black-box "oracle" to the attacker, i.e., the attacker can only query F with a sample $\mathbf{x}$ and obtain the prediction scores $F(\mathbf{x})$ or the predicted label y. The attacker is also assumed to have an auxiliary dataset D_{aux}, which consists of data samples sampled from a similar data distribution as the training data distribution of F, in addition to a subset of the training set samples. Take the MIA as an example, the attacker is asked to determine whether a given data record $\mathbf{x}$ is part of the training data D_{train} according to $F(\mathbf{x})$. A common approach is to leverage the auxiliary dataset D_{aux} and train a membership classifier that takes $F(\mathbf{x})$ as input and predicts the membership.

3) Defender: The defender could be the model owner or a third party with access to the target classifier's prediction results. The defender possesses a training dataset D_{train} and a reference dataset D_{ref}, composed of non-member data, to implement the defense. For any query to the target classifier from users, the defender modifies the prediction results of the target classifier before returning it to users. The attacker has access only to the modified prediction results from the defender. In particular, the defender wants to achieve the following three goals:

Defense: The defender aims at defending the membership inference attack, model inversion attack, and attribute inference attack. Specifically, the defender wants to reduce the membership and attribute classification accuracy and increase the reconstruction error of the input sample performed by the attacker.

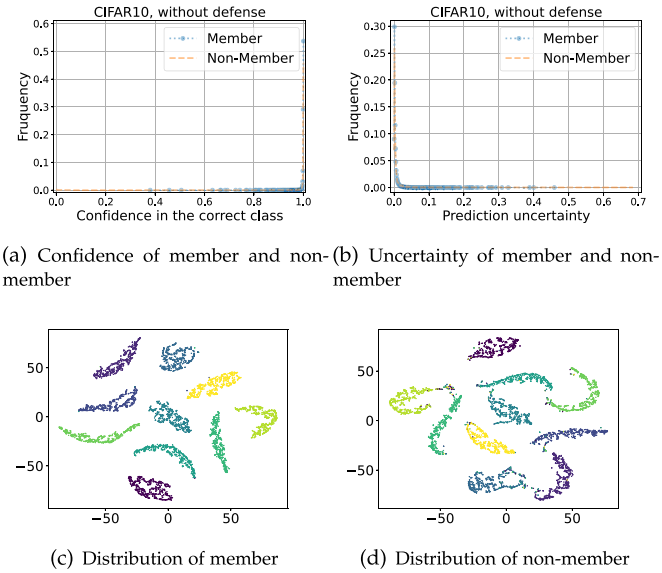
Utility: The classification accuracy on the test dataset D_{test} is one of the metrics to evaluate the utility of the model. The defender aims at defending the target model with the least loss of utility (i.e., the least reduction of classification accuracy on D_{test}).

Efficiency: The defense mechanism should incur acceptable overhead in the total training time and the test time of predicting a data sample.

B. Motivation

Membership inference attacks can be largely divided into three categories depending on the underlying distinguishability of confidence scores that they exploit: *individual shape*, *statistical distribution*, and *prediction label*. In this paper, to fairly evaluate the defense performance of our approach, we consider all three categories of membership inference attacks:

Individual shape: The distinguishability of individual shapes is exposed because members and non-members differ in the distribution of confidence scores. For example, the classifier usually predicts the class of member data with a high degree of confidence. This implies that the value of the maximal score in the vector would be larger for member samples compared to non-member samples. This can be illustrated by Fig. 1(a) and (b), where we calculate the frequency of confidence on members and non-members in CIFAR10 for correct class and prediction uncertainty. The prediction uncertainty is measured as the normalized entropy $\frac{-1}{\log(k)}\sum_i \hat{\mathbf{y}}_i \log(\hat{\mathbf{y}}_i)$ of the confidence vector $\mathbf{y} = F(\mathbf{x})$, where k is the number of classes. It can be seen that there are more non-members than members at lower confidence levels. This is because, without defense, the target model often exhibits overfitting, resulting in a higher training set accuracy compared to the test set accuracy. Conversely, at higher confidence levels, members outnumber non-members. In addition, the distribution of members tends to predict low values of uncertainty compared to non-members.



(a) Confidence of member and non-member (b) Uncertainty of member and non-member

(c) Distribution of member (d) Distribution of non-member

Fig. 1. Individual and statistical distinguishability between members and non-members in CIFAR10.

Statistical distribution: The distinguishability of statistical distribution implies that confidence scores on samples from the same class are more similar than those on non-member samples. Conversely, the confidence scores of samples from different classes exhibit less similarity compared to non-member samples. In an intuitive way, we reduce the dimensionality of the latent vectors corresponding to these confidence scores and plot them in a coordinate graph, as shown in Fig. 1(c) and (d). We can see that members of the same class cluster more closely together than non-members, while members of different classes are distinctly separated from each other.

Prediction label: The prediction label for a sample is the class corresponding to the maximal score in the confidence vector. The distinguishability of prediction labels directly leads to the difference between training accuracy and testing accuracy, where the former is higher in general. For example, the training accuracy of the classifier on CIFAR10 is 99.99%, while the testing accuracy is 95.92% in our experiment.

IV. Approach: Purifier

We propose PURIFIER as a defense against data inference attacks. The main idea is to alter the distribution of the confidence score vector $F(\mathbf{x})$ so that it appears indistinguishable between members and non-members and hinders the reconstruction of the

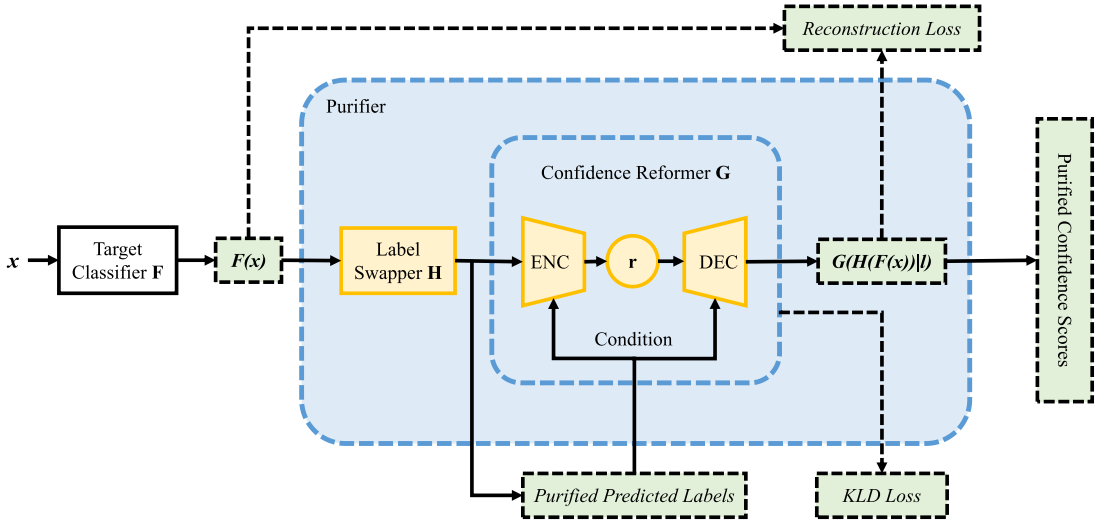


Fig. 2. Architecture of PURIFIER. PURIFIER consists of a *label swapper* H and a *confidence reformer* G. H can reduce the gap of classification accuracy between members and non-members by modifying the predicted labels of specific training data. G is a Conditional Variational Auto-encoder (CVAE), with the purified predicted label through H as the condition. G can reform the confidence scores by mapping $H(F(\mathbf{x}))$ to the latent space $\mathbf{r}$ with the encoder and mapping it back with the decoder.

input vector and the inference of sensitive attributes. PURIFIER consists of a *label swapper* H and a *confidence reformer* G, as shown in Fig. 2. The *label swapper* H takes the original confidence score vectors as input. It modifies the predicted labels of some members by swapping the maximum confidence score with another confidence score. This process aims to minimize the classification accuracy disparity between members and non-members, thereby achieving indistinguishability in the predicted labels. The *confidence reformer* G takes as input the modified confidence score vectors from H and reforms them as if they were generated from non-members, thereby ensuring indistinguishability of individual shape and statistical distribution.

A. Design of Label Swapper

A notable discrepancy exists in the classification accuracy of predicted labels when comparing member and non-member data. The adversary may exploit this disparity to distinguish between the two. To address this problem, we design a mechanism named *label swapper* in the inference stage. The *label swapper* H is used to modify the predicted labels of members to reduce the classification accuracy gap between members and non-members. As illustrated in (5), H randomly selects training samples to substitute their predicted labels with another predicted score at a certain swap rate p_{swap}, which is determined by the disparity in classification accuracy between member data and non-member data.

$$p_{swap} = \frac{acc_{train} - acc_{test}}{acc_{train}} \tag{5}$$

where acc_{train} and acc_{test} are the training accuracy and the test accuracy of the target classifier respectively. Note that H determines whether the input sample is a member or a non-member and only performs label swapping on member data. At the swap rate p_{swap}, the training accuracy can be decreased to testing accuracy, achieving indistinguishability of the prediction label.

Algorithm 1: Pre-Process of *Label Swapper*.

Input: The training dataset D_{train}, the target classifier F, the swap rate p_{swap}, the number of classes n, the number of neighbors of **KNN** k, the distance precision of **KNN** d

Output: The function **knn** after initialization, the offset of the false label list L_{of}

$C = \varnothing$;
for $(\boldsymbol{x}_{train_i}, y_{train_i}) \in D_{train}$ **do**
  $\mathbf{c}_i = F(\mathbf{x}_{train_i})$;
  $l_i = \mathbf{argmax}(\mathbf{c}_i)$;
  $C = C \bigcup \{\mathbf{c}_i\}$;
end
knn $= \mathbf{KNN}(C, k, d)$;
$L_{of} = \{o_i | o_i = \text{rand from}(0, n), \text{where } 0 \leq i < p_{swap} * ||D_{train}||\}$;
return knn, L_{of}

A naive implementation is as follows. When the input confidence vector corresponds to the member data, H swaps its label with a probability p_{swap}. However, this approach is susceptible to a replay attack. As the model repeatedly predicts the same member sample, it may produce different results. This indicates that the input is a member sample and exposes a more severe distinguishability between members and non-members.

To mitigate the replay attack, we do not use p_{swap} as the swap probability. Instead, we number each member sample from 0 to $||D_{train}||$-1. For the member with a number i in the range of $[0, p_{swap} * ||D_{train}||]$, we randomly generate an offset o_i that is less than the number of classes n, and exchange the confidence corresponding to its label with the confidence of the class after

the offset o_i. This approach ensures that the predicted results for the same member sample remain consistent.

In addition, we also consider another potential attack, where adversaries infer whether a certain sample $\mathbf{x} \in D_{train}$ is the member or non-member by adding small perturbations to the input data and observing changes in the predicted result. To defend against this attack, we introduce the kNN method as a component of the label swapper to identify suspicious noisy members. Specifically, if the Euclidean distance between the confidence vector of a sample and those of its nearest k neighboring members is less than a certain threshold parameter d, the sample is classified as a member. Furthermore, if the member number i corresponding to this sample satisfies $i < p_{swap} * ||D_{train}||$, its label should be swapped. In the inference stage, we query kNN with the confidence vector of the input sample and get its corresponding ground truth label.

Algorithm 1 presents the preprocess of label swapper. We first select the data $(\mathbf{x}_{train_j}, y_{train_j})$ from D_{train} at rate p_{swap} randomly to form D_{swap}. After that, we query the target classifier F to get the confidence scores $\mathbf{c}_j$ of the sample $(\mathbf{x}_{train_j}, y_{train_j}) \in D_{swap}$. Then, we store the acquired confidence vector $\mathbf{c}_j$ in confidence vector set C. kNN is initialized with C, k, d. L_{of} is initialized with $p_{swap} \times ||D_{train}||$ as the integer number of random offset ranging from $(0, n)$.

B. Design of Confidence Reformer

To achieve both individual and statistical indistinguishability between members and non-members, PURIFIER reforms the confidence scores with the *confidence reformer* G, which is a CVAE.

In the training stage, G is trained on $F(\mathbf{x})$, where $\mathbf{x}$ belongs to a reference dataset D_{ref}. D_{ref} and D_{train} are drawn from the same distribution but have no overlap, which means that D_{ref} consists of non-member samples. Formally, G is trained to minimize the objective function defined in (6).

$$L(G) = \mathop{\mathbb{E}}_{\mathbf{x}\sim p_r(\mathbf{x})}[\mathcal{L}_{rec} + \mathcal{L}_{kld}] \tag{6}$$

where $p_r(\mathbf{x})$ represents the conditional probability of $\mathbf{x}$ for samples in D_{ref}, $\mathcal{L}_{rec}$ is the reconstruction loss function (i.e., **MSE** loss), which is defined in (7), $\mathcal{L}_{kld}$ is the Kullback-Leibler divergence loss function, which is defined in (8).

$$\mathcal{L}_{rec} = \mathcal{R}((G(F(\mathbf{x})|l), F(\mathbf{x})) \tag{7}$$

where l represents the predicted label of $\mathbf{x}$ by F.

$$\mathcal{L}_{kld} = -0.5 \times \sum_{i=1}^{D} \left(1 + \log\left(\sigma_i^2\right) - \mu_i^2 - \sigma_i^2\right) \tag{8}$$

where μ and σ^2 are the mean and variance parameters in the latent space of G.

The training process of *confidence reformer* is summarized in Algorithm 2. For each epoch, we first draw a batch of data samples $\{(\mathbf{x}_j, y_j)\}_{j=1}^{q}$ from the reference set D_{ref}. Then we query the target classifier F to obtain the confidence scores $\mathbf{c}_j$, and the predicted label l_j. After that, the loss is calculated on the objective function 6, and gradient descent is used to update

Algorithm 2: Training Process of *Confidence Reformer*.

Input: The reference dataset D_{ref}, the target classifier F, size of batch q, number of epochs P, learning rate η, label loss coefficient λ
Output: *confidence reformer* G_θ

$\theta = \mathbf{initialize}(G_\theta)$;
for $p = 1$ *to* P **do**
  for *each batch* $\{(\boldsymbol{x}_j, y_j)\}_{j=1}^{q} \subset D_{ref}$ **do**
    $\mathbf{c}_j = F(\mathbf{x}_j)$;
    $l_j = \mathbf{argmax}(\mathbf{c}_j)$;
    $g = \nabla_\theta \frac{1}{q} \sum_{j=1}^{q} (\mathcal{L}_{rec} + \mathcal{L}_{kld})$; ▷ refer Eq(6)
    $\theta = \mathbf{updateParameters}(\eta, \theta, g)$;
  end
end
return G_θ

the parameters θ of *confidence reformer* G. G is trained on D_{ref}, which consists of non-member samples. Consequently, G assimilates the characteristic pattern of non-member samples. In the inference stage, G processes the input confidence vector $H(\mathbf{c}_j)$, with the modified label $l' = \mathbf{argmax}(H(\mathbf{c}_j))$ as the condition. $H(\mathbf{c}_j)$ first traverses through the encoder, resulting in its mapping to the encoded latent space $\mathbf{r}$. Subsequently, the decoder maps the confidence vector back from the latent space $\mathbf{r}$, and obtains the reformed confidence vector $G(H(\mathbf{c}_j)|l')$. In this reforming process, the reconstruction loss $\mathcal{L}_{rec}$ encourages the decoder of G to generate confidence vectors that have a similar pattern as the non-member ones on D_{ref} (non-members) with the same label. Thus, the confidence vector of the member sample integrates the characteristics of the non-member samples. In general, G mitigates individual variations in $\mathbf{c}_j$, ultimately achieving individual indistinguishability.

In addition, to mitigate the difference in statistical distribution between members and non-members, *confidence reformer* G introduces Gaussian noises in the latent space $\boldsymbol{\mu}, \boldsymbol{\sigma}$, where the label l' is used as the condition. Despite the potential for these noises to amplify the reconstruction error, G adaptively learns a robust latent representation capable of preserving the statistical distribution of non-members of label l'. During the inference process, the added noises break down the clustering of confidence scores on members, while the decoder generates the reformed versions that are similar to the ones on D_{ref}, mitigating the difference in statistical distribution.

C. Defense Process of PURIFIER

Following the pre-processing by the label swapper and the training of the confidence reformer, we can perform defense with the *label swapper* H and the trained *confidence reformer* G. H is employed to determine whether a given confidence score belongs to a member sample and to execute label swapping. Then, G is responsible for the transformation of these confidence scores.

As shown in Algorithm 3, at the inference stage, given an input sample $\mathbf{x}$, we first query the target classifier F to obtain the confidence scores $\mathbf{c}$ and the predicted label l. Then, we feed $\mathbf{c}$

Algorithm 3: Inference Process of PURIFIER.

Input: The input sample $\mathbf{x}$, the target classifier F, the trained *confidence reformer* G_θ, the function $\mathbf{knn}(\cdot)$, a helper function $\mathbf{swap}(\cdot)$ that takes a vector as input and swap the largest element with another element, the offset of the false label list L_{of}

Output: The purified confidence score $\mathbf{c}_{purified}$

$\mathbf{c} \leftarrow F(\mathbf{x})$;
$l = \mathbf{argmax}(\mathbf{c})$;
$index = \mathbf{knn}(\mathbf{c})$; ▷ Start of label swapper
if $index < p_{swap} * ||D_{train}||$ **then**
  $offset = L_{of}[index]$;
else
  $offset = 0$;
end
$l^{'} = (offset + l)$ **mod** n;
$\mathbf{c}^{'} = \mathbf{swap}(\mathbf{c}, l, l^{'})$; ▷ End of label swapper
$\mathbf{c}_{purified} = G_\theta(\mathbf{c}^{'}|l^{'})$; ▷ process of CVAE
return $\mathbf{c}_{purified}$

into the label swapper H. H determines whether $\mathbf{c}$ is close to the confidence vector of a member sample by using kNN. If so, H regards it as the member sample and swaps the maximum score with another score. At this stage, $\mathbf{c}$ is indistinguishable in the prediction label. Next, we feed $\mathbf{c}$ into the *confidence reformer* G, where the condition is the label l', to obtain the purified confidence vector $\mathbf{c}_{purified}$. This ensures indistinguishability in terms of individual shape and statistical distribution. Finally, PURIFIER returns the purified confidence scores $\mathbf{c}_{purified}$.

V. EXPERIMENTS

In this section, we conduct experiments on different popular datasets and models to verify the effectiveness of PURIFIER. The experiments are conducted on a PC with four Titan XP GPUs (12GBytes graphic memory), 128 GBytes memory, and an Intel Xeon E5-2678 CPU. We first describe the dataset and model configurations in detail. We then conduct a series of experiments aimed at demonstrating that: (1) PURIFIER is effective in defending against MIAs, the model inversion attack, and the attribute inference attack, outperforming other defense methods. (2) With the defense provided by PURIFIER, we achieve individual, statistical, and label indistinguishability. (3) Furthermore, We evaluate the computation overhead of PURIFIER. (4) Finally, we conduct three additional experiments to explore the impact of the order of the label swapper and confidence reformer, the size of the reference set, and noisy member samples on the performance of PURIFIER.

A. Dataset and Model Settings

1) Dataset Settings: We use CIFAR10, Purchase100, FaceScrub530, CIFAR100, Texas, Location, and UTKFace datasets which are commonly used in previous works on data inference attacks.

TABLE I
DATA ALLOCATION

Dataset	D_1	D_2	D_3
CIFAR10	50,000	5,000	5,000
Purchase100	20,000	20,000	20,000
FaceScrub530	30,000	10,000	8,000
CIFAR100	50,000	5,000	5,000
Texas	10,000	10,000	10,000
Location	1,600	1,600	1,600
UTKFace	10,000	5,000	5,000

A dataset is divided into a training set D_1, a reference set D_2, and a test set D_3.

CIFAR10 [1], [5], [7], [10]: This dataset serves as a benchmark for evaluating image recognition algorithms in machine learning. It comprises 60,000 color images, each with dimensions of 32 × 32 × 3. The dataset is divided into ten classes, each class symbolizing an object (e.g., airplane, car, etc.).

Purchase100 [1], [4], [5], [7]: This dataset is derived from Kaggle's "acquired valued shopper" challenge.[1] We use the preprocessed and simplified version [1]. The dataset consists of 197,324 data records with each record having 600 binary features. The dataset is clustered into 100 classes.

FaceScrub530 [12]: The dataset contains URLs of 100,000 images of 530 people. We obtain the preprocessed and simplified version of this dataset from [12] which has 48,577 facial images and each image is resized to 64 × 64.

CIFAR100 [1], [10]: The dataset serves as a benchmark in machine learning for the evaluation of image recognition algorithms. It comprises 60,000 color images, each with dimensions of 32 × 32 × 3. The dataset is divided into 100 classes, with each class containing 600 images.

Texas [1], [32]: The data utilized in this study originates from the web link information of the University of Texas, USA. This dataset encompasses web pages and the hyperlink relationships that exist between them. We use the same dataset as previous works, which is clustered with 6,169 attributes into 100 classes and contains a total of 67,330 samples.

Location [1], [32]: This dataset is derived from the publicly accessible set of mobile users' location "check-ins" within the Foursquare social network, specifically limited to the Bangkok region and gathered between April 2012 and September 2013. The Location record encompasses 446 attributes, which have been clustered into 30 classes, comprising a total of 5,010 samples.

UTKFace [13], [33]: This dataset contains 20,606 images of individuals with age, gender, and race annotations. Each image is resized to 50 × 50. We train the classifier to predict the gender attribute and use the race attribute as the sensitive attribute in our experiments.

Table I presents the data allocation in our experiments. We divide each dataset into the training set D_1, the reference set D_2 and the test set D_3 randomly. They have no overlap with each other. Then, we randomly divide D_1 and D_3 into two halves, denoted as D_1^1, D_1^2, D_3^1, D_3^2. We repeatedly sample D_3^1, D_3^2

[1] https://www.kaggle.com/c/acquire-valued-shoppers-challenge/data

TABLE II
TARGET CLASSIFIER SETTING

Dataset	Model Architecture	Optimizer	Learning Rate	Train Epoch
CIFAR10	DenseNet121 [35]	Adam	0.1	300
Purchase100	Four FC layers	Adam	0.001	100
FaceScrub530	Same as [12]	Adam	0.0002	150
CIFAR100	DenseNet121 [35]	SGD	0.1	300
Texas	Five FC layers	Adam	0.001	20
Location	Three FC layers	Adam	0.01	30
UTKFace	Similar as [13] (max-pool → avg-pool)	Adam	0.001	30

TABLE III
TYPES AND NUMBERS OF MODELS USED IN DIFFERENT ATTACKS, AS WELL AS THE DATASETS FOR TRAINING AND TESTING MODELS

Attack	Shadow Model			Inference Model		
	Num.	D_{train}	D_{test}	Num.	D_{train}	D_{test}
NSH	0	N.A.	N.A.	1	D_1^1+$D_3^{1'}$	D_1^2+$D_3^{2'}$
Mlleaks	1	D_1^1+D_3^1	D_1^2+D_3^2	1	D_1^1+D_3^1	D_1^2+$D_3^{2'}$
Adaptive	1	D_1^1+D_3^1	D_1^2+D_3^2	1	D_1^1+D_3^1	D_1^2+$D_3^{2'}$
BlindMI	0	N.A.	N.A.	0	N.A.	D_1^2+$D_3^{2'}$
Gap	0	N.A.	N.A.	0	N.A.	D_1^2+$D_3^{2'}$
Transfer	1	D_1^1+D_3^1	D_1^2+D_3^2	0	N.A.	D_1^2+$D_3^{2'}$
Boundary	0	N.A.	N.A.	0	N.A.	D_1^2+$D_3^{2'}$
FP	16	D_1^1+D_3^1	D_1^2+D_3^2	0	N.A.	D_1^2+D_3^2
Enhanced	16	D_1^1+D_3^1	D_1^2+D_3^2	0	N.A.	D_1^2+D_3^2
Model Inversion	0	N.A.	N.A.	1	$D_{1,2,3}^{0.8}$	$D_{2,3}^{0.2}$
Attribute Inference	0	N.A.	N.A.	1	D_1^2	D_3^2

Note that some attacks do not use models but algorithms, and we also report the datasets used in inference.

to the number of samples of D_1^1, D_1^2, denoted as $D_3^{1'}$, $D_3^{2'}$. In addition, we randomly sample 80% from D_1, D_2, and D_3 to form $D_{1,2,3}^{0.8}$, and the remaining 20% of D_2 and D_3 to form $D_{2,3}^{0.2}$. For the target model, we use D_1 for training and D_3 for testing, which means D_1 samples are members and D_3 are non-members. In the membership inference attack and attribute inference attack, we assume that the attacker has access to half of the members (i.e., D_1^1). Therefore, for MIA, D_1^1 and $D_3^{1'}$ are used to train the attack model, and D_1^2 and $D_3^{2'}$ are used for testing. For the attribute inference attack, D_1^1 is used for training and D_3^1 is used for testing. In the model inversion attack, for the FaceScrub530 classifier, the attacker uses a CelebA [34] dataset to train the inversion model, following the same setting in [12]. For other classifiers, the attacker uses $D_{1,2,3}^{0.8}$ to train the inversion model and $D_{2,3}^{0.2}$ to test the inversion error. We use D_2 as the reference dataset for defenses that require training additional defense models, e.g., MemGaurd [17], Min-Max [4] and our approach.

2) Target Classifier Setting: We use the same model architectures as in previous work [4], [7], [12] to train the target classifiers. Table II presents the model architecture, optimizer, learning rate, and number of training epochs for the target classifier corresponding to 7 datasets. The training set is D_0 and the test set is D_2.

3) Attack Setting: We conduct 9 member inference attacks, 1 model inversion attack, and 1 attribute inference attack in our experiments. Table III summarizes the types and numbers of models used in different attacks, as well as the datasets for training and testing models. Note that some attacks do not use models but algorithms, and we also report the datasets used in inference.

NSH [4]: We use an inference classifier with the same model architecture as [4]. The classifier is trained with Adam optimizer for 50 epochs. For CIFAR10, FaceScrub530 and CIFAR100, the learning rate is 0.001. For Purchase100, Texas and Location, the learning rate is 0.01.

Mlleaks [5]: We train the shadow model with the same architecture and related settings as the target classifier. The membership classifier is a Multi-Layer Perception (MLP) with two Fully Connected (FC) layers. The optimizer is Adam. The learning rate for CIFAR10 is 0.01, while it is reduced to 0.0001 for Texas. For other datasets, the learning rate is 0.001. The training epochs for CIFAR10 and Purchase100 are set to 50. For other datasets, the number of training epochs is 100.

Adaptive [5]: This is an adaptive version of the Mlleaks attack, where the adversary is assumed to have access to all the information about the defender's PURIFIER and its reference data D_2. Hence, the input of the membership classifier is the purified confidence score vectors. The rest of the settings remain consistent with Mlleaks.

BlindMI [9]: We consider BlindMI-DIFF-w/, where the attacker is assumed to know the softmax output of the target classifier and the ground truth of the corresponding sample. We use the softmax output to generate non-member samples. The source code of [9] is adopted to implement the attack.

Gap [7]: No additional models are required.

Transfer [7]: The shadow models we use have the same architecture as the target model. The optimizer is Adam. The learning rate is 0.1 for CIFAR10 and CIFAR100, 0.0001 for Location and 0.0002 for FaceScrub530, and 0.001 for the other datasets. The number of training epochs for CIFAR10 and CIFAR100 is set to 300, while it is 150 for FaceScrub530. For other datasets, the number of training epochs is 100. Three thresholds are used in previous work, and we use the one that performs best.

Boundary [7]: We use HopSkipJump noise with a total of 300 evaluations per sample to ensure the attack performance is stable. We report the results on the L_2 norm. Due to the high query complexity, we report the results on a subset of 500 samples.

MIA from First Principles (FP) [10]: This attack provides two algorithms: online and offline, and we use the latter. We train 16 shadow models, each of which shares the same architecture as the target model. The learning rate is 0.001 for Purchase100 and Texas, 0.0002 for FaceScrub530, 0.01 for Location, and 0.1 for the other datasets. The number of training epochs for Purchase100 and CIFAR100 is set to 100, while it is 150 for FaceScrub530, 120 for CIFAR10, 20 for Texas, and 30 for Location.

Enhanced MIA (Enhanced) [11]: This paper proposes 4 attack methods, namely S, P, R, and D. We adopt method D and train 16 shadow models. The training settings are the same as FP.

Model inversion attack [12]: The source code of [12] is adopted to implement this attack.

Attribute inference attack [13]: The race classifier is an MLP with three FC layers. We use the Adam optimizer with a learning rate of 0.001. The number of training epochs is set to 50.

4) Defense Setting: We implement PURIFIER and reproduce 6 defense methods. Table IV summarizes the types and numbers of models used in different defenses and the datasets for training and testing models.

PURIFIER . For kNN in the *label swapper* H, we set k=1 for all datasets, d=1e-5 for CIFAR10 and Purchase100, 1e-4 for Texas and Location, 5*1e-5 for CIFAR100, 1e-7 for

TABLE IV
TYPES AND NUMBERS OF MODELS USED IN DIFFERENT DEFENSES AND THE DATASETS FOR TRAINING AND TESTING MODELS

Defense	Shadow Model			Defense Model		
	Num.	D_{train}	D_{test}	Num.	D_{train}	D_{test}
PURIFIER	0	N.A.	N.A.	1	D_2	N.A.
Min-Max	0	N.A.	N.A.	1	D_1	D_3
MemGaurd	1	D_1^1+D_3^1	D_1^2+D_3^2	0	N.A.	D_3
Model-Stacking	2	D_1	D_3	1	D_1	D_3
MMD Defense	0	N.A.	N.A.	1	D_1	D_3
SELENA	25	D_1	D_3	1	D_1	D_3
Relax-Loss	0	N.A.	N.A.	1	D_1	D_3

FaceScrub530, and 1e-11 for UTKFace. We use CVAE to implement the *confidence reformer* G. It has the layer size of [20, 32, 64, 128, 2, 128, 64, 32, 20] for CIFAR10, [200, 128, 256, 512, 20, 512, 256, 128, 100] for Purchase100, Texas and Location, [1060, 512, 1024, 2048, 100, 2048, 1024, 512, 1060] for FaceScrub530 and CIFAR100. The number of training epochs is 300 for all datasets. We use Adam optimizer with the l earning rate 0.001 for CIFAR10, 0.0001 for Purchase100, Texas and Location, and 0.0005 for Facescrub530 and CIFAR100.

Min-Max: As Section II-B mentioned, for the sake of the model utility and membership privacy, λ is introduced to balance two optimization targets. We set $\lambda = 0.5$ for all datasets. The number of epochs of the whole training phase for Location and Purchase100 is set to 100, while it is 150 for FaceScrub530 and CIFAR10, 300 for CIFAR10, and 50 for Texas. We use an inference model with the same model architecture as [4]. The attack model and target model are trained with Adam optimizer for 76 sub-epochs in each epoch. For FaceScrub530, the learning rate is 0.0002. For Texas, the learning rate is 0.0001. And others are 0.001.

MemGuard: We adopt the same attack binary classifier in the Mlleaks attack to distinguish the confidence vectors from the member and the non-member. Gradient descent with normalized gradient is conducted with hyper-parameter $c_1 = 1, c_2 = 10, c_3 = 0.1$ to generate perturbation vectors on member vectors.

Model-Stacking: We train two shadow models with the same architecture and related settings as the target classifier. Then, an MLP with three FC layers is trained to integrate the outputs of the shadow models to obtain the final prediction. The optimizer for the MLP is Adam, with a learning rate set at 0.01 and a total of 200 training epochs. D_0 is randomly divided into three equal parts and used to train three models respectively.

MMD Defense: This defense employs MMD as a regularizer in the loss function, and we set the weight of MMD in the loss function to 1e-5.

SELENA: We train 25 shadow models and a distillation model with the same architecture and related settings as the target classifier. Each sample is put into only 10 models as a component of the training dataset. In the training of shadow models, We apply SGD optimizer with a learning rate of 0.1. The same training strategy is adopted in the distillation process.

Relax-Loss: The pivotal parameter is the target loss threshold, denoted as α, which balances the loss levels between member and non-member data during the training phase. For CIFAR10, CIFAR100, Purchase100, and Texas, we maintain α at its original setting. However, for datasets not included in the original paper, we adjust α accordingly. For FaceScrub530, we set $\alpha = 1.0$. For Location and UTKFace, we set $\alpha = 0.8$.

5) Metric Setting: We employ the following metrics to evaluate the defense performance, inversion error, and efficiency of defense methods.

Classification Accuracy: This metric is assessed on both the training and test sets of the target classifier. It reflects the efficacy of the target classifier in the classification task.

Inference Accuracy: This represents the accuracy of the attacker's predictive model in determining the membership and the additional sensitive attribute of input samples. For MIA, since the number of members and non-members in the test set is the same, the higher proximity of the accuracy value to 50% signifies a diminished attack effect.

Area Under Curve (AUC): AUC is defined as the area under the Receiver Operating Characteristic (ROC) curve. This metric is employed to assess the member inference effect of Transfer and Boundary attacks. The closer the AUC value is to 0.5, the worse the inference effect.

Inversion Error: The inversion error is quantified by the mean squared error between the original input sample and the reconstruction.

Efficiency: Following [12], we measure the efficiency of a defense method by reporting its training time and testing time relative to the original time required by the target classifier.

B. Effectiveness of PURIFIER

In this section, we assess the defensive effectiveness of PURIFIER against a variety of data inference attacks and compare it with existing defense techniques.

1) Effectiveness Against Membership Inference Attack: Table V presents the defense performance of PURIFIER against nine different MIAs. For each dataset, PURIFIER can reduce the accuracy or AUC of membership inference attacks while maintaining the utility of the classifier. For example, on FaceScrub530, PURIFIER has almost no influence on the classifier test accuracy (the test accuracy only drops by 0.6%), but it decreases the attack accuracy of nine MIAs to about 50%, and the AUC is closer to 0.5, which means that PURIFIER has a significant defensive effect. It is noteworthy that the test accuracy of most datasets drops within 0.7%, only CIFAR100's test accuracy drops by 2.02%. This alteration in test accuracy can be attributed to the fact that the k NN of the label swapper may mistakenly classify non-member samples as members, thereby modifying the predicted label.

In addition, Table V also shows the experimental results of six other defense methods. It can be seen that in most cases, the defense effect of PURIFIER is the best, which shows that PURIFIER is better than other defense methods in concealing member privacy information. For example, on FaceScrub530, PURIFIER achieves the best defense effect on eight attacks. Relax-Loss has a better defense effect on BlindMI, but the corresponding attack accuracy is only 0.08% lower than PURIFIER. For Adaptive, Gap, and Transfer attacks, PURIFIER shows the best defense performance on all datasets.

TABLE V
DEFENSE PERFORMANCE OF PURIFIER AND OTHER DEFENSE METHODS AGAINST VARIOUS ATTACKS

Dataset	Defense	Utility		Membership Inference Attack Accuracy/AUC									Inversion Error
								Label only attacks					
		Train acc.	Test acc.	NSH	Mlleaks	Adaptive	BlindMI	Gap	Transfer	Boundary	FP	Enhanced	L2 norm
CIFAR10	None	99.99%	95.92%	56.03%	56.26%	N.A.	54.76%	52.04%	0.5186	0.5214	53.40%	53.91%	1.4357
	Purifier	95.93%	95.64%	51.49%	**50.00%**	**50.00%**	**50.05%**	**50.24%**	**0.4973**	0.4651	**50.54%**	**50.00%**	**1.4950**
	Min-Max	99.40%	94.38%	53.97%	52.93%	52.75%	53.52%	52.51%	0.5253	0.5011	53.03%	54.30 %	1.4770
	MemGuard	99.99%	95.92%	53.63%	52.24%	52.07%	52.03%	52.04%	0.5190	0.5029	53.12%	53.97%	1.4439
	Model-Stacking	95.80%	92.12%	51.93%	51.01%	50.99%	52.69%	51.84%	0.5205	**0.5008**	50.90%	50.95%	1.4723
	MMD Defense	99.99%	87.44%	59.50%	57.60%	57.32%	53.92%	56.28%	0.5365	0.5437	56.21%	59.13%	1.4414
	SELENA	98.40%	93.90%	52.14%	52.35%	52.21%	51.08%	52.25%	0.5097	0.4896	50.63%	51.50%	1.4350
	Relax-Loss	99.29%	87.42%	**50.07%**	55.27%	58.94%	54.77%	55.94%	0.5087	0.5605	50.88%	50.85%	1.4413
Purchase100	None	100.00%	84.36%	70.36%	64.43%	N.A.	69.82 %	57.82%	0.5566	N.A.	65.37%	62.72%	0.1426
	Purifier	84.55%	83.80%	**50.08%**	**50.00%**	**50.00%**	**49.81%**	**50.64%**	**0.5235**	N.A.	50.58%	50.10 %	**0.1526**
	Min-Max	99.89%	82.03%	65.13%	63.95%	64.06%	57.39%	58.93%	0.5760	N.A.	56.26%	54.82%	0.1428
	MemGuard	100.00%	84.36%	62.28%	57.86%	57.74%	61.35%	57.82%	0.5752	N.A.	65.23%	62.78%	0.1426
	Model-Stacking	81.84%	69.68%	61.16%	55.53%	55.28%	60.36%	56.08%	0.5806	N.A.	53.73%	52.90%	0.1472
	MMD Defense	100.00%	82.65%	69.48%	69.89%	69.13%	66.62%	58.67%	0.5718	N.A.	66.44%	64.35%	0.1439
	SELENA	83.24%	79.53%	51.90%	52.97%	52.84%	53.04%	51.83%	0.5602	N.A.	50.16%	50.50%	0.1440
	Relax-Loss	99.50%	82.17%	52.60%	61.31%	62.30%	57.84%	58.67%	0.5442	N.A.	**50.47%**	**50.08%**	0.1435
FaceScrub530	None	100.00%	77.68%	69.34%	75.04%	N.A.	50.40%	61.16%	0.5868	0.7739	71.68%	67.96%	0.0114
	Purifier	77.72%	77.06%	**50.89%**	**49.91%**	**50.17%**	50.08%	**50.41%**	**0.5385**	**0.4745**	**50.16%**	**50.00%**	**0.0454**
	Min-Max	98.99%	68.31%	65.56%	69.84%	69.13%	61.16%	65.34%	0.6428	0.6430	70.40%	69.36%	0.0182
	MemGuard	100.00%	77.68%	62.48%	60.06%	59.64%	62.42%	61.16%	0.6075	0.6418	71.69%	67.80%	0.0117
	Model-Stacking	86.30%	57.05%	62.00%	51.86%	51.74%	60.62%	64.63%	0.5994	0.6379	50.62%	52.26%	0.0417
	MMD Defense	100.00%	77.38%	64.88%	67.95%	67.38%	63.55%	61.31%	0.6034	0.6783	71.07%	67.00%	0.0111
	SELENA	81.06%	72.05%	51.68%	51.23%	51.89%	54.05%	50.50%	0.5733	0.5844	50.44%	52.01%	0.0131
	Relax-Loss	99.52%	75.24%	51.90%	70.64%	70.75%	**50.00%**	62.14%	0.6335	0.7353	53.51%	59.94%	0.0109
CIFAR100	None	99.98%	66.36%	76.98%	73.78%	N.A.	76.34%	76.86%	0.6495	0.6668	64.81%	72.92%	0.9073
	Purifier	66.37%	64.34%	55.03%	**50.02%**	**50.00%**	**50.02%**	**50.70%**	**0.5851**	**0.5042**	**50.36%**	**50.53%**	**0.9378**
	Min-Max	98.27%	68.36%	60.19%	61.98%	61.66%	60.15%	64.96%	0.6713	0.6329	58.10%	61.00%	0.9106
	MemGuard	100.00%	68.91%	58.17%	57.45%	56.96%	59.76%	65.53%	0.6624	0.5877	61.46%	68.07%	0.9123
	Model-Stacking	65.53%	60.04%	57.30%	59.11%	58.75%	58.82%	52.75%	0.6672	0.5623	50.28%	50.42%	0.9203
	MMD Defense	100.00%	67.50%	57.89%	57.22%	56.97%	65.31%	66.25%	0.7487	0.6532	73.67%	83.19%	0.9231
	SELENA	78.00%	62.10%	**50.32%**	50.42%	50.33%	57.59%	57.95%	0.6227	0.4982	50.71%	50.64%	0.9184
	Relax-Loss	99.37%	46.64%	53.14%	57.43%	60.81%	56.24%	76.37%	0.6522	0.7795	50.40%	50.49%	0.9266
Texas	None	79.17%	47.91%	66.37%	58.93%	N.A.	53.88%	69.02%	0.6334	N.A.	59.53%	63.18%	N.A.
	Purifier	47.98%	47.59%	**50.00%**	45.89%	**50.00%**	**49.91%**	**50.25%**	**0.5996**	N.A.	**50.03%**	**50.00%**	N.A.
	Min-Max	75.51%	49.05%	54.12%	57.03%	56.89%	53.77%	62.23%	0.6326	N.A.	52.56%	53.12%	N.A.
	MemGuard	75.04%	49.88%	55.80%	54.99%	56.75%	53.09%	62.58%	0.6557	N.A.	56.81%	59.41%	N.A.
	Model-Stacking	53.18%	47.02%	51.20%	**50.08%**	56.93%	52.86%	53.08%	0.6424	N.A.	52.52%	53.08%	N.A.
	MMD Defense	77.32%	50.01%	56.96%	58.39%	59.73%	67.40%	63.66%	0.6570	N.A.	63.48%	70.74%	N.A.
	SELENA	77.90%	55.25%	54.40%	51.00%	57.79%	58.04%	61.33%	0.6261	N.A.	50.96%	52.17%	N.A.
	Relax-Loss	85.50%	48.19%	53.05%	58.48%	59.19%	47.35%	68.66%	0.6201	N.A.	50.26%	50.01%	N.A.
Location	None	100.00%	59.44%	82.37%	84.00%	N.A.	76.13%	71.13%	0.6513	N.A.	79.72%	77.12%	N.A.
	Purifier	59.50%	58.88%	53.69%	**50.31%**	**51.12%**	**50.75%**	**54.75%**	**0.6269**	N.A.	50.41%	51.19%	N.A.
	Min-Max	99.85%	56.49%	59.52%	58.38%	59.71%	65.08%	71.68%	0.6772	N.A.	58.00%	59.91%	N.A.
	MemGuard	100.00%	58.44%	61.21%	58.28%	59.13%	65.22%	70.78%	0.6960	N.A.	69.31%	67.59%	N.A.
	Model-Stacking	70.29%	57.82%	**53.24%**	51.88%	52.66%	67.41%	56.24%	0.6805	N.A.	53.66%	55.59%	N.A.
	MMD Defense	100.00%	59.96%	63.25%	54.77%	56.32%	71.29%	70.02%	0.6654	N.A.	81.03%	78.34%	N.A.
	SELENA	77.90%	55.25%	54.40%	51.00%	52.23%	65.86%	61.33%	0.6509	N.A.	50.75%	55.37%	N.A.
	Relax-Loss	99.50%	58.94%	58.56%	69.94%	70.00%	77.94%	70.28%	0.6480	N.A.	**50.31%**	**50.56%**	N.A.

Results of the Transfer and boundary attack are reported in AUC [7]. Note that N.A. means that the setting is not applicable.

PURIFIER also outperforms other defenses in terms of the defensive-practical trade-off. Model-Stacking and Relax-Loss demonstrate superior defense effects compared to PURIFIER in certain instances. For instance, on CIFAR10, NSH's accuracy when targeting Relax-Loss is 1.42% lower than when attacking PURIFIER, yet the corresponding test accuracy decreases by 8.22%. On Location, FP attacks Model-Stacking with a 0.45% lower accuracy than PURIFIER, while the test accuracy drops by 1.06%. The key to PURIFIER's ability to preserve utility lies in its emphasis on the private information of member samples, thereby bringing members closer to non-members while minimally affecting non-member samples. In contrast, other defense methods influence both members and non-members. For instance, MMD employs a set regularizer to reduce the classification accuracy disparity between members and non-members.

2) Effectiveness Against Model Inversion Attack: We further investigate the defense performance of PURIFIER against the model inversion attack. Table V presents the inversion errors of the model inversion attack based on different defense methods on CIFAR10, Purchase100, FaceScrub530, and CIFAR100. We can see that the inversion errors of PURIFIER on the four datasets are the highest, indicating that PURIFIER can effectively defend against the model inversion attack. For instance, the inversion loss on FaceScrub530 is approximately 4 times that of no defense (i.e., from 0.0114 to 0.0454). Note that the impact of the model inversion attack on CIFAR10, Purchase100 and CIFAR100 is less than that on FaceScrub530. This is because the inversion attack does not work well even without any defense on these classifiers.

We believe that the efficacy of PURIFIER in defending against model inversion can be attributed to the information loss caused by alterations in the distribution of the confidence vector. The greater the impact of the defense method on the distribution of the confidence vector, the better the defensive effect against model inversion. Beyond PURIFIER, Model-Stacking integrates multiple models to generate the final prediction, resulting in a significant change in the confidence vector compared to a single model's prediction. Consequently, its defensive efficacy (the inversion loss is 0.0417) ranks second only to PURIFIER (the inversion loss is 0.0454). However, Min-Max and Relax-Loss focus on the loss function, MemGuard introduces noise into the confidence vector, MMD concentrates on classification accuracy, and SELENA yields a model identical to the target model structure. Therefore, these methods exert limited influence on the confidence vector of an individual sample (the inversion loss only increases by 0.0017 at most).

Furthermore, Fig. 3 illustrates the image reconstruction results of the model inversion attack based on different defense

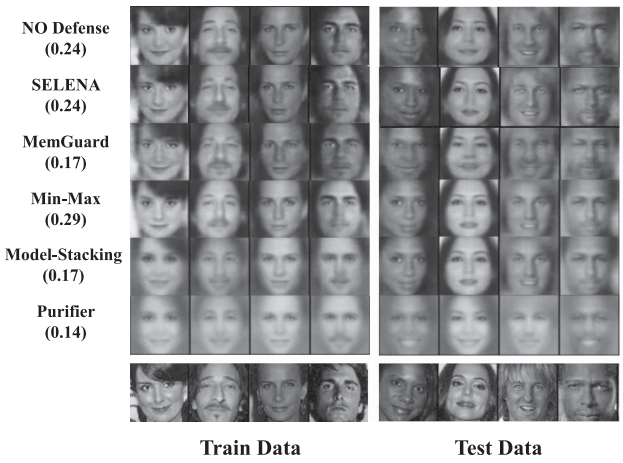


Fig. 3. Model inversion attack against the FaceScrub530 classifier defended by different approaches.

TABLE VI
ATTRIBUTE INFERENCE ATTACK AGAINST THE UTKFACE CLASSIFIER WITH DIFFERENT DEFENSE METHODS

Dataset	Defense	Attack Accuracy					
		Race0	Race1	Race2	Race3	Race4	Total
UTKFace	None	82.08%	54.60%	26.37%	20.36%	0.00%	53.40%
	Purifier (VAE)	**100.00%**	**0.00%**	**0.00%**	**0.00%**	**0.00%**	**42.40%**
	Min-Max	93.19%	10.64%	6.23%	9.68%	0.00%	45.64%
	MemGaurd	N.A.	N.A.	N.A.	N.A.	N.A.	N.A.
	Model-Stacking	90.47%	14.15%	0.00%	13.20%	0.00%	43.80%
	MMD Defense	**100.00%**	**0.00%**	**0.00%**	**0.00%**	**0.00%**	**42.40%**
	SELENA	83.04%	53.44%	15.64%	12.12%	0.00%	51.24%
	Relax-Loss	74.77%	75.14%	21.33%	26.98%	1.81%	55.16%

TABLE VII
GAP OF THE CLASSIFIER'S CONFIDENCE IN PREDICTING THE CORRECT CLASS (I.E., CONFI) AND THE PREDICTION UNCERTAINTY (I.E., UNCER) BETWEEN MEMBERS AND NON-MEMBERS

Metric	Defense	CIFAR10		Purchase100		FaceScrub530	
		Max	Avg.	Max	Avg.	Max	Avg.
Confi	None	0.0793	0.002	0.0954	0.0082	0.6093	0.1213
	Purifier	**0.0057**	**0.0015**	**0.0452**	**0.0011**	**0.0023**	**0.0004**
Uncer	None	0.0403	0.0013	0.0673	0.0076	0.5709	0.0766
	Purifier	**0.0037**	**0.0007**	**0.0112**	**0.0005**	**0.0029**	**0.0004**

methods on FaceScrub530. We measure the inversion quality by reporting the average facial similarity score to the ground truth using Microsoft Azure Face Recognition Service [36], which is shown on the left side of Fig. 3. The smaller the number, the lower the similarity between the reconstructed sample and the original sample. It can be observed that the attacker can reconstruct almost the same image without any defense measures. Under the defense of PURIFIER, the similarity score between the reconstructed sample and the original sample is the smallest (0.1 lower than No Defense), and the corresponding reconstructed image loses a lot of details compared to the original image. The reconstructed images of other defense methods are much clearer than PURIFIER, especially Min-Max and SELENA, which indicates that PURIFIER is more effective in defending against the model inversion attack than other methods.

3) Effectiveness Against Attribute Inference Attack: In this section, we explore the defense effect of PURIFIER against attribute inference attacks. We cannot achieve a successful attribute inference attack by strictly replicating the model architecture and related settings of the UTKFace classifier mentioned in [13]. We argue that it is challenging to infer 5 race classes based solely on the sensitive information in the confidence vector of length 2. Thus, we conduct an enhanced attribute inference attack on the output vector with a length of 64, which is the last hidden layer of the target model. Note that since PURIFIER only modifies the model output vector, to make it applicable to the hidden layer vector, we remove the part of PURIFIER that needs to use the label, i.e., only use VAE to reconstruct the hidden layer vector. Min Max, Model Stacking, MMD, SELENA, and Relax Loss participate in the target model training process so that the hidden layer vector can be directly extracted. MemGaurd generates adversarial samples for the confidence vector and the label is required, so it is not available in this experiment.

Table VI shows the experimental results under undefended and various defense strategies. The race classes, from 0 to 4, correspond to white, black, Asian, Indian, and others (such as Hispanic, Latin American, and Middle Eastern), respectively. It can be seen that except for Relax Loss, the accuracy of attribute inference attack diminishes by 4.16% $\sim$ 13% under the other defenses. Notably, when employing PURIFIER and MMD for protection, the attack model consistently predicts Race0 for all samples, indicating a failure of the attack. PURIFIER's VAE reconstructs the hidden layer vector and directly alters the hidden layer information. MMD modifies the distribution of softmax output through regularization, indirectly impacting the hidden layer parameters via gradient backpropagation. The hidden layer parameters of other methods are also different from those in the case of no defense. Consequently, these defense techniques exhibit varying degrees of effectiveness. Relax Loss aims to reduce the differentiation between the loss distributions of training and testing, and alterations in the loss distribution have little effect on the sensitive attribute information contained in the hidden layer.

C. Indistinguishabilities in Purified Scores

In this subsection, we design more specific experiments to analyze the influence of purified confidence scores on MIAs by evaluating three indistinguishabilities: individual shape, statistical distribution, and prediction label.

1) Individual Indistinguishability: To verify the indistinguishability of confidence scores between members and non-members, we plot the confidence of the target classifier predicting the correct class and the prediction uncertainty in Fig. 4, which presents the results on three datasets: CIFAR10, Purchase100, and FaceScrub530. We can observe that PURIFIER can reduce the gap between two curves corresponding to members and non-members respectively.

We also show the maximum gap and the average gap between curves in Table VII. The results demonstrate that our method can substantially decrease both the maximum and average confidence gaps between the target classifier's confidence in predicting the correct class as well as the prediction uncertainty on its members versus non-members. For instance, on FaceScrub530, the maximum and average confidence gaps of the predicted correct category are decreased by 0.607 and 0.1209, respectively. On Purchase100, the maximum and average confidence gaps of

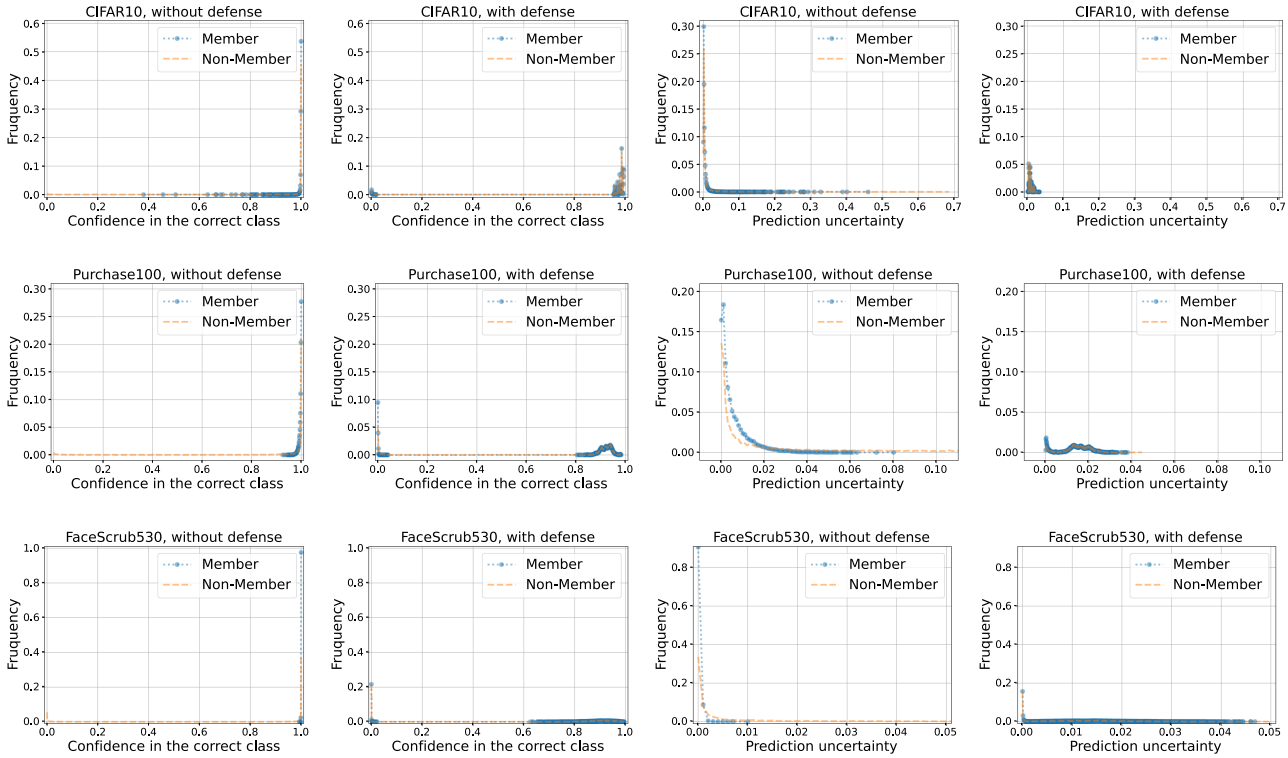
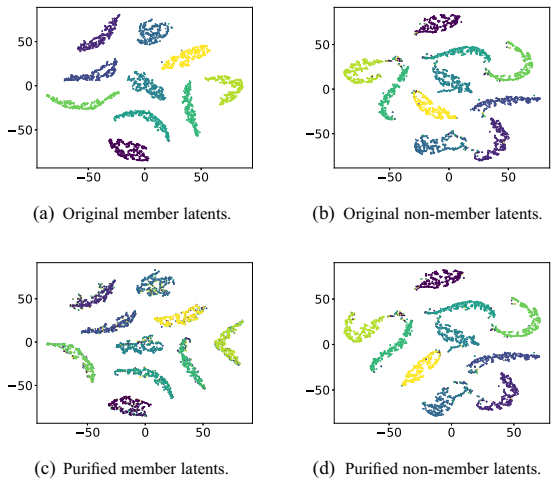


Fig. 4. Distribution of the target classifier's confidence in predicting the correct class and the prediction uncertainty on members and non-members of the training set.

the predicted uncertainty are decreased by 0.0561 and 0.0071, respectively. This demonstrates that PURIFIER effectively reduces the individual differences between members and non-members.

2) Statistical Indistinguishability: We use t-SNE [37] to visualize the statistical distribution of confidence score vectors in the encoder latent space of the *confidence reformer* on a two-dimensional plane. Fig. 5 shows the distributional differences between members and non-members in the latent space on CIFAR10. As illustrated in Fig. 5(a) and (b), the latent vectors of members are more likely to be clustered according to their labels, while those of non-members are more scattered. Fig. 5(c) and (d) also depict the statistical distribution of members and non-members defended by PURIFIER in the latent space. After being processed by PURIFIER, Gaussian noises are injected to make the clustered member latent vectors more scattered on the latent space and shifted to a certain extent towards the distribution of non-members of the same class. This demonstrates that PURIFIER can reduce the statistical divergence between members and non-members while maintaining semantic utility.

3) LABEL INDISTINGUISHABILITY: PURIFIER employs *label swapper* to detect and replace the prediction label of members. The use of *label swapper* results in a negligible decrease in test accuracy, whereas replacing the labels of member samples leads to a more significant reduction in training accuracy. This is demonstrated in Table V, where the model's training accuracy closely mirrors its test accuracy. Except for CIFAR100, which exhibits a difference of 2%, the disparities for other datasets are approximately 1% or even less. Such label indistinguishability significantly diminishes the efficacy of the *label-only attack*.



(a) Original member latents. (b) Original non-member latents.

(c) Purified member latents. (d) Purified non-member latents.

Fig. 5. The statistical distribution of latent vectors on the CIFAR10 dataset. Different colors stand for latent vectors with different labels. (a) and (b) depict latent vectors of the original member and non-member confidence score vectors; (c) and (d) depicts latent vectors of member and non-member confidence score vectors with PURIFIER defended.

We perform the ablation study on the label swapper to verify its effectiveness in our defense. As shown in Table VIII, we defend the classifiers against label-only transfer attack under three settings: no defense, defense with only confidence reformer G, defense with label swapper H and confidence reformer G . The results show that the AUC drops significantly with the help

TABLE VIII
ABLATION STUDY ON THE LABEL SWAPPER

Dataset	Defense	Label Only Transfer Attack
CIFAR10	None	0.5186
	G	0.5069
	$H + G$	**0.4973**
CIFAR100	None	0.6495
	G	0.6479
	$H + G$	**0.5851**

G represents the confidence reformer, and H represents the label swapper.

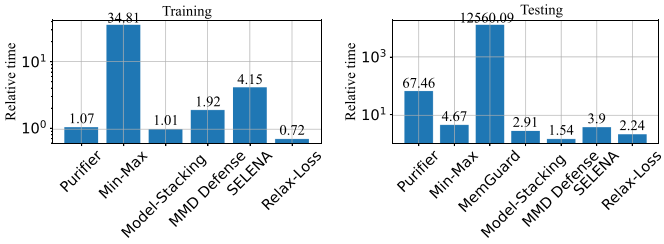


Fig. 6. Efficiency of different defense methods.

of label swapper compared to those mechanisms without it, i.e., the AUC of CIFAR10 and CIFAR100 decreases by 0.0213 and 0.0644 respectively. Besides, the result shows that the label-only transfer attack can achieve similar AUC among models without defense and PURIFIER without the label swapper (the AUC of CIFAR10 and CIFAR100 decreases by 0.0117 and 0.0016 respectively), which means that PURIFIER without label swapper fails to alleviate the label-only attack. In conclusion, the label swapper is indispensable for defending against label-only attacks.

D. Efficiency of PURIFIER

Fig. 6 shows the efficiency of PURIFIER in comparison with other defenses on the Purchase100 training and test sets. Note that since MemGuard focuses on post-processing each prediction vector of the undefended model, we omit the training time of MemGuard in Fig. 6 and only compare its test time. The training time (about 7 minutes) of PURIFIER is 1.07 times that of the target classifier, which is better than most defenses such as Min-Max (about 4 hours) and SELENA (about 29 minutes). The testing time (52.86 seconds) of PURIFIER is 67.46 times that of the target classifier, which is approximately 14.4 to 43.8 times that of other defenses except MemGuard. This is because kNN computes the distance between each input sample and all the members. In contrast, the time spent on other processes of PURIFIER is insignificant. Nevertheless, it can be seen that the test time of MemGuard is 186.2 times that of PURIFIER. This is because MemGuard has to solve a complex optimization problem to obfuscate the prediction vector for each query, while PURIFIER only requires a simple distance calculation. In conclusion, we argue that the computation time of PURIFIER is acceptable.

E. Further Experiments

1) Effect of Component Order: The relative order of the *confidence reformer* and the *label swapper* is also worth discussing.

TABLE IX
DEFENSE PERFORMANCE OF DIFFERENT PLACEMENT ORDERS OF CONFIDENCE REFORMER AND LABEL SWAPPER

Dataset	Order	Train acc.	Test acc.	NSH	Mlleaks	BlindMI	Gap
CIFAR10	P1	97.60%	95.52%	51.65%	50.26%	50.64%	50.84%
	P2	95.93%	95.64%	**51.49%**	**50.00%**	**50.05%**	**50.24%**
Purchase100	P1	86.59%	82.23%	51.71%	50.09%	50.96%	51.68%
	P2	84.55%	83.80%	**50.08%**	**50.00%**	**49.81%**	**50.64%**
Facescrub530	P1	77.58%	77.52%	51.56%	51.04%	**50.00%**	**50.02%**
	P2	77.72%	77.06%	**50.89%**	**49.91%**	50.08%	50.41%

P1 refers to placing Confidence Reformer before Label Swapper, and P2 reversely.

As we mentioned in Section IV, the confidence reformer is to achieve individual and statistical indistinguishability, while the label swapper is to achieve label indistinguishability. The independence of their functions also indicates that we can put the label swapper before the confidence reformer or vice versa. To investigate the effect of the order of the label swapper and the confidence reformer on PURIFIER, we evaluate the utility and defense performance against four attacks (NSH, Mlleaks, BlindMI, Gap) of PURIFIER in two orders (i.e., placing the label swapper after the confidence reformer and placing the label swapper before the confidence reformer) on the CIFAR10, Purchase100, and Facescrub530 datasets.

As shown in Table IX, placing the label swapper before the confidence reformer in most cases can achieve better classifier utility and PURIFIER defense performance. For instance, the test accuracy of the Purchase100 classifier is 1.57% higher than P1 under the PURIFIER defense of P2, while the corresponding attack accuracy drops by about 1% on average. A reasonable explanation is that when the label swapper precedes the confidence reformer, the intention to swap the output label to the one with another confidence score becomes more diluted. Because the confidence vector, post-swap, will be reconstructed by the confidence reformer. In other words, positioning the confidence reformer after the label swapper introduces additional randomness into the output result, thereby making the label swap less detectable. Conversely, label swapping following the transformation of the confidence vector by the confidence reformer may destroy the conversion effect and reintroduce disparities between members and non-members.

2) Influence of Different Reference Datasets: We also investigate the impact of the reference dataset by using different sizes or distributions of data to train PURIFIER. Specifically, for in-distribution data, we vary the size of D_2 and also replace D_2 with D_1. For out-of-distribution data, we use Purchase100 data to train the PURIFIER for the CIFAR100 classifier and use Texas data to train the PURIFIER for the Purchase100 classifier.

We demonstrate the defense performance of different in-distribution training data of Purchase100 against three MIAs (i.e., NSH, Mlleaks, and Adaptive) in Table X. We can see that PURIFIER is still effective. The membership inference accuracy drops to nearly 50% regardless of the size of D_2. PURIFIER is robust to the size of the D_2. The difference in the defense performance is negligible as the size of D_2 changes from 5,000 to 60,000. This is beneficial for the defender since one can achieve good performance with a small reference set. When we use the classifier's training data D_1 to train the PURIFIER, the defense performance is comparable to those on D_2. For

TABLE X
EFFECT OF THE PURIFIER'S IN-DISTRIBUTION TRAINING DATA ON THE DEFENSE PERFORMANCE

Training set	NSH	Mlleaks	Adaptive
D_2 (5,000)	50.16%	50.00%	50.68%
D_2 (10,000)	50.02%	50.00%	50.00%
D_2 (20,000)	50.08%	50.00%	50.00%
D_2 (40,000)	50.14%	50.00%	50.00%
D_2 (60,000)	50.02%	49.98%	50.00%
D_1 (20,000)	50.72%	49.93%	50.00%

The dataset is Purchase100.

TABLE XI
EFFECT OF THE PURIFIER'S OUT-OF-DISTRIBUTION TRAINING DATA ON THE DEFENSE PERFORMANCE

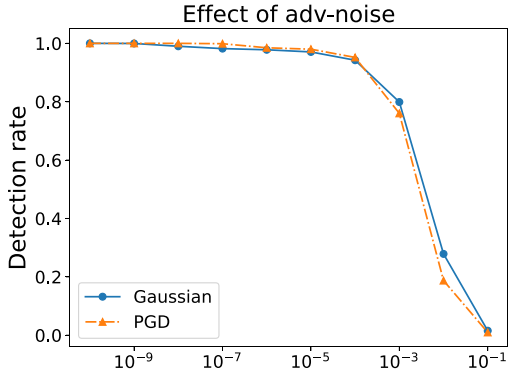
Classifier	Purifier	NSH	Mlleaks	Adaptive
CIFAR100	Purchase100	65.61%	50.02%	50.00%
Purchase100	Texas	50.00%	50.00%	50.00%

instance, the attack accuracy of the NSH attack is 50.72%, which is slightly higher than the results on D_2 , but acceptable.

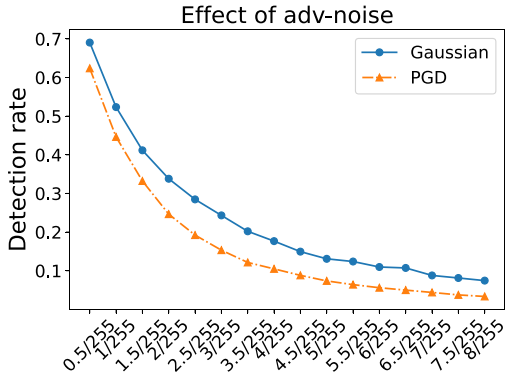
Table XI shows the effect of the out-of-distribution training data. We can observe that for Purchase100 and Texas, although the classifiers are trained on other datasets, the defense of PURIFIER is still effective (mostly 50%). An exception is that the accuracy of NSH attacking Purchase100 is 65.61%, However, compared to 70.36% without defense, PURIFIER has a certain defensive effect.

3) Effect of PURIFIER on Noisy Member Detection: Furthermore, we explore the ability of PURIFIER to detect noisy members. Since the label swapper needs to determine whether an input sample is a member, it should be robust to noise. We perform this experiment to verify whether PURIFIER can handle noisy samples. More specifically, PURIFIER can process a noisy member as a member. We generate noise members by adding Gaussian perturbation and adversarial perturbation (PGD [38]) of different magnitudes to the original member samples. We use the detection rate as a metric, which is defined as the ratio of noise samples that are detected as members among all noise samples.

Fig. 7 shows the results of the detection rate varying with perturbation magnitude on the CIFAR10 dataset. From Fig. 7(a), we can see that when the perturbation $\epsilon \leq$ 1e-4, PURIFIER can maintain a detection rate of more than 95%. When ϵ continues to increase, the detection rate begins to drop significantly. In practice, a perturbation that is too small may not change the image pixel value. We choose $\frac{0.5}{255}$ as the minimum perturbation unit. This is because any alteration in pixel value exceeding 0.5 will result in a change in its integer part. If an image pixel value is stored using rounding up, we apply a positive perturbation. Conversely, if it is stored by rounding down, a negative perturbation is utilized. Therefore, we further intercept curves in the range of $\frac{0.5}{255}$ to $\frac{8}{255}$, which is shown in Fig. 7(b). It can be seen that when $\epsilon = \frac{0.5}{255}$, PURIFIER can still maintain a detection rate of 69.04% (Gaussian perturbation) and 62.38% (PGD perturbation), which shows that PURIFIER has a certain degree of robustness to noise members.



(a) The perturbation ranges from 1e-9 to 1e-1.



(b) The perturbation ranges from $\frac{0.5}{255}$ to $\frac{8}{255}$.

Fig. 7. The detection rate of PURIFIER for noisy members with added Gaussian perturbation and adversarial perturbation on the CIFAR10 dataset. The x-axis represents the perturbation magnitude.

Furthermore, it is noteworthy that the detection rate curves for Gaussian perturbation and PGD perturbation are closely aligned. This indicates that the effectiveness of PGD perturbation is not significantly superior to that of Gaussian perturbation. This is because the adversarial attack optimizes the perturbation distribution within a constrained perturbation range to generate adversarial samples. Since PURIFIER directly classifies noise samples as member samples based on Euclidean distance (i.e., L_2 norm) within the range of ϵ through kNN, PURIFIER can work as long as the perturbation constraint of the adversarial attack also adheres to L_2 norm. One potential adaptive attack involves altering the perturbation constraint, such as employing the L_∞ norm to add perturbation when it is known that kNN uses L_2 norm. However, kNN can also adaptively modify the norm it employs.

VI. RELATED WORKS

Inference Attacks: The inference attacks against machine learning can be divided into model inference and data inference attacks. In model inference attacks [20], [21], [22], [23], an attacker could infer the parameters [22], hyper-parameters [23], architecture [20] and functionality [21] of a target model. We focus on data inference attacks in this paper. Xiao et al. [39] study the adversarial reconstruction problem where they aim to

prevent the latent representations from being decoded into the original input data. To this end, they regularize the encoder with an adversarial loss from a decoder. They study the face attribute prediction model which outputs 40 binary facial attributes. Our paper, on the contrary, studies black-box classifiers whose output is constrained by a probability distribution (i.e., values sum up to 1). Moreover, they do not consider the adversarial scenario where the attacker has no access to the same data distribution as the original training data. Jia and Gong [40] propose the adversarial formulation for privacy protection. They aim at protecting the privacy of users' sensitive attributes from being inferred from their public data. Our work investigates inference attacks that leverage prediction results of machine learning models to infer useful information about the input data.

General Membership Inference Attack: MIA is performed to determine whether a given data sample is part of a target dataset. Homer et al. [41] propose one of the first MIAs in the biomedical setting on genomic data. Some studies also perform MIAs on other biomedical data such as MicroRNA [42] and DNA methylation [43]. Pyrgelis et al. [44], [45] further show that it is possible to perform MIA on location datasets as well. Shokri et al. [1] perform MIAs in the machine learning setting which is the same as our work.

Secure & Privacy-Preserving Machine Learning: Many studies make use of trusted hardware and cryptographic computing to provide secure and privacy-preserving training and use of machine learning models. These techniques include homomorphic encryption, garbled circuits and secure multiparty computation on private data [46], [47], [48], [49], [50], [51] and secure computing using trusted hardware [52], [53]. While these methods safeguard sensitive data from direct observation by the attacker, they fail to prevent information leakage that may occur during model computation. This leakage can potentially be exploited by a variety of inference attacks.

VII. Conclusion

In this paper, we propose PURIFIER to defend data inference attacks. PURIFIER learns the pattern of non-member confidence score vectors and purifies confidence score vectors to this pattern without getting involved with the training process of the target model. It makes confidence vectors on members indistinguishable from those on non-members in terms of individual shape, statistical distribution, and prediction label. Our extensive experiments show that PURIFIER is effective in mitigating existing data inference attacks, outperforming previous defense methods, while imposing negligible utility loss.

References

[1] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," in *Proc. 2017 IEEE Symp. Secur. Privacy*, San Jose, CA, USA, 2017, pp. 3–18.

[2] M. Fredrikson, S. Jha, and T. Ristenpart, "Model inversion attacks that exploit confidence information and basic countermeasures," in *Proc. 22nd ACM SIGSAC Conf. Comput. Commun. Secur.*, Denver, CO, USA, 2015, pp. 1322–1333.

[3] B. Hitaj, G. Ateniese, and F. Perez-Cruz, "Deep models under the GAN: Information leakage from collaborative deep learning," in *Proc. 2017 ACM SIGSAC Conf. Comput. Commun. Secur.*, 2017, pp. 603–618.

[4] M. Nasr, R. Shokri, and A. Houmansadr, "Machine learning with membership privacy using adversarial regularization," in *Proc. 2018 ACM SIGSAC Conf. Comput. Commun. Secur.*, Toronto, Canada, 2018, pp. 634–646.

[5] A. Salem, Y. Zhang, M. Humbert, M. Fritz, and M. Backes, "ML-leaks: Model and data independent membership inference attacks and defenses on machine learning models," in *Proc. 26th Annu. Netw. Distrib. System Secur. Symp.*, 2019.

[6] S. Yeom, I. Giacomelli, M. Fredrikson, and S. Jha, "Privacy risk in machine learning: Analyzing the connection to overfitting," in *Proc. IEEE 31st Comput. Secur. Foundations Symp.*, 2018, pp. 268–282.

[7] Z. Li and Y. Zhang, "Membership leakage in label-only exposures," in *Proc. 2021 ACM SIGSAC Conf. Comput. Commun. Secur.*, 2021, pp. 880–895.

[8] J. Li, N. Li, and B. Ribeiro, "Membership inference attacks and defenses in classification models," in *Proc. 11th ACM Conf. Data Appl. Secur. Privacy*, 2021, pp. 5–16, doi: 10.1145/3422337.3447836.

[9] B. Hui, Y. Yang, H. Yuan, P. Burlina, N. Z. Gong, and Y. Cao, "Practical blind membership inference attack via differential comparisons," 2021, *arXiv:2101.01341*.

[10] N. Carlini, S. Chien, M. Nasr, S. Song, A. Terzis, and F. Tramer, "Membership inference attacks from first principles," in *Proc. 2022 IEEE Symp. Secur. Privacy*, 2022, pp. 1519–1519.

[11] J. Ye, A. Maddi, S. K. Murakonda, V. Bindschaedler, and R. Shokri, "Enhanced membership inference attacks against machine learning models," in *Proc. 2022 ACM SIGSAC Conf. Comput. Commun. Secur.*, 2022, pp. 3093–3106.

[12] Z. Yang, J. Zhang, E.-C. Chang, and Z. Liang, "Neural network inversion in adversarial setting via background knowledge alignment," in *Proc. 2019 ACM SIGSAC Conf. Comput. Commun. Secur.*, New York, NY, USA, 2019, pp. 225–240.

[13] C. Song and V. Shmatikov, "Overlearning reveals sensitive attributes," in*Proc. 8th Int. Conf. Learn. Representations*, 2020.

[14] M. Abadi et al., "Deep learning with differential privacy," in *Proc. 2016 ACM SIGSAC Conf. Comput. Commun. Secur.*, Vienna, Austria, 2016, pp. 308–318.

[15] R. Iyengar, J. P. Near, D. Song, O. Thakkar, A. Thakurta, and L. Wang, "Towards practical differentially private convex optimization," in *Proc. 2019 IEEE Symp. Secur. Privacy*, pp. 299–316.

[16] R. Shokri and V. Shmatikov, "Privacy-preserving deep learning," in *Proc. 22nd ACM SIGSAC Conf. Comput. Commun. Secur.*, 2015, pp. 1310–1321.

[17] J. Jia, A. Salem, M. Backes, Y. Zhang, and N. Z. Gong, "MemGuard: Defending against black-box membership inference attacks via adversarial examples," in *Proc. 2019 ACM SIGSAC Conf. Comput. Commun. Secur.*, 2019, pp. 259–274.

[18] D. Chen, N. Yu, and M. Fritz, "Relaxloss: Defending membership inference attacks without losing utility," in *Proc. Int. Conf. Learn. Representations*, 2022.

[19] M. Fredrikson, E. Lantz, S. Jha, S. Lin, D. Page, and T. Ristenpart, "Privacy in pharmacogenetics: An end-to-end case study of personalized warfarin dosing," in *Proc. 23rd USENIX Conf. Secur. Symp.*, 2014, pp. 17–32.

[20] S. J. Oh, M. Augustin, M. Fritz, and B. Schiele, "Towards reverse-engineering black-box neural networks," *Explainable AI: Interpreting, Explaining Visualizing Deep Learn.*, pp. 121–144, 2019.

[21] T. Orekondy, B. Schiele, and M. Fritz, "Knockoff nets: Stealing functionality of black-box models," in *Proc. 2019 IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 4949–4958.

[22] F. Tramèr, F. Zhang, A. Juels, M. K. Reiter, and T. Ristenpart, "Stealing machine learning models via prediction APIs," in *Proc. 25th USENIEX Secur. Symp.*, Austin, TX, USA, 2016, pp. 601–618.

[23] B. Wang and N. Z. Gong, "Stealing hyperparameters in machine learning," in *Proc. 2018 IEEE Symp. Secur. Privacy*, 2018, pp. 36–52.

[24] G. Ateniese, L. V. Mancini, A. Spognardi, A. Villani, D. Vitali, and G. Felici, "Hacking smart machines with smarter ones: How to extract meaningful data from machine learning classifiers," *Int. J. Secur. Netw.*, vol. 10, no. 3, pp. 137–150.

[25] L. Wei, B. Luo, Y. Li, Y. Liu, and Q. Xu, "I know what you see: Power side-channel attack on convolutional neural network accelerators," in *Proc. 34th Annu. Comput. Secur. Appl. Conf.*, San Juan, PR, USA, 2018, pp. 393–406.

[26] X. Wu, M. Fredrikson, S. Jha, and J. F. Naughton, "A methodology for formalizing model-inversion attacks," in *Proc. IEEE 29th Comput. Secur. Foundations Symp.*, 2016, pp. 355–370.

[27] S. Hidano, T. Murakami, S. Katsumata, S. Kiyomoto, and G. Hanaoka, "Model inversion attacks for prediction systems: Without knowledge of non-sensitive attributes," in *Proc. 15th Annu. Conf. Privacy Secur. Trust*, 2017, pp. 115–11509.

[28] Y. Zhang, R. Jia, H. Pei, W. Wang, B. Li, and D. Song, "The secret revealer: Generative model-inversion attacks against deep neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 250–258.
[29] K. M. Borgwardt, A. Gretton, M. J. Rasch, H.-P. Kriegel, B. Schölkopf, and A. J. Smola, "Integrating structured biological data by kernel maximum mean discrepancy," *Bioinformatics*, vol. 22, no. 14, pp. e49–e57, 2006.
[30] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola, "A kernel two-sample test," *J. Mach. Learn. Res.*, vol. 13, no. 1, pp. 723–773, 2012.
[31] X. Tang et al., "Mitigating membership inference attacks by self-distillation through a novel ensemble architecture," in *Proc. 31st USENIX Secur. Symp.*, Boston, MA, USA: USENIX Association, 2022, pp. 1433–1450. [Online]. Available: https://www.usenix.org/conference/usenixsecurity22/presentation/tang
[32] C. A. Choquette-Choo, F. Tramer, N. Carlini, and N. Papernot, "Label-only membership inference attacks," in *Proc. 38th Int. Conf. Mach. Learn.*, 2021, pp. 1964–1974. [Online]. Available: https://proceedings.mlr.press/v139/choquette-choo21a.html
[33] Z. Zhang, Y. Song, and H. Qi, "Age progression/regression by conditional adversarial autoencoder," in *Proc. IEEE Conf. Comput. Vis. pattern Recognit.*, 2017, pp. 5810–5818.
[34] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *Proc. Int. Conf. Comput. Vis.*2015, pp. 3730–3738,.
[35] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. 2017 IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 2261–2269.
[36] M. Azure, 2022. [Online]. Available: https://azure.microsoft.com/en-us/services/cognitive-services/face/
[37] L. Van Der Maaten, "Accelerating t-SNE using tree-based algorithms," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 3221–3245, 2014.
[38] A. Madry, "Towards deep learning models resistant to adversarial attacks," 2017, *arXiv: 1706.06083*.
[39] T. Xiao, Y.-H. Tsai, K. Sohn, M. Chandraker, and M.-H. Yang, "Adversarial learning of privacy-preserving and task-oriented representations," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 12434–12441.
[40] J. Jia and N. Z. Gong, "AttriGuard: A practical defense against attribute inference attacks via adversarial machine learning," in *Proc. 27th USENIX Secur. Symp.*, Baltimore, MD, USA, 2018, pp. 513–529.
[41] N. Homer et al., "Resolving individuals contributing trace amounts of DNA to highly complex mixtures using high-density SNP genotyping microarrayss," *PLoS Genet.*, vol. 4, no. 8, Art. no. e1000167.
[42] M. Backes, P. Berrang, M. Humbert, and P. Manoharan, "Membership privacy in MicroRNA-based studies," in *Proc. 2016 ACM SIGSAC Conf. Comput. Commun. Secur.*, Baltimore, MD, 2016, pp. 319–330.
[43] I. Hagestedt et al., "MBeacon: Privacy-preserving beacons for DNA methylation data," in *Proc. 2019 Netw. Distrib. System Secur. Symp. Internet Soc.*, San Diego, CA, USA, 2019.
[44] A. Pyrgelis, C. Troncoso, and E. D. Cristofaro, "Knock knock, who's there? Membership inference on aggregate location data," in *Proc. 2018 Netw. Distrib. System Secur. Symp. Internet Soc.*, San Diego, CA, USA, 2018.
[45] A. Pyrgelis, C. Troncoso, and E. De Cristofaro, "Under the hood of membership inference attacks on aggregate location time-series," 2019, *arXiv:1902.07456*.
[46] J. Liu, M. Juuti, Y. Lu, and N. Asokan, "Oblivious neural network predictions via MiniONN transformations," in *Proc. 2017 ACM SIGSAC Conf. Comput. Commun. Secur.*, 2017, pp. 619–631.
[47] K. Bonawitz et al., "Practical secure aggregation for privacy-preserving machine learning," in *Proc. 2017 ACM SIGSAC Conf. Comput. Commun. Secur.*, 2017, pp. 1175–1191.
[48] L. T. Phong, Y. Aono, T. Hayashi, L. Wang, and S. Moriai, "Privacy-preserving deep learning via additively homomorphic encryption," *IEEE Trans. Inf. Forensics Secur.*, vol. 13, no. 5, pp. 1333–1345, May 2018.
[49] N. Dowlin, R. Gilad-Bachrach, K. Laine, K. Lauter, M. Naehrig, and J. Wernsing, "CryptoNets: Applying neural networks to encrypted data with high throughput and accuracy," in *Proc. 33rd Int. Conf. Int. Conf. Mach. Learn.*, 2016, pp. 201–210.
[50] P. Mohassel and Y. Zhang, "SecureML: A system for scalable privacy-preserving machine learning," in *Proc. 2017 IEEE Symp. Secur. Privacy*, San Jose, CA, USA, 2017, pp. 19–38.
[51] C. Dwork and V. Feldman, "Privacy-preserving prediction," in *Proc. 31st Conf. On Learn. Theory*, 2018, pp. 1693–1702.
[52] O. Ohrimenko et al., "Oblivious multi-party machine learning on trusted processors," in *Proc. 25th USENIX Secur. Symp.*, Austin, TX, USA, 2016., 2016, pp. 619–636.
[53] C. Juvekar, V. Vaikuntanathan, and A. Chandrakasan, "Gazelle: A low latency framework for secure neural network inference," in *Proc. 27th USENIX Secur. Symp.*, 2018, pp. 1651–1669.

Ziqi Yang received the PhD degree from the National University of Singapore, in 2019. He is currently an assistant professor with the College of Computer Science and Technology and the Institute of Cyberspace Research (ICSR), Zhejiang University, Hangzhou, China. He is interested in cybersecurity and machine learning with a recent focus on the intersections between security, privacy, and machine learning.

Yiran Zhu is currently working toward the PhD degree with the College of Computer Science and Technology, Zhejiang University. Her research interests include AI security, privacy, binary analysis, and machine learning.

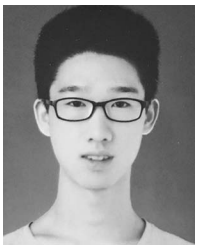
Jie Wan is currently working toward the PhD degree in cyber security with Zhejiang University. His research interests include machine learning and AI security.

ChuXiao Xiang received the bachelor's degree from the University of Science and Technology Beijing. He is currently working toward the graduate degree with the College of Computer Science and Technology, Zhejiang University. His research interest includes adversarial examples.

Tong Tang received the bachelor's degree from the Beijing University of Technology, and the MS degree in computer science from University College London. He is currently working toward the PhD degree in computer science with Zhejiang University. His research interests includes intersections between security, privacy, and machine learning.

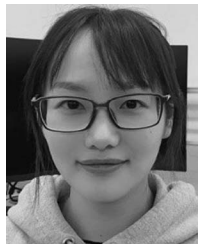
Yilin Wang received the bachelor's degree from Northeastern University. He is currently working toward the graduate degree with the College of Computer Science and Technology, Zhejiang University. His research interests include poisoning attacks, backdoor attacks, and federated learning.

Ruite Xu is currently working toward the graduate degree with the College of Computer Science and Technology, Zhejiang University. His research interests include backdoor, neural network watermarking, and misuse of generated content in generative artificial intelligence.



Lijin Wang is currently working toward the graduate degree with the Software College of Zhejiang University, majoring in Artificial Intelligence. His interests include data privacy and security as well as machine learning.

Fan Zhang (Member, IEEE) received the PhD degree from the Department of Computer Science and Engineering, University of Connecticut, USA, in 2012. He is an associate professor with the School of Cyber Science and Technology, College of Computer Science and Technology, Zhejiang University, and affiliated with College of Information Science and Electronic Engineering, Zhejiang University. His research interests include hardware security, system security, cryptography, and computer architecture.



Jiarong Xu received the BS degree from Donghua University, China, in 2016, and the PhD degree from Zhejiang University, China, in 2021. She is currently an assistant professor with the Department of Information Management and Business Intelligence, Fudan University, China. Her research interests include graph representation learning, social network analysis, and data mining.

Zhan Qin received the graduate degree from the Department of Computer Science and Engineering, State University of New York at Buffalo. He is the 100-Talents Young professor with the Institute of CyberSpace Research in Zhejiang University since 2019. He is an assistant professor with the University of Texas at San Antonio, since 2017. His research enables AI security and Data Security. Specifically, he focuses on adversarial attack and defense to deep learning model, privacy-preserving data collection, computation and publication. He also explores and develops novel security sensitive algorithms and protocols for computation and communication on IoT devices.