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

Michael Shell, *Member, IEEE,* John Doe, *Fellow, OSA,* and Jane Doe, *Life Fellow, IEEE*

Abstract—Neural networks are susceptible to data inference attacks, including the membership inference attack, the adversarial model inversion attack and the attribute inference attack. In this paper, we propose a method, namely PURIFIER, to defend against membership inference attacks. The PURIFIER works by transforming the confidence scores predicted by the target classifier and generating purified confidence scores, which are indistinguishable in individual shape, statistical distribution and prediction label between members and non-members of dataset. We set up experiments on a large number of widely adopted datasets and models. The results show that PURIFIER helps defend membership inference attacks with high effectiveness and efficiency, outperforming previous defense methods, and also incurs negligible utility loss. Besides, our further experiments show that PURIFIER is also effective in defending adversarial model inversion attacks and attribute inference attacks.

Index Terms—Membership Inference, Data Privacy.

1 INTRODUCTION

MACHINE learning as service has been widely adopted in various fields, ranging from face recognition to intelligent medicine. While providing convenience for people's life, machine learning that takes use of private information also brings potential danger of data leakage.

Usually, users have access to APIs given by the service providers, which return a confidence score vector or a label from the output of machine learning model. However, many studies indicate that the predicted information can also be used by adversaries for *data inference attack*, which aims at inferring secret information about data involved in the workflow of the target model[1], [2], [3]. Data inference attacks could be largely divided into three categories: *membership inference attacks* [1], [4], [5], [6], [7], [8], [9], [10], [11], *attribute inference attacks* [12] and *adversarial model inversion attacks* [13]. In this paper, we take the membership inference attack as a starting example to study the mitigation of data inference attacks.

In the membership inference attack, the adversary is asked to determine whether a given data sample is in the target model's training data. Many studies acknowledge that the confidence scores tell more prediction information beyond the label and thus they should be provided in the prediction results. Therefore, a number of approaches have been proposed to defend the membership inference attack while preserving the confidence scores [1], [4], [5], [14], [15], [16]. On the other hand, some studies believe that removing the confidence information in the prediction result is a way of defending the membership inference attacks. However, these defenses are broken by label-only attacks [6], [17], [8], whereby only the predicted label is exploited to infer the membership.

It has been widely recognized that one of the major reasons why membership inference attack works is that the prediction results are distinguishable between members and non-members. For example, when a model overfits on the training dataset, it behaves more confidently when handling inputs from members than non-members. Many studies exploit the prediction differences between members and non-members to perform membership inference attacks.

In general, the prediction differences can be manifested in the following three aspects. (1) *Individual shape*. The confidence scores of members and non-members differ in their individual shapes. This is because the target model often assigns a higher probability to the predicted result when given a member than a non-member. This is exploited by many attacks [4], [5] (2) *Statistical distribution*. Confidence scores of members and non-members are also distinguishable in their statistical distribution. Our experiments show that confidence scores on the members are more clustered in the encoded latent space, while those on non-members are more scattered. 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*. Differences in confidence scores on members and non-members can cause differences in prediction labels. Member samples have a higher probability of being correctly predicted than the non-member samples, which leads to a difference in classification accuracy. Various label-only attacks exploit such distinguishability to perform their attacks [6], [7], [8].

In this paper, we propose a defense method, namely PURIFIER, against membership inference attack. It takes the confidence scores predicted by the target model as input, and outputs transformed confidence scores, which behave indistinguishably between members and non-members in

- *M. Shell was with the Department of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA, 30332. E-mail: see http://www.michaelshell.org/contact.html*
- *J. Doe and J. Doe are with Anonymous University.*

Manuscript received April 19, 2005; revised August 26, 2015.

individual shape, *statistical distribution*, and *prediction label*. To be more specific, (1) to purify individual shape, we propose a novel training strategy to train a module named *confidence reformer*. The confidence reformer is trained on confidence scores predicted by the target model on non-members, which enables the model to learn the individual shape of confidence scores on non-members. The confidence scores on members become indistinguishable from those on non-members after being transformed by the confidence reformer. (2) To purify statistical distribution, we introduce Conditional Variational Auto-Encoder (CVAE) in the confidence reformer, with the purpose of adding Gaussian noises to confidence scores. The Gaussian noises scatter the originally statistically-clustered confidence scores, thus blurring the distinction between members and non-members in statistical distribution. (3) To purify prediction label, we propose a mechanism named *label swapper*. To defend label-only attacks which takes use of classification accuracy gap between members and non-members, label swapper modifies the prediction labels on members to the class of the second largest confidence at a specially designed rate. To further improve the robustness of PURIFIER, we design the label swapper to tolerate small perpetuations added by the attacker to members to sniff their membership privacy.

Experiments show that PURIFIER has a good performance in defending membership inference attacks. It turns out that PURIFIER is also effective in defending the attribute inference attack and the adversarial model inversion attack. We have performed extensive experiments on 7 widely used datasets, including CIFAR10, CIFAR100, Facescrub530, UTKFace, Purchase100, Texas and Location. For instance, when defending membership inference attacks, the accuracy of NSH attack [4] on CIFAR100 decreases from 76.98% to 50.91%, and the accuracy of Mlleak attack [5] on CIFAR100 decrease from 73.78% to 50.19%. On average, when applying PURIFIER, the inference accuracy of NSH attack on CIFAR100 is 2.23% to 9.28% lower than other defense methods. It is worth mentioning that although applying SELENA [18] usually has a similar inference accuracy as applying PURIFIER, the training time of PURIFIER is only 0.423 times of the target model, much lower than that of SELENA (22.42 times). When defending attribute inference attack, the accuracy to predict race on UTKFace decreases from 31.06% to 21.07% (almost random guessing). When defending model inference attack, the inversion loss on FaceScrub530 is raised 4+ times from 0.0114 to 0.0454 after applying PURIFIER. We believe that the purification process contributes to the removal of the redundant information (hidden in the confidence scores) that is useful to recover the input sample, and preserves only the essential semantic information for the prediction task. As a result, the adversary can obtain no more useful information than the prediction itself from the purified prediction results.

Contributions. In summary, we make the following contributions in this paper.

1) To the best of our knowledge, we are the first to study membership inference attack from the three aspects: *individual shape*, *statistical contribution* and *prediction label*.
2) We design PURIFIER to defend data inference attack, consisting of label swapper and confidence reformer. By transforming confidence scores, PURIFIER achieves indistinguishability between members and non-members in the above three aspects.
3) On the basis of our extensive experiments, PURIFIER outperforms other defense methods in both effectiveness and efficiency when defending data inference attacks.

2 INFERENCE ATTACKS ON MACHINE LEARNING

It has been shown that machine learning are vulnerable to various inference attacks [19], [1], [6], [13], which enables adversaries to get useful information about the target model from only the prediction APIs. Depending on the inference goals, these inference attacks generally fall into two classes, i.e., *model inference* and *data inference*. Specifically, model inference aims at obtaining the information about the target model itself such as its parameters and architecture [20], [21], [22], [23]. Data inference, on the contrary, focuses on extracting information about the data on which the target model operates [24], [2], [19], [1], [25], [26], [13], [6]. In this paper, we concentrate on three of the most important and exemplary data inference attacks, notably membership inference attack, attribute inference attack and model inversion attack. In this section, we first introduce these three data inference attacks and then introduce existing defenses. Finally, we analyze the limitations of existing defense mechanisms.

2.1 Data Inference Attacks

Membership inference, attribute inference and model inversion attacks are three 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 membership inference attack, the attacker is asked to determine whether a given data record is part of the training data of the target model.

Confidence-based Attack [1]. Shokri et al. introduces membership inference against black-box models, where the attacker has access only to the prediction scores of the target model. To infer the membership, the attacker trains a binary classifier (also referred to as attack model) which takes as input the confidence scores of the target model on a given data sample and predicts the data sample to be a member or non-member of the training dataset of the target model. Prior to training the attack model, the attacker trains a set of shadow models on an auxiliary dataset drawn from the same data distribution as the target model's training data to replicate the target model. The attack model is then trained on the confidence scores predicted by the shadow models instead of the target model on the members and non-members of the shadow models' training data.

NSH Attack [4]. This is a confidence & label-based membership inference attack proposed by Nasr, Shokri and Houmansadr. The attacker is assumed to have knowledge of the membership labels, and thus can directly query the target classifier to get the confidence score vectors of members and non-members without training the shadow model.

Mlleaks Attack [5]. In this attack, the attacker knows the ground truth of auxiliary dataset but does not know their membership labels. Therefore, a shadow model is required to replicate the target model.

Adaptive Attack [5]. This is a sort of membership inference attack conducted on the pruned model with defense when the attacker knows the defense.

BlindMI Attack [9]. This is a membership inference attack which probes the target model and extracts membership semantics via differential comparison. The high-level idea is that BlindMI first generates a dataset with non-members via transforming existing samples into new samples, and then differentially moves samples from a target dataset to the generated, non-member set in an iterative manner. If the differential move of a sample increases the set distance, BlindMI considers the sample as non-member or vice versa.

Label-only Attack [7], [17]. Label-only attack is a black-box membership inference attack conducted based on only the output label of the target model. In the Gap Attack [17], the attacker is assumed to have the ground truth of the data sample and predicts that it is a member if and only if the target classifier gives the correct label on the sample. In the Transfer attack [7], auxiliary dataset is re-labeled by querying the target model, thus the adversary can train a shadow model to lauch a score-based membership inference attack locally. In the Boundary attack [7], auxiliary dataset is not available. The adversary utilize adversarial example techniques to perturb the input to mislead the target model and consider the samples with perturbations larger than a threshold as member samples.

MIA from First Principles [10]. Nicholas Carlini et al. argue that membership inference attacks are currently evaluated using average-case "accuracy" metrics that fail to characterize whether the attack can confidently identify any members of the training set. They argue that attacks should instead be evaluated by computing their true-positive rate at low (e.g., <0.1%) false-positive rates, and find that most prior attacks perform poorly when evaluated in this way. They perform MIA by training N shadow models on random samples from auxiliary dataset, so that half of models are trained on the target sample, and half are not (they call these respectively IN and OUT models). The membership is predicted by comparing confidence scores on the target sample of target model with those of IN and OUT models.

Enhanced MIA [11]. Jiayuan Ye et al. present a comprehensive hypothesis testing framework that supports not only formally expressing the prior work in a consistent way, but also designing new membership inference attacks that use reference models to achieve a significantly higher power (true positive rate) for any (false positive rate) error. They propose 4 attacks in their work, called S, P, R and D, each with diffrent setting on the boundary between members and non-members.

Attribute Inference Attack. Attribute inference aims to infer sensitive attributes [19], [27], [26], [28] or statistical information [24] about the training data. For example, the race attribute of the samples can be inferred with an API to the gender classifier on UTKFace dataset [12]. The adversary has an auxiliary dataset and queries the target model to get the confidence scores. Then the adversary train a classifier on the confidence scores, using the race attribute as label. Thus the race of a sample can be predicted with the output of gender classifier.

Model Inversion Attack. Model inversion aims to reconstruct the input data from its confidence scores predicted by the target model. Fredrikson et al. [2] proposed a method to infer a representative sample of a training class against a white-box target model. It casts the inversion task as an optimization problem in the input domain to find the best representative for a given class. Yang et al. [13] proposed a model inversion attack in the black-box setting. Specifically, they train a separate inversion model on an auxiliary dataset which acts as the inverse of the target model. The inversion model takes the confidence scores of the target model as input and tries to reconstruct the original input data.

2.2 Defenses against Data Inference Attacks

Previous defense mechanisms against data inference attacks are mostly limited to mitigating membership inference attacks. Unfortunately, little has been studied about the approach of simultaneously defending attribute inference attacks and model inversion attacks on classification models. Therefore, we introduce existing defenses against membership inference attacks as typical examples in the literature defending data inference attacks.

Min-Max Game [4]. Nasr et al. propose to add an adversarial regularizer to the loss function of the target model such that it is trained to minimize the prediction loss and also to maximize membership privacy. The training process is formulated as a min-max optimization problem.

MemGaurd [15]. Jia et al. study to transform the confidence score vector into an adversarial example to evade the membership classification of the attack model. Specifically, the defender adds carefully-crafted noise to the confidence score vector predicted by the target model so as to turn it into an adversarial example. To this end, the defender first trains its own "attack model" which works similarly as the attacker's attack model, and thus it can craft the adversarial example against its attack model in a white-box manner.

Model Stacking [5]. Model stacking is essentially an ensemble approach which combines multiple simple classifiers as a complicated one to make the final prediction. It is often used as a way of reducing overfitting, and can be leveraged to mitigate membership inference attacks.

MMD defense [29]. Li et al. propose a defense method that aims to close the gap by intentionally reduces the training accuracy. The training process attempts to match the training and validation accuracies, by means of a new set regularizer using the Maximum Mean Discrepancy between the softmax output empirical distributions of the training and validation sets.

SELENA [18]. Xinyu Tang et al. propose a new framework to train privacy-preserving models that induce similar behavior on member and non-member inputs to mitigate membership inference attacks, SELENA, which has two major components. The first component is an ensemble architecture for training called Split-AI. This architecture splits the training data into random subsets and trains a model on each subset of the data. An adaptive inference strategy is used at test time. The ensemble architecture aggregates the outputs of only those models that do not contain the input sample in their training data. The second component, Self-Distillation, (self-)distills the training dataset through the Split-AI ensemble, without using any external public datasets.

Relax Loss [30]. Existing works evidence a strong connection between the distinguishability of the training and testing loss distributions and the model's vulnerability to MIAs. Motivated by existing results, Dingfan Chen et al. propose a training framework based on a relaxed loss with a more achievable learning target, which leads to narrowed generalization gap and thus reduces privacy leakage.

One-Hot Encoding. It encodes a confidence vector into a one-hot vector (i.e., the entry with the largest confidence is set to 1 and the other entries are all 0).

2.3 Limitations of Existing Defenses

Previous studies of defense mechanisms against the membership inference attack did not discuss their impact on the attribute inference attack and model inversion attack which is one of the important data inference attacks that threaten the security and privacy of machine learning data. To the best of our knowledge, no known defense method of membership inference, attribute inference and model inversion attacks is available.

It is shown that overfitting is not the only reason that causes membership inference attack [1]. Even if different machine learning models are overfitted to the same degree, they could leak different amounts of membership information. Specifically, due to their different structures, they might "remember" different amounts of information about their training data. Actually, the attacker exploits the information about how the target model's confidence scores distinguish members from non-members to launch membership inference attack [1]. As what existing defense mechanisms already do, reducing overfitting contributes to the decrease of such distinguishability. However, such defense methods could be more effective if the distinguishability can be directly reduced.

Meanwhile, the efficiency of some existing defense methods is not high enough. For example, when performing MemGaurd [15], the defender adds carefully-crafted noise to the confidence scores so as to turn it into an adversarial example. It takes much time to find the suitable noise, causing that MemGaurd is not efficient enough.

3 Problem Statement

We focus on classification models of neural networks, i.e., a machine learning classifier F is trained on its training dataset D_{train} to map a given sample $\mathbf{x}$ to a specific class based on the confidence vectors $F(\vec{x})$ which is the classifier output. There are three parties in our problem, namely *model owner*, *attacker* and *defender*.

3.1 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 $p_x(\vec{x})$. The classifier F is trained with the goal of making predictions on unseen data which we refer to as test dataset D_{test}. Let $\vec{x}$ represent the data drawn from p_x, and $\vec{y}$ be the vectorized class of $\vec{x}$. The training objective is to find a function F to well approximate the relation between each data point $(\vec{x}, \vec{y})$. Formally, we have $F : \vec{x} \mapsto \vec{y}$. 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 $\vec{x} \in D_{test}$ through the prediction APIs. The classifier F returns a confidence score vector $F(\vec{x})$ 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 $F(\vec{x})_i$ is the probability of the data $\vec{x}$ belonging to class i. We usually take the class with the maximum probability to be the predicted label of the data $\vec{x}$.

3.2 Attacker

The attacker aims at performing data inference attacks against the target classifier F. We focus on membership inference attack [1], attribute inference attack [12] and model inversion attack [13], [2]. We assume that the attacker has a black-box access to the classifier F, where the attacker can only query F with its data sample $\vec{x}$ and obtain the prediction scores $F(\vec{x})$. The attacker is also assumed to have an auxiliary dataset D_{aux} such as a set of data samples drawn from a similar data distribution as the target classifier's training data distribution.

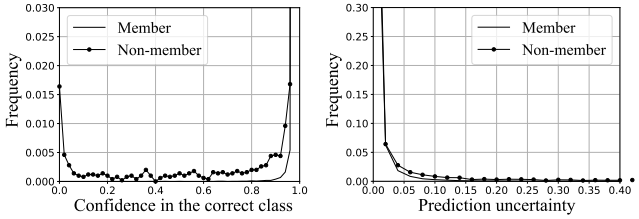
In the membership inference attack, the attacker is asked to determine whether a given data record $\vec{x}$ is part of the training data D_{train} according to $F(\vec{x})$. A common approach is to leverage the auxiliary dataset D_{aux} and train a membership classifier which takes $F(\vec{x})$ as input and predicts the membership.

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

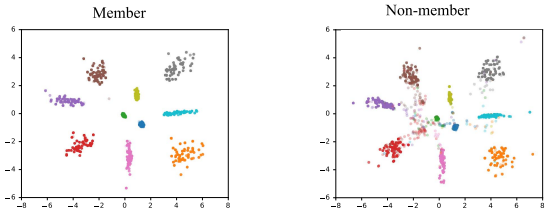
Individual shape. The distinguishability of individual shape exposes because a member sample and a non-member sample differ in the distribution of confidence scores they get. For example, the classifier usually predicts the class of member data with a high degree of confidence, which means that the value of the maximal score in the vector would be larger for member sample than non-member sample. This can be shown by Figure 1 (a), where we calculate the frequency of confidence on members and non-member in correct class and prediction uncertainty. It can be clearly seen that there are more non-members than members at low confidence, and more members than non-members at high confidence.

Statistical distribution. The distinguishability of statistical distribution means that confidence scores on member samples of the same class are more similar than those on non-member samples. In an intuitive way, we reduce the dimension of the latent vectors of the confidence scores, and plot them in a coordinate graph, as shown in Figure 1 (b). We can see that members data of the same class are more closely clustered than non-members.

Prediction label. The prediction label for a sample is the class corresponding with the maximal score in confidence vector. The distinguishability of prediction labels directly leads to the difference between training accuracy and testing



(a) Confidence and uncertainty of member and non-member



(b) Distribution of member and non-member

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

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.

In the attribute inference attack, the attacker aims at inferring the additional sensitive attribute about the given data record $\vec{x}$ according to the $F(\vec{x})$. A classifier is trained on $F(\vec{x})$ and it takes $F(\vec{x})$ as input and is able to infer other attributes of $\vec{x}$ [12].

In the model inversion attack, the attacker aims at inferring a reconstruction of $\vec{x}$ from $F(\vec{x})$. In the black-box setting, the attacker trains a separate inversion model on D_{aux} which acts as the inverse of the target F. The inversion model takes $F(\vec{x})$ as input and is able to reconstruct $\vec{x}$ [13].

3.3 Defender

The defender could be the model owner or a third party who has access to the target classifier's prediction results. The defender has the train dataset D_{train} and a reference dataset D_{ref} which is composed of non-member data to implement the defense. This strategy ensures that the defense is compatible with existing models without retraining the target classifier. 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, attribute inference attack and the model inversion 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 least loss of utility (i.e., 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.

4 Approach: Purifier

We propose Purifier as a defense against data inference attacks. The main idea is to transform the confidence vector $F(x)$ so that it appears indistinguishable on members and non-members. Purifier consists of a *label swapper* H and a *confidence reformer* G, as shown in Figure 2. The *label swapper* H takes the original confidence score vectors, and modifies the predicted labels of members to reduce the gap of classification accuracy between members and non-members, achieving indistinguishability of prediction label. The *confidence reformer* G takes as input the swapped confidence score vectors from H and reforms them as if they were predicted on non-members, achieving indistinguishability of individual shape and statistical distribution.

In the rest of this section, we expand on how the three distinguishabilities lead to the privacy leak of membership and how Purifier is designed to make the confidence score on members and non-members indistinguishable from the aforementioned three aspects.

4.1 Design of Confidence Reformer

In order to achieve individual and statistical indistinguishability between members and non-members, Purifier reforms the confidence scores with the *confidence reformer* G, which is a CVAE. G takes the confidence vector $H(F(\vec{x}))$ (the confidence vector modified by *label swapper*) as input, with the corresponding label l being the condition. $H(F(\vec{x}))$ first goes through the encoder, where it is mapped to the encoded latent space $\vec{r}$. The decoder then maps the confidence vector back from the latent space $\vec{r}$, and the reformed confidence vector $G(H(F(\vec{x}))|l)$ is obtained. G is trained on the confidence scores predicted by F and modified by H on the defender's reference dataset D_{ref}, which consists of non-member samples. Moreover, to preserve the classification accuracy, we also take the label loss into consideration when training G. Formally, G is trained to minimize the following objective function.

$$L(G) = \mathop{\mathbb{E}}_{\vec{x}\sim p_r(\vec{x})}[\mathcal{R}((G(H(F(\vec{x}))|l), F(\vec{x})) + \lambda\mathcal{L}((G(H(F(\vec{x}))|l), l)] \quad (1)$$

where $p_r(\vec{x})$ represents the conditional probability of $\vec{x}$ for samples in D_{ref}, l represents the label of $H(F(\vec{x}))$ (i.e., $l = \mathbf{argmax}(H(F(\vec{x})))$). $\mathcal{R}$ is a reconstruction loss function (L_2 norm) and $\mathcal{L}$ is the cross entropy loss function. The parameter λ is to adjust the ratio of two loss functions during training.

The training process of *confidence reformer* goes as Algorithm 1. For each epoch, we first draw a mini-batch of data points $\{(\vec{x}_{ref}, y_{ref})\}_{j=1}^{q}$ from the reference set D_{ref}. Then we query the target classifier F to obtain the confidence scores $\vec{c}_{r_j}$, and the label l_{r_j}. After that, the loss is calculated on the objective function 1 and gradient descent is used to update the parameters θ of *confidence reformer* G. With this training process, while preserving the classification

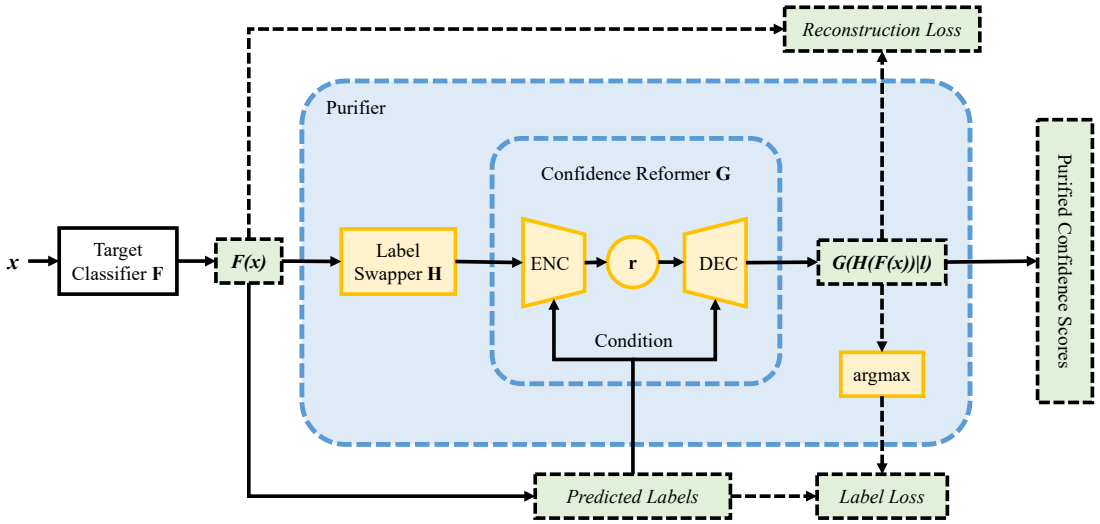


Fig. 2: Architecture of PURIFIER. PURIFIER consists of a *label swapper* H and a *confidence reformer* G. H can reduce the gap of classification accuracy between members and non-members by modifying the predicted labels of specific training data. G is a Conditional Variational Auto-encoder(CVAE), with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

Algorithm 1: Training process of *confidence reformer*.

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

$\theta \leftarrow$ **initialize**(G_θ) ;
for $p = 1$ *to* P **do**
  for *each mini-batch* $\{(\vec{x}_{ref_j}, y_{ref_j})\}_{j=1}^{q} \subset D_{ref}$ **do**
    $\vec{c}_{r_j} \leftarrow H(F(\vec{x}_{ref_j}))$;
    $l_{r_j} \leftarrow$ **onehot**(**argmax**$(\vec{c}_{r_j})$) ;
    $g \leftarrow \nabla_\theta \frac{1}{q} \sum_{j=1}^{q} \mathcal{R}(G_\theta(\vec{c}_{r_j}|l_{r_j}), \vec{c}_{r_j}) + \lambda \mathcal{L}(G_\theta(\vec{c}_{r_j}|l_{r_j}), l_{r_j})$;
    $\theta \leftarrow$ **updateParameters**(η, θ, g);
  end
end
return G_θ

accuracy, G also learns the pattern of *individual shape* on non-member samples, through which G could remove difference in the individual shape of input confidence vector, achieving individual indistinguishability.

To mitigate the difference in statistical distribution between members and non-members, *confidence reformer* G introduces Gaussian noises in the latent space r, where the label l is used as the condition. During the training process, the reconstruction loss $\mathcal{R}$ encourages the decoder of G to generate confidence scores that have a similar pattern as the non-member ones on D_{ref} (non-members) with the same label l. Although noises introduced in the latent space $\vec{r}$ will increase the reconstruction error, as a result, G adaptively learns a robust latent representation that could preserve the statistical distribution of the non-members of label l even if noises are added. 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.

4.2 Design of Label Swapper

With *confidence reformer* G, we can achieve indistinguishability of individual shape and statistical distribution. However, to keep the accuracy of classifier, G hardly changes the confidence class when transforming confidence vector, which means that the *confidence reformer* can't cope with the distinguishability of prediction label well.

To handle this problem, we design a mechanism named *label swapper*. The *label swapper* H modifies the prediction labels of members to reduce the gap of classification accuracy between members and non-members. In detail, H randomly selects training samples to replace their predicted labels with the ones with the second largest predicted scores at a certain swap rate p_{swap}.

$$p_{swap} = (acc_{train} - acc_{test})/acc_{train} \tag{2}$$

where acc_{train} and acc_{test} are the training accuracy and the test accuracy of the target classifier respectively. Note that H only swaps member data. At this swap rate p_{swap}, the training accuracy can decrease to testing accuracy, achieving indistinguishability of the prediction label.

A naive implementation is as follows. When the input confidence vector belongs to member, H swap its label with probability p_{swap}. It leads to the problem that whether to swap the prediction label of a member sample is uncertain each time. Thus the attacker can get different prediction results with replay attacks, which exposes a more severe distinguishability between members and non-members.

To mitigate the replay attacks and further improve the efficiency of label swapper, we design a prediction indexing set P_index instead of the whole D_train for the label swapper to swap labels. Specifically, as Algorithm 2 presents, we first select the data 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 $\vec{c}_j$ of the sample $(\vec{x}_{train_j}, y_{train_j}) \in D_{swap}$. After mapping the original confidence vector $\vec{c}_j$ to its hash value h_j, h_j is added to the prediction indexing set P_{index}. The setting of P_{index} also helps to improve efficiency. It is

Algorithm 2: Pre-process of *label swapper*.

Input: The training dataset D_{train}, the target classifier F, size of the data need to be modify the labels t
Output: The prediction indexing set P_{index}

$P_{index} \leftarrow \emptyset$;
$D_{train} \leftarrow \mathbf{shuffle}(D_{train})$;
$D_{swap} \leftarrow \{(\vec{x}_{train_j}, y_{train_j})\}_{j=1}^{t} \subset D_{train}$;
for *each* $(\vec{x}_{train_j}, y_{train_j}) \in D_{swap}$ **do**
  $c_j \leftarrow F(\vec{x}_{train_j})$;
  $r_j \leftarrow \mathbf{getLatent}(c_j)$;
  $h_j \leftarrow \mathbf{Hash}(c_j, r_j)$;
  $P_{index} \leftarrow P_{index} \cup \{h_j\}$;
end
return P_{index}

Algorithm 3: Inference process of PURIFIER.

Input: The input sample $\vec{x}$, the target classifier F, the trained *confidence reformer* G_θ, the prediction indexing set P_{index}, a helper function $Swap$ that take a vector as input and swap the largest element with the second largest element, the number of neighbors of kNN k, the distance precision of kNN d
Output: The purified confidence score $c_{purified}$

$c \leftarrow F(\vec{x})$;
$r \leftarrow \mathbf{getLatent}(c)$;
$h \leftarrow \mathbf{Hash}(c, r)$;
if $\mathbf{kNN}(h, P_{index}, k, d)$ **then**
  $c \leftarrow Swap(c)$;
end
$l \leftarrow \mathbf{onehot}(\mathbf{argmax}(c))$;
$c_{purified} = G_\theta(c|l)$;
return $c_{purified}$

challenging for the label swapper to efficiently store and index the member information in D_{train}, while P_{index} has a much smaller dimension and size than D_{train}.

We should note that small perturbations may be added to the input data by attackers to indirectly infer membership of a target member sample $x \in D_{swap}$. More specifically, after adding small perturbation to a member, the sample isn't in P_{index}, so its label won't be swapped, which causes the member to be less robust to noise. Thus we can't just simply match whether an input confidence vector is in P_{index}. Instead, H uses k nearest neighbor (kNN) to identify these suspicious noisy members. If the distances between the confidence vector on a sample and those on the k-nearest members are all less than a certain parameter d, then the sample is considered as a member and should have its label swapped.

4.3 Defense Process of PURIFIER

After the training process of *confidence reformer* and pre-process of *label swapper*, we can perform defense with the trained *confidence reformer* G_θ and prediction indexing set P_{index}. G_θ is used to transform the confidence scores, and P_{index} is used to determine whehter to confidence scores need to swap.

As algorithm 3 shows, in the inference stage, given an input sample $\vec{x}$, we first query the target classifier F to get the confidence scores c. Then, we input c into the *label swapper* H. H checks if c has a match in P_{index} using kNN and swaps the largest score with the second largest score if c is matched. At this stage, c is indistinguishable in prediction label. Then, we input c into the *confidence reformer* G, with the prediction label l being the condition, to get the purified confidence vector $c_{purified}$. This ensures indistinguishability in terms of individual shape and statistical distribution. Finally, PURIFIER returns the purified confidence scores $c_{purified}$.

5 EXPERIMENTS

In this section, we perform experiments on various widely adopted datasets and models to test the performance of PURIFIER. We first introduce the dataset and model settings in detail. Then we conduct various experiments with the purpose of showing that: (1) PURIFIER is effective in defending membership inference attack, attribute inference attack and model inversion attack. (2) PURIFIER outperforms other defense methods in effectiveness. (3) with the defense of PURIFIER, individual, statistical and label indistinguishability are achieved. (4) PURIFIER is efficient. Also, we furthur discuss what if swapping confidence reformer and label swapper, and proving that PURIFIER has a better performance with label swapper before confidence reformer. In the end, we conduct three additional experiments to more comprehensively show the performance of PURIFIER.

5.1 Dataset and Model Settings

5.1.1 Dataset Settings

We use CIFAR10, Purchase100, FaceScrub530, UTKFace, CIFAR100, Texas and Location datasets which are widely adopted in previous studies on membership inference attacks, attribute inference attacks and model inversion attacks.

CIFAR10 [1], [5], [7], [10]. It is a machine learning benchmark dataset for evaluating image recognition algorithms. It consists of 60,000 color images, each of size 32 x 32. The dataset has 10 classes, where each class represents an object (e.g., airplane, car, etc.)

Purchase100 [1], [4], [5], [7]. This dataset is based on Kaggle's "acquired valued shopper" challenge. [1] We used the preprocessed and simplified version of this dataset [1]. It is composed of 197,324 data records and each data record has 600 binary features. The dataset is clustered into 100 classes.

FaceScrub530 [13]. This dataset consists of URLs for 100,000 images of 530 individuals. We obtained the preprocessed and simplified version of this dataset from [13] which has 48,579 facial images and each image is resized to 64 $\times$ 64.

UTKFace[31], [12]. This dataset consists of URLs for 22000 images of individuals. We train the classifier to classify the gender attribute and use the race attribute as the sensitive attribute in our experiments.

CIFAR100[1], [10]. It is a machine learning benchmark dataset for evaluating image recognition algorithms with 10 classes. It consists of 60,000 color images, each of size 32 x 32. The dataset has 100 classes and each class has 600 images.

Texas[1], [17]. We use the same data as previous studies, which cluster the data with 6169 attributes to 100 classes.

Location[1], [17]. This dataset is based on the publicly available set of mobile users' location "check-ins" in the Foursquare social network, restricted to the Bangkok area and collected from April 2012 to September 2013. The record of Location has 446 attributes and has clustered into 30 classes.

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

TABLE 1: Data allocation. A dataset is divided into training set D_1 of the target classifier, reference set D_2 and test set D_3. In membership inference attack, we assume that the attacker has access to a subset D^A of D_1 and a subset $D^{'A}$ of D_3.

Dataset	D_1	D_2	D_3	D^A	$D^{'A}$
CIFAR10	50,000	5,000	5,000	25,000	2,500
Purchase100	20,000	20,000	20,000	10,000	10,000
FaceScrub530	30,000	10,000	8,000	15,000	4,000
UTKFace	12,000	5,000	5,000	6,000	2,500
CIFAR100	50,000	5,000	5,000	25,000	2,500
Texas	10,000	10,000	10,000	5,000	5,000
Location	1,600	1,600	1,600	800	800

Table 1 presents the data allocation in our experiments. We divide each dataset into the target classifier's training set D_1, the validation set D_2 and the test set D_3. They have no overlap with each other. In membership inference attack and attribute inference attack, we assume that the attacker has access to a subset D^A of D_1 and a subset D'^A of D_3 to form its auxiliary dataset D_{aux}. We use the remaining data in D_1 and D_3 to test the membership inference accuracy. We use D_2 as the reference dataset for defenses that require such a dataset, e.g., MemGaurd [15], Min-Max [4] and our approach. In the model inversion attack, for the FaceScrub530 classifier, the attacker uses a CelebA [32] dataset to train the inversion model, following the same setting in [13]. For other classifiers, the attacker samples 80% from D_1, D_2 and D_3 respectively to train the inversion model, and uses the remaining 20% data to test the inversion error.

5.1.2 Target Classifier Setting

We use the same model architectures as in previous work [4], [13], [7] to train the target classifiers. That is,

For **CIFAR10** and **CIFAR100** datasets, we use DenseNet121 [33]. We train our classifier with stochastic gradient descent (SGD) optimizer for 350 epochs with a learning rate of 0.1 from epoch 0 to 150, 0.01 from 150 to 250, and 0.001 from 250 to 350. The classifier is regularized with L_2 regularization (weight decay parameter 5e-4).

For **Purchase100** dataset, we use the same model and training strategy as in previous work [4] to train the target classifier. It is a 4-layer fully connected neural network.

For **FaceScrub530** dataset, we use the same conventional neural network and the same training strategy as in previous work [13] to train the target classifier.

For **UTKFace** dataset, we use the same neural network and the same training strategy as used in FaceScrub530 dataset, except that the output layer dimension is changed to 2.

For **Texas** dataset, we use a 4-layer fully connected neural network with the Tanh as the activation function.

For **Location** dataset, we use a 3-layer fully connected neural network with the Tanh as the activation function.

5.1.3 PURIFIER Setting

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. We use ReLU and batch normalization in hidden layers. We train PURIFIER on Purchase100 dataset for 150 epochs, CIFAR10, Texas and Location datasets for 100 epochs, FaceScrub530 and CIFAR100 datasets for 300 epochs. We use Adam optimizer with the learning rate 0.01 for CIFAR10, 0.0001 for Purchase100, Texas and Location, and 0.0005 for Facescrub530 and CIFAR100.

5.1.4 Attack Setting

In our experiments, we implement the following data inference attacks.

NSH attack [4]. The attacker take use of both the membership labels and ground truth of D_{aux}, and no shadow is trained. The membership classifier makes use of both the confidence scores and the ground truth of a data sample to predict its membership.

Mlleaks attack[5]. We use half of D_{aux} to train the shadow model which has the same architecture as the target classifier and use the whole D_{aux} to train the membership classifier with their labeled membership information in terms of the shadow model. The membership classifier is a multi-layer perceptron with a 128-unit hidden layer and a sigmoid output layer. All weights were initialized with normal distribution with a mean of 0 and standard deviation of 0.01, and all biases are initialized to 0. We use the Adam optimizer with a learning rate of 0.001. The number of training epochs is set to 50 for each dataset.

Adaptive attack[5]. This is an adaptive version of the Mlleaks attack, where the attacker is assumed to know all the details of the defender's PURIFIER and its training data D_2. Hence, the attacker trains the same PURIFIER and appends it to the shadow model. The membership classifier is then trained on the purified confidence score vectors.

BlindMI attack [9]. We consider BlindMI-DIFF-w/, where the attacker is assumed to know the soft label and the ground truth of the target dataset. We use sobel to generate non-member samples. The size of the non-member dataset $|S_{nonmem}|$ is 20.

Label-only attack [6], [7]. In the Transfer attack, auxiliary dataset D_{aux} is re-labeled by querying the target model. The shadow models that we adopt share the same architecture with the target model. Three different thresholds are chosen in previous work, and we consider the one that gives the best result on D_{aux}. In the Boundary attack, we adopt HopSkipJump noise, with a total evaluation of 15000 per sample, which ensures the attack performance is stable. Similarly, we report the results on the L_2 norm. Due to high query-complexity, the results are reported on a subset that consists of 1000 samples.

MIA from first principles [10]. This paper provides two algorithms: online and offline. To implement this attack, it is necessary to first train shadow models. We train 16 shadow models and adopt online algorithm in our experiments. In the experimental results, we refer to it as FP attack for short.

Enhanced MIA [11]. This paper proposes 4 attack methods, namely S, P, R and D. And training a number of shadow models is also necessary in this attack. We adopt method D and train 16 shadow models in our experiments.

Adversarial model inversion attack [13]. The attacker trains an inversion model on D_{aux} to perform the model inversion attack.

Attribute inference attack [12]. The attacker trains a classification on D_{aux} to predict the additional sensitive attribute

taking the confidence vectors as input. The classifier is a multi-layer perceptron. All weights were initialized with normal distribution with a mean of 0 and standard deviation of 0.01, and all biases are initialized to 0. We use the Adam optimizer with a learning rate of 0.00001. The number of training epochs is set to 1500 for the UTKFace dataset.

5.1.5 Metric Setting

We use the following metrics to measure the defense performance, inversion error and efficiency of a defense method.

Classification Accuracy. It is measured on the training set and the test set of the target classifier. It reflects how good the target classifier is on the classification task.

Inference Accuracy. This is the classification accuracy of the attacker's attack model in predicting the membership and additional sensitive attribute of input samples.

Inversion Error. We measure the inversion error by computing the mean squared error between the original input sample and the reconstruction. For the FaceScrub530 classifier, it is measured on D1 and D3. For other classifiers, it is measured on the 20% of D1 and D3 respectively.

Efficiency. Following [13], 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.

5.2 Effectiveness of PURIFIER

5.2.1 Effectiveness against Membership Inference Attack

Table 2 presents the defense performance of PURIFIER against different membership inference attacks. For each classification task, PURIFIER decreases the attack accuracy and preserves the classification accuracy. PURIFIER reduces the accuracy of NSH attack significantly on different datasets. For instance, it reduces the accuracy of NSH attack from 69.34% to 51.56% in FaceScrub530 dataset. As for Mlleaks attack, PURIFIER effectively reduces the attack accuracy to nearly 50%. Compared with the Mlleaks attack, the performance of the adaptive attack does not show a large difference where PURIFIER reduces the accuracy to nearly 50%. PURIFIER is also effective against BlindMI attack. For example, PURIFIER reduces the accuracy of BlindMI from 62.61% to 50.00% on FaceScrub530 dataset.

To further study the effectiveness of PURIFIER against stronger attackers, we evaluate the performance of PURIFIER against NSH attack and Mlleaks attack which use the same number of the data as the target model to train the shadow model instead of half the number. As Table 3 shows, PURIFIER is also effective to reduce the attack accuracy of the attackers with more powerful shadow models. For instance, the attack accuracy of the Mlleaks attack drops to 50.01% from 70.20%, which means that PURIFIER is able to mitigate stronger attacks which exploting more information (e.g., an attacker has more data to train shadow models) than what we assumed in the main paper.

5.2.2 Effectiveness against Adversarial Model Inversion Attack

We further investigate the defense performance of PURIFIER against adversarial model inversion attack. We train an inversion attack model on top of each classifier with or without defense both on the MNIST and the FaceScrub530 datasets. Although PURIFIER is designed to protect models from membership inference attacks, it turns out that PURIFIER is also effective in mitigating model inversion attack. Figure 3 shows the results of our experiment on adversarial model inversion attack on FaceScrub530. We quantify the inversion quality by reporting the average facial similarity scores compared with the ground truth using the Microsoft Azure Face Recognition service [34], which is shown on the left side of Figure 3. The less the number is, the less similarity the reconstructed samples share with the original samples. With the defense of PURIFIER, the reconstructed samples have smallest similarity scores with original samples, which means PURIFIER outperforms other defense methods. Similarly, Figure 4 shows the results against model inversion attack on MNIST. We report the inversion error on the left side of Figure 4 to quantify the inversion performance. As illustrated in Figure 4, the attacker is able to reconstruct nearly identical images when no defense is used. However, it is much more difficult for the attacker to reconstruct the image from the purifier-defended model. The reconstructed images lose a lot of details compared with the original ones, only representing a blurred image of their classes.

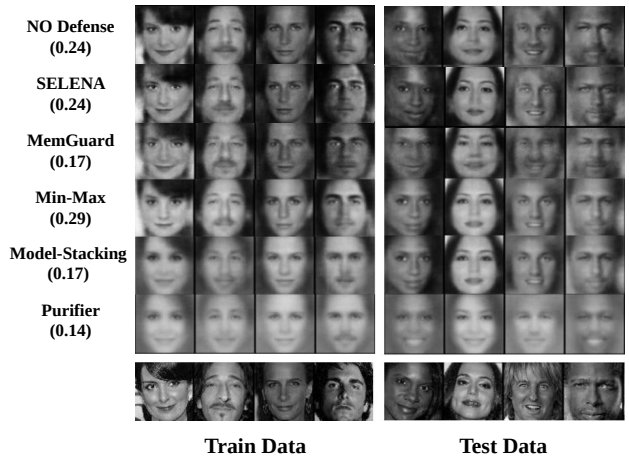


Fig. 3: Model inversion attack against the FaceScrub530 classifier defended by different approaches.

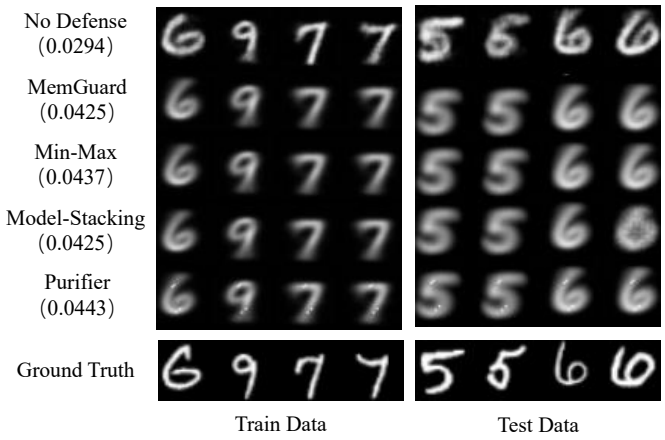


Fig. 4: Model inversion attack against the MNIST classifier defended by different approaches. Numbers on the left indicate inversion errors

TABLE 2: Defense performance of PURIFIER against various attacks. Results of Transfer attack and Boundary attack are reported in AUC [7]. Note that N.A. means that the setting is not applicable.

Dataset	Defense	Utility		Membership Inference Attack Accuracy/AUC									Inversion Error
		Train acc.	Test acc.	NSH	Mlleaks	Adaptive	BlindMI	Label only attacks			FP	Enhanced	L2 norm
								Gap	Transfer	Boundary			
CIFAR10	None	99.99%	95.92%	56.03%	56.26%	N.A.	54.76%	52.04%	0.5048	0.5214	57.96%	64.49%	1.4357
	Purifier	95.93%	95.92%	50.91%	50.03%	50.00%	49.89%	50.01%	0.5010	0.4684	54.72%	50.12%	1.4939
Purchase100	None	100.00%	84.36%	70.36%	64.43%	N.A.	69.82%	57.82%	0.5431	N.A.	77.57%	77.19%	0.1426
	Purifier	84.47%	84.36%	50.00%	50.00%	50.25%	49.48%	50.06%	0.4961	N.A.	75.06%	50.16%	0.1520
FaceScrub530	None	100.00%	77.68%	69.34%	75.04%	N.A.	62.61%	61.16%	0.5869	0.7739	87.09%	83.62%	0.0114
	Purifier	77.51%	77.28%	50.08%	50.66%	50.12%	50.31%	50.12%	0.4990	0.5069	53.34%	50.84%	0.0454
CIFAR100	None	100.00%	69.98%	76.98%	73.78%	N.A.	76.34%	76.86%	0.5980	0.6668	82.37%	82.54%	0.9073
	Purifier	66.35%	66.34%	50.91%	50.19%	50.45%	49.87%	50.01%	0.4938	0.3742	77.32%	50.72%	0.9354
Texas	None	79.17%	50.91%	66.37%	58.93%	N.A.	53.88%	69.02%	0.5431	N.A.	72.28%	74.06%	N.A.
	Purifier	50.99%	47.88%	50.92%	46.38%	54.48%	52.35%	51.56%	0.4926	N.A.	53.34%	50.84%	N.A.
Location	None	100.00%	60.44%	82.37%	84.00%	N.A.	76.13%	71.13%	0.5893	N.A.	92.22%	93.14%	N.A.
	Purifier	60.50%	59.44%	51.50%	50.06%	51.56%	50.06%	50.53%	0.4851	N.A.	87.73%	50.95%	N.A.

TABLE 3: Results of PURIFIER against attackers with the more powerful shadow models.

Dataset	Defense	NSH (strong)	NSH (weak)	Mlleaks (strong)	Mlleaks (weak)
CIFAR10	None	58.46%	56.03%	70.20%	56.26%
	Purifer	51.08%	50.91%	50.57%	50.03%
Purchase100	None	88.27%	70.36%	68.50%	64.43%
	Purifer	52.62%	50.00%	50.00%	50.00%
FaceScrub530	None	70.30%	69.34%	73.46%	75.04%
	Purifer	50.46%	50.08%	49.96%	50.66%
CIFAR100	None	85.21%	76.98%	76.54%	73.78%
	Purifer	51.37%	50.91%	50.45%	50.19%
Texas	None	68.13%	66.37%	60.28%	58.93%
	Purifer	60.53%	50.92%	54.75%	46.38%
Location	None	82.81%	82.37%	86.56%	84.00%
	Purifer	52.00%	51.50%	50.00%	50.06%

TABLE 4: Attribute inference attack against the UTKFace classifier with and without PURIFIER.

Dataset	Defense	Utility		Attack Accuracy
		Train acc	Test acc.	
UTKFace	None	99.92%	83.08%	31.06%
	Purifier	83.87%	82.92%	21.07%

We report all the inversion error on CIFAR10, Purchase100, FaceScrub530 and CIFAR100 in Table 2. As shown in Table 2 and Figure 3, the inversion loss on the FaceScrub530 dataset is raised 4+ times (i.e. from 0.0114 to 0.0454) after applying PURIFIER, indicating that the performance reduction of the inversion attack is significant. Note that the effect of defense against the adversarial model inversion attacks on Purchase100 and CIFAR10 seems less significant compared with FaceScrub530. This is because the inversion attack does not perform well on these classifiers even without any defense.

5.2.3 *Effectiveness against Attribute Inference Attack*

We deploy PURIFIER under the attribute inference attack and find that PURIFIER is also effective in mitigating it. We train an attribute inference classifier on UTKFace dataset to predict the race of the given sample. Table 4 shows the results of our experiment. The attribute inference accuracy on the UTKFace dataset is reduced to 21.07% (almost random guessing) after applying PURIFIER.

5.3 Comparison with Other Defenses

We compare PURIFIER with the following defenses. ①MinMax [4]. ②MemGuard [15]. ③Model-Stacking [5]. ④MMD Defense [8]. ⑤SELENA [18]. ⑥Relax-Loss [30]. ⑦One-Hot Encoding. Moreover, we also test the defense performance of adding random noise to confidence scores.

Table 5 shows the defense performance of PURIFIER and other defense methods against membership inference attacks on different datasets. Among all the 10 attacks and the 6 datasets, PURIFIER achieve the best performance 31 times and the second best performance 12 times, compared to other defense methods (One-Hot Encoding and Random Noise are not included in the statistics because their transformation on confidence vectors leads to a large degree of semantic information loss). For the Mlleaks Attack, PURIFIER can achieve the best performance, similar to One-Hot Encoding and Random Noise. PURIFIER also achieves a better security-utility tradeoff than other defenses. It imposes a reduction in test accuracy of about 1%. In comparison, Model-Stacking and SELENA can mitigate membership inference attacks to some extent, but they incur an intolerable reduction in the model's test accuracy. MemGuard reaches acceptable defense performance with a negligible decrease in test accuracy, however, its defense performance is not as good as that of PURIFIER. For example, against NSH attack, the inference accuracy with MemGaurd will be at least 2.5% higher than that of PURIFIER.

For BlindMI attacks, PURIFIER has achieved the best performance on CIFAR10, Purchase100, CIFAR100, and Location. Specifically, on CIFAR100, PURIFIER reduces the attack accuracy to 49.87%, which is 10.29% better than other defense on average . The adavntage of PURIFIER is even more obvious on Location, on which PURIFIER was able to reduce the attack accuracy to 50.06%, while most of other defenses can only achieve a defense performance between 76.67% and 81.29%. On Facescrub530, PURIFIER's performance is second best only to RelaxLoss, with a negligible gap within 0.31%, where the attack accuracy of BlindMI can be reduced to less than 50.31%. The reason why PURIFIER can obtain such success against BlindMI is that among all the defenses mentioned above, only PURIFIER has been specially designed to defend against attacks targeted at statistical distribution like BlindMI.

It should be noted that the defense performance against transfer attack of PURIFIER on Texas and Location is abviously worse than that of SELENA, with average AUC of 0.4889 and 0.3117 respectively. One possible reason is that the scales of training sets that are used on Texas and Location are relatively small, which can bring some randomness to

TABLE 5: Defense performance of PURIFIER and other defense methods.

Dataset	Defense	Training acc.	Test acc.	NSH	Mlleaks	Adaptive	BlindMI	Gap	Transfer	Boundary	FP	Enhanced (D)	Inversion Error
CIFAR10	Purifier	95.93%	95.92%	50.91%	50.03%	50.00%	49.89%	50.01%	0.5010	0.4684	54.72%	50.12%	1.4939
	Min-Max	99.40%	94.38%	53.97%	52.93%	52.75%	53.52%	52.51%	0.5043	0.5011	55.78%	51.73%	1.4770
	MemGuard	99.99%	95.92%	53.63%	52.24%	52.07%	52.03%	52.04%	0.5043	0.5029	55.66%	51.65%	1.4439
	Model-Stacking	95.80%	92.12%	51.93%	51.01%	50.99%	52.69%	51.84%	0.5039	0.5008	55.03%	51.09%	1.4723
	MMD Defense	99.99%	87.44%	59.50%	57.60%	57.32%	53.92%	56.28%	0.5043	0.5437	56.81%	52.65%	1.4414
	SELENA	98.40%	93.90%	52.14%	52.35%	52.21%	51.08%	52.25%	0.5023	0.4896	54.56%	50.09%	1.4350
	Relax-Loss	99.29%	87.42%	50.07%	55.27%	58.94%	54.77%	55.94%	0.5035	0.5605	54.96%	50.23%	1.4413
	One-Hot Encoding	99.99%	95.92%	52.17%	50.00%	50.00%	51.88%	52.04%	0.5048	0.5214	54.73%	50.44%	1.4414
	Random Noise	99.99%	95.92%	55.97%	50.01%	50.00%	51.69%	52.04%	0.5048	0.5214	55.66%	50.47%	1.4342
Purchase100	Purifier	84.47%	84.36%	50.00%	50.00%	50.25%	49.48%	50.06%	0.4961	N.A.	75.06%	50.16%	0.1520
	Min-Max	99.89%	82.03%	65.13%	63.95%	64.06%	57.39%	58.93%	0.5411	N.A.	76.13%	58.92%	0.1428
	MemGuard	100.00%	84.36%	62.28%	57.86%	57.74%	61.35%	57.82%	0.5339	N.A.	76.24%	59.13%	0.1426
	Model-Stacking	81.84%	69.68%	61.16%	55.53%	55.28%	60.36%	56.08%	0.5287	N.A.	75.11%	52.47%	0.1472
	MMD Defense	100.00%	82.65%	69.48%	69.89%	69.13%	66.62%	58.67%	0.5423	N.A.	77.03%	62.32%	0.1439
	SELENA	83.24%	79.53%	51.90%	52.97%	52.84%	53.04%	51.83%	0.5434	N.A.	75.05%	50.13%	0.1440
	Relax-Loss	99.50%	82.17%	52.60%	61.31%	62.30%	57.84%	58.67%	0.5013	N.A.	75.03%	50.21%	0.1435
	One-Hot Encoding	100.00%	84.36%	57.65%	50.00%	50.00%	57.67%	57.82%	0.5431	N.A.	76.03%	53.37%	0.1524
	Random Noise	100.00%	84.36%	60.06%	50.02%	50.04%	54.44%	57.82%	0.5430	N.A.	77.07%	64.32%	0.1409
FaceScrub530	Purifier	77.51%	77.28%	50.08%	50.66%	50.12%	50.31%	50.12%	0.4990	0.5069	53.34%	50.84%	0.0454
	Min-Max	98.99%	68.31%	65.56%	69.84%	69.13%	61.16%	65.34%	0.5679	0.6430	57.25%	58.34%	0.0182
	MemGuard	100.00%	77.68%	62.48%	60.06%	59.64%	62.42%	61.16%	0.5228	0.6418	56.97%	58.03%	0.0117
	Model-Stacking	86.30%	57.05%	62.00%	51.86%	51.74%	60.62%	64.63%	0.5321	0.6379	54.92%	55.63%	0.0417
	MMD Defense	100.00%	77.38%	64.88%	67.95%	67.38%	63.55%	61.31%	0.5843	0.6783	60.13%	62.63%	0.0111
	SELENA	81.06%	72.05%	51.68%	51.23%	51.89%	54.05%	50.50%	0.5440	0.5844	53.01%	50.87%	0.0131
	Relax-Loss	99.52%	75.24%	51.90%	70.64%	70.75%	50.00%	62.14%	0.5002	0.7353	53.17%	50.87%	0.0109
	One-Hot Encoding	100.00%	77.68%	57.87%	50.00%	50.04%	61.23%	61.16%	0.5869	0.7739	54.73%	52.43%	0.0420
	Random Noise	100.00%	77.68%	56.85%	50.04%	50.04%	60.83%	61.16%	0.5869	0.7739	62.01%	63.73%	0.0175
CIFAR100	Purifier	66.35%	66.34%	50.91%	50.19%	50.45%	49.87%	50.01%	0.4938	0.3742	77.32%	50.72%	0.9354
	Min-Max	98.27%	68.36%	60.19%	61.98%	61.66%	60.15%	64.96%	0.5347	0.6329	80.53%	62.31%	0.9106
	MemGuard	100.00%	68.91%	58.17%	57.45%	56.96%	59.76%	65.53%	0.5769	0.5877	79.97%	64.32%	0.9123
	Model-Stacking	65.53%	60.04%	57.30%	59.11%	58.75%	58.82%	52.75%	0.5653	0.5623	78.13%	57.51%	0.9203
	MMD Defense	100.00%	67.50%	57.89%	57.22%	56.97%	65.31%	66.25%	0.5877	0.6532	79.68%	61.34%	0.9231
	SELENA	78.00%	62.10%	50.32%	50.42%	50.33%	57.59%	57.95%	0.4995	0.4982	77.16%	50.82%	0.9184
	Relax-Loss	99.37%	46.64%	53.14%	57.43%	60.81%	56.24%	76.37%	0.4990	0.7795	77.17%	50.87%	0.9266
	One-Hot Encoding	100.00%	69.10%	55.97%	50.00%	50.01%	59.42%	65.01%	0.5980	0.6668	78.93%	56.71%	0.9304
	Random Noise	100.00%	69.10%	56.48%	50.03%	50.03%	60.16%	65.01%	0.5980	0.6668	76.37%	57.92%	0.9103
Texas	Purifier	50.99%	47.88%	50.92%	46.38%	54.48%	52.35%	51.56%	0.4926	N.A.	53.34%	50.84%	N.A.
	Min-Max	75.51%	49.05%	54.12%	57.03%	56.89%	53.77%	62.23%	0.5328	N.A.	69.92%	64.57%	N.A.
	MemGuard	75.04%	49.88%	55.80%	54.99%	56.75%	53.09%	62.58%	0.5287	N.A.	68.04%	64.02%	N.A.
	Model-Stacking	53.18%	47.02%	51.20%	50.08%	56.93%	52.86%	53.08%	0.5204	N.A.	62.90%	55.40%	N.A.
	MMD Defense	77.32%	50.01%	56.96%	58.39%	59.73%	67.40%	63.66%	0.5419	N.A.	63.03%	64.26%	N.A.
	SELENA	77.90%	55.25%	54.40%	51.00%	57.79%	58.04%	61.33%	0.3222	N.A.	59.42%	50.72%	N.A.
	Relax-Loss	85.50%	48.19%	53.05%	58.48%	59.19%	47.35%	68.66%	0.5009	N.A.	59.27%	50.16%	N.A.
	One-Hot Encoding	78.81%	49.88%	53.89%	50.00%	60.73%	60.89%	64.13%	0.5431	N.A.	62.23%	53.93%	N.A.
	Random Noise	78.80%	49.88%	53.05%	50.01%	58.73%	60.00%	64.13%	0.5430	N.A.	62.60%	54.08%	N.A.
Location	Purifier	60.50%	59.44%	51.50%	50.06%	51.56%	50.06%	50.53%	0.4851	N.A.	87.73%	50.95%	N.A.
	Min-Max	99.85%	56.49%	59.52%	58.38%	59.71%	65.08%	71.68%	0.5327	N.A.	90.40%	63.28%	N.A.
	MemGuard	100.00%	58.44%	61.21%	58.28%	59.13%	65.22%	70.78%	0.6038	N.A.	89.89%	60.34%	N.A.
	Model-Stacking	70.29%	57.82%	53.24%	51.88%	52.66%	67.41%	56.24%	0.5431	N.A.	89.22%	55.20%	N.A.
	MMD Defense	100.00%	59.96%	63.25%	54.77%	56.32%	71.29%	70.02%	0.5912	N.A.	89.13%	60.21%	N.A.
	SELENA	77.90%	55.25%	54.40%	51.00%	52.23%	65.86%	61.33%	0.3012	N.A.	87.56%	50.15%	N.A.
	Relax-Loss	99.50%	58.94%	58.56%	69.94%	70.00%	77.94%	70.28%	0.4974	N.A.	87.41%	50.43%	N.A.
	One-Hot Encoding	100.00%	58.44%	54.26%	50.00%	52.14%	66.67%	69.78%	0.5893	N.A.	88.42%	55.17%	N.A.
	Random Noise	100.00%	58.44%	55.20%	50.08%	51.97%	66.02%	69.78%	0.5893	N.A.	88.40%	55.21%	N.A.

TABLE 6: Attribute inference attack against the UTKFace classifier with different defense methods.

Dataset	Defense	Utility		Attack Accuracy
		Train acc	Test acc.	
UTKFace	None	99.92%	83.08%	31.06%
	Purifier	83.87%	83.97%	21.07%
	Min-Max	99.87%	82.92%	27.64%
	MemGaurd	99.83%	83.89%	27.23%
	Model-Stacking	82.74%	80.32%	28.04%
	MMD Defense	98.84%	84.72%	29.37%
	SELENA	88.23%	81.62%	24.33%
	Relax-Loss	99.47%	79.49%	23.16%

the experiment results, without defense model on stable convergence. More convincible results are still remained to be tested on larger training sets.

We can also see that PURIFIER performs better than most of other defense methods agianst FP attack. Although SELENA performs better on 5 datasets, it outperforms PURIFIER by less than 0.3%. However, along with other defense methods, PURIFIER does not perform well in defending FP attack on some datasets, including Purchase100, CIFAR100 and location. We believe that this is due to the fact that FP attack trains a large number of shadow models, thus it can obtain a lot of information about member distribution. In the training process of PURIFIER, the loss function we set must take into account the differences between members and non members, as well as the classification accuracy of the model. Therefore, the trained CVAE is not the optimal solution to reduce the differences between members and non members. As a result, the confidence scores processed by the confidence reformer can still be distinguished by FP attack.

PURIFIER also achieves the best performance in defending model inversion attack on CIFAR10 and Facescrub530. Table 5 shows that PURIFIER has the largest inversion error(also called reconstruction error) compared with other defenses on these datasets, quantitatively demonstrating that PURIFIER achieves better defense performance against adversarial model inversion attack than other defenses. Figure 3 depicts the reconstructed samples from confidence vectors given by each defense model on FaceScrub530 dataset. With PURIFIER as defense, the reconstructed images are much less similar to the ground truth image and look more blurred. Other defense methods, however, could hardly protect the model from adversaries recovering details of the original image. The results can be quantitatively verified by the similarity scores gathered from the Microsoft Azure Face Recognition service. For instance, the average similarity scores of reconstructed images of MemGuard-defended models are 0.17, which are larger than that of PURIFIER (i.e., 0.14). PURIFIER achieves the smallest similarity scores among other defense methods, indicating that PURIFIER is most effective in protecting the target model against adversarial model inversion attack.

Moreover, we also perform attribute inference attack as section 5.2.3 goes on these defense methods. Table 6 shows the results. We can see that although the classification accuracy on the training dataset of PURIFIER is the lowest, the accuracy on the testing dataset doesn't decrease, which means that PURIFIER maintains the function of the model. Against attribute inference attack, PURIFIER has the lowest attack accuracy, 2.09% lower than Relax-Loss, which has the second best defense performance.

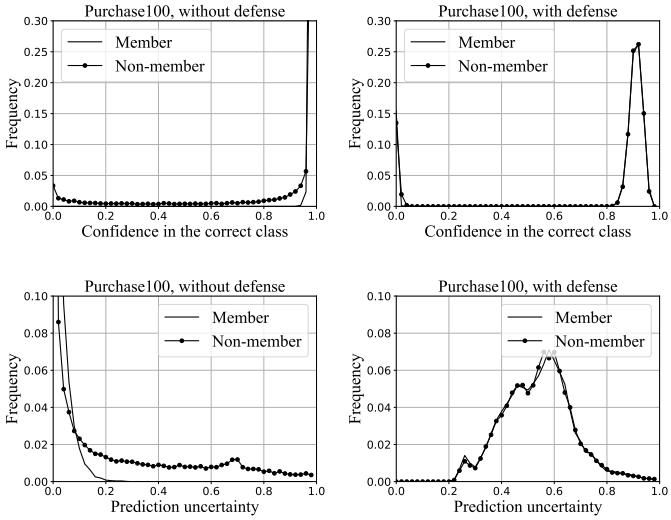


Fig. 5: Distribution of the target classifier's confidence in predicting the correct class and the prediction uncertainty on members and non-members of training set.

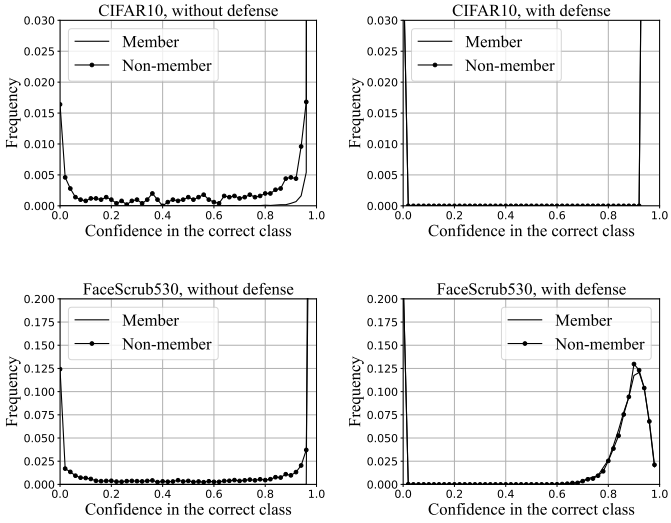


Fig. 6: Distribution of the target classifier's confidence in predicting the correct class on members and non-members of its training set.

5.4 Indistinguishabilities in Purified Scores

In this subsection, we design further specific experiments to analyze how the purified confidence scores affect membership inference attacks by evaluating three indistinguishabilities: individual shape, statistical distribution and prediction label.

5.4.1 *Individual Indistinguishability*

As designed, PURIFIER reshapes the input confidence score vectors according to the pattern of the learned non-member samples. To examine the indistinguishability of the confidence scores on members and non-members, we plot the histogram of the target classifier's confidence in predicting the correct class and the prediction uncertainty in Figure 5. The prediction uncertainty is measured as the normalized entropy $\frac{-1}{\log(k)} \sum_i \hat{\vec{y}}_i \log(\hat{\vec{y}}_i)$ of the confidence vector $\vec{y} = F(\vec{x})$, where k is the number of classes. As Figure 5 shows, PURIFIER

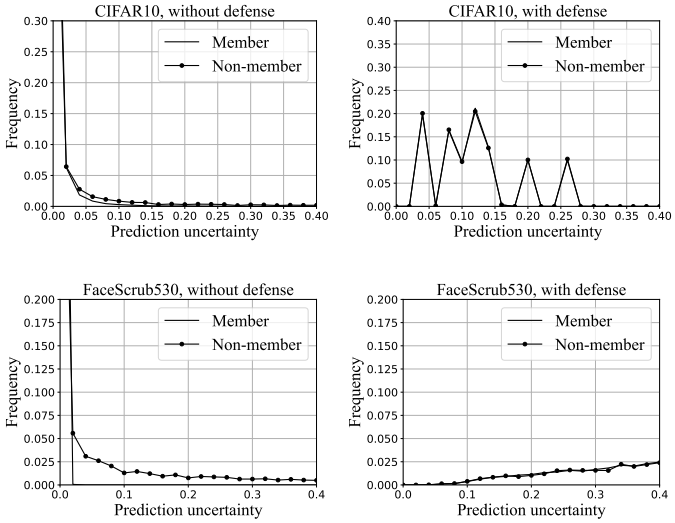
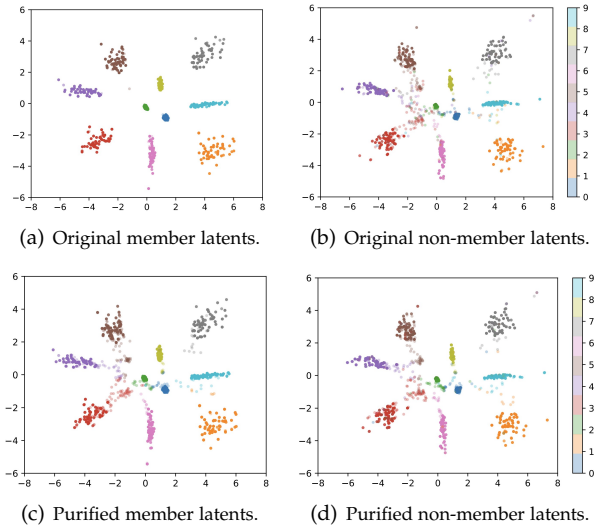


Fig. 7: Distribution of the target classifier's prediction uncertainty on members and non-members of its training set. The uncertainty is measured as the normalized entropy of the confidence score vector.



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

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

Fig. 8: 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.

TABLE 7: 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	0.103	0.004	0.412	0.016	0.415	0.017
	Purifier	0.009	0.000	0.019	0.001	0.012	0.001
Uncer	None	0.114	0.005	0.201	0.015	0.418	0.017
	Purifier	0.006	0.000	0.007	0.001	0.006	0.001

can reduce the gap between the two curves representing members and non-members respectively. Similar results can be obtained on CIFAR10 in Figure 6, Figure 7 and on FaceScrub530 in classifiers.

We also report the maximum gap and the average gap between the curves in Table 7. The results show that our approach can significantly reduce both the maximum and average gaps between the target classifier's confidence in predicting the correct class as well as the prediction uncertainty on its members versus non-members. This demonstrates that PURIFIER successfully reduces the individual differences between members and non-members.

5.4.2 *Statistical Indistinguishability*

We present the statistical distribution of confidence score vectors in the encoder latent space of the *confidence reformer*. Figure 8 visibly displays the distributional differences between members and non-members in the latent space on CIFAR10. As illustrated in the first row, latent vectors of the members tend to cluster together according to their labels, while those of non-members are more scattered in the map. The second row of Figure 8 also shows the statistical distribution of members and non-members processed with PURIFIER in the latent space. When processed with PURIFIER, Gaussian noises are added to make the clustered member latent vectors to be more scattered on the latent space. This demonstrates that PURIFIER can reduce the statistical differences between members and non-members while preserving semantic utility.

5.4.3 *Label Indistinguishability in Purified Scores*

PURIFIER uses *label swapper* to identify and swap the prediction label of members. *label swapper* incurs negligible reduction of test accuracy, while swapping the labels of the member samples reduces the training accuracy to a greater extent. It causes that the gap between the accuracy of members and non-members which is often exploited to perform *label-only attack* is minimized. This is shown in Table 2, where the training accuracy of the model is close to the test accuracy. Many label-only membership inference attacks are less effective under PURIFIER with *label swapper*. This reflects that the purified member confidence vectors are less distinguishable from those of the non-members in terms of labels.

We conduct the ablation study on the label swapper to investigate its effectiveness in our defense mechanism. As the Table 8 shows, we defend the classifiers against label-only transfer attack in the case of no defense, defense with only confidence reformer, defense with confidence reformer and label swapper. Results show a significant AUC drop with the help of label swapper compared to those mechanisms without it. Additionally, the result indicates that label-only transfer attack can obtain approximately the same AUC among models with no defense and PURIFIER without the label swapper, which means that PURIFIER without label swapper fails to mitigate the label-only attack as we have assumed. In summary, the label swapper plays an indelible role in defending the label-only attacks.

5.5 Efficiency of PURIFIER

Figure 9 presents the efficiency of PURIFIER compared with other defenses. We perform our experiments on a PC equipped with four Titan XP GPUs with 12GBytes of graphic

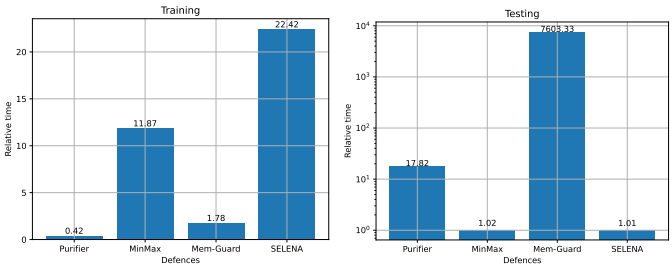


Fig. 9: Efficiency of different defense methods.

TABLE 8: Ablation study on the Label Swapper. G represents the confidence reformer, and H represents the label swapper.

Dataset	Defense	Label Only Transfer Attack
CIFAR10	None	0.5048
	G	0.5045
	H + G	0.5010
CIFAR100	None	0.6668
	G	0.6632
	H + G	0.4938

memory,128 GBytes of memory and an Intel Xeon E5-2678 CPU. The training time of PURIFIER is only 0.423 time of the target classifier, which outperforms Min-Max, MemGaurd and SELENA. The testing time of PURIFIER is 18.06 times as much as the target classifier, which is considered acceptable compared to MemGuard whose testing time is 7,000+ times more than the original classifier.

5.6 What If Swapping Confidence Reformer and Label Swapper

The placement order of *confidence reformer* and *label swapper* is also worth discussing. As we said in section 4, the confidence reformer is designed to achieve individual and statistical indistinguishability, and the label swapper is designed to achieve label indistinguishability. The independence of their functions also indicates that we can place label swapper before confidence reformer, or reverse it.

To discuss the placement order of confidence reformer and label swapper, we test the performance of PURIFIER with label swapper after confidence reformer and PURIFIER with label swapper before confidence reformer respectively against six different attacks, including NSH, Mlleaks, Adaptive, Gap, Transfer and Boundary attacks, on six datasets.

Table 9 shows that, in most cases, PURIFIER with label swapper before confidence reformer outperforms the other one with an average 0.19% margin. One reasonable explanation is that with label swapper before confidence reformer, the intention of swapping the output label to the one with the second-largest confidence score is more likely to be leavened since the swapping result will then go through the confidence reformer for further process. In other words, with confidence reformer placed after label swapper, more randomness is added to the output result, making label swapping less straightforward to detect.

To further unravel the insightful reason why placing label swapper after confidence reformer could cause worse distinguishability, we conduct another experiment. Theoretically, the transformation effect of confidence reformer appears to be softening. If we swap labels later, the softening effect may be destroyed, which can lead to a difference between members and non-members. For example, assume that after processed by confidence reformer, the confidence score of some class is always not higher than 0.95. If it is found that the confidence score of this class is 98%, then it is obvious that the sample has been swapped, indicating that it is a member of training dataset. Aiming to take advantage of this weakness, we design an adaptive attack on CIFAR10 which tries to capture the outputs that have been switched to the second-most confident class, using a shadow model.

The result shows that when placing the label swapper after confidence reformer, 10.97% of the outputs are found having switched labels, while none of them can be detected in the case where label swapper is placed before confidence reformer. With confidence reformer as a secondary treatment, the outputs of label swapper have a greater chance to be modified so as to become unrecognizable and thus escape the detection of adversaries, which further demonstrates the rationality of placing the label swapper before confidence reformer.

5.7 Further Experiments

5.7.1 Influence of Different Training Datasets

We also investigate the effect of the PURIFIER's training data by using data with different sizes or distributions 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 CIFAR10 data to train the PURIFIER for the FaceScrub530 classifier, and use randomly generated data to train the PURIFIER for the Purchase100 classifier.

We present the effect of the in-distribution training data in Table 10. The results show that PURIFIER is still effective. The membership inference accuracy (Mlleaks and Adaptive) is reduced to nearly 50% regardless of the size of D_2. PURIFIER is also insensitive 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 good for the defender, as one can achieve good performance with a small reference set. However, when the size of D_2 becomes too large (i.e., 40,000 to 60,000), the classification accuracy drops to a certain extent. The reason could be that PURIFIER starts to learn the detailed information of the confidence score vectors. As a result, the purified confidence score vector no longer concentrates on general patterns but becomes an accurate reconstruction, which hinders the classification utility.

When we use the classifier's training data D_1 to train the PURIFIER, the defense performance is comparable to the ones on D_2. For example, the attack accuracy of the NSH attack is 52.29%, which is marginally higher than the result of 51.71% on D_2, but still acceptable.

Table 11 shows the effect of the out-of-distribution training data. PURIFIER can still mitigate the attacks, but at the cost of sacrificing the utility of the target classifier significantly. This is not surprising because PURIFIER cannot extract useful patterns from the confidence scores on out-of-distribution data, which makes the purified confidence information meaningless.

TABLE 9: Defense performance of different placement order of Confidence Reformer and Label Swapper. P1 refers to placing Confidence Reformer before Label Swapper, and P2 reversely.

Dataset	Order	Train acc.	Test acc.	NSH	Mlleaks	Adaptive	Gap	Transfer	Boundary
CIFAR10	P1	97.60%	95.52%	51.65%	50.26%	50.23%	50.84%	0.4974	0.4949
	P2	95.93%	95.92%	50.91%	50.03%	50.00%	50.01%	0.5010	0.4684
Purchase100	P1	86.59%	82.23%	51.71%	50.09%	50.13%	51.68%	0.4978	N.A.
	P2	84.47%	84.36%	50.00%	50.00%	50.25%	50.06%	0.4961	N.A.
Facescrub530	P1	77.58%	77.52%	51.56%	51.04%	50.00%	50.02%	0.4983	0.6185
	P2	77.51%	77.28%	50.08%	50.66%	50.12%	50.12%	0.4990	0.5069
CIFAR100	P1	70.02%	69.98%	50.01%	50.15%	51.02%	50.02%	0.5120	0.4975
	P2	66.35%	66.34%	50.91%	50.19%	50.45%	50.01%	0.4938	0.3742
Texas	P1	51.01%	50.91%	51.29%	50.00%	51.18%	50.05%	0.5028	N.A.
	P2	50.99%	47.88%	50.92%	46.38%	54.48%	51.56%	0.4926	N.A.
Location	P1	60.45%	60.43%	51.75%	50.31%	51.41%	50.01%	0.5015	N.A.
	P2	60.50%	59.44%	51.50%	50.06%	51.56%	50.53%	0.4851	N.A.

TABLE 10: Effect of the PURIFIER's in-distribution training data on the defense performance. The numbers are reported on the Purchase100 dataset.

Training set	Test acc	NSH	Mlleaks	Adaptive
D_2 (5,000)	84.47%	52.35%	50.08%	50.62%
D_2 (10,000)	84.45%	51.14%	50.13%	50.12%
D_2 (20,000)	84.36%	51.67%	50.10%	50.11%
D_2 (40,000)	83.89%	52.01%	50.11%	50.18%
D_2 (60,000)	84.64%	52.02%	50.11%	50.09%
D_1 (20,000)	84.66%	52.08%	50.11%	50.08%

TABLE 11: Effect of the PURIFIER's out-of-distribution training data on the defense performance.

Classifier	Purifier	Test acc	NSH	Mlleaks	Adaptive
FaceScrub530	CIFAR10	41.33%	53.29%	50.11%	50.48%
Purchase100	Random	6.91%	51.31%	50.17%	50.82%

5.7.2 *Inference about Reference Data*

Our approach uses a reference dataset to train PURIFIER. Involving an in-distribution reference dataset in the defense mechanism is common in the literature. For instance, MemGuard uses a reference set to train the defense classifier. Min-Max uses it to train the inference model. Unfortunately, little has been discussed on whether such reference dataset brings vulnerability for data inference attacks. Assuming the reference data are considered as members, we present the inversion error and the inference accuracy (we consider NSH attack) on the reference set D_2 and the test set D_3 for each defense in Table 12. Results show that the inference accuracy does not increase on the reference set compared with the original training data of the target classifier. PURIFIER can still preserve the defense effect against the adversarial model inversion attack and the membership inference attack.

5.7.3 *Effect of* PURIFIER *on Noisy Member Detection*

Furthermore, we investigate the effectiveness of PURIFIER to detect noisy members. Label swapper needs to judge whether an input sample is a member, which should be robust to noise. We conduct this experiment with the purpose of testing whether the PURIFIER can handle noisy samples well. More specifically, when a member with noise is input, the PURIFIER can process it as a member. We use the adversarial attack methods (FGSM [35]) to create noisy members.

TABLE 12: Results of model inversion attack and membership inference attack on the reference set for different defenses. The experiments are performed on the FaceScrub530 dataset.

Defense	Inversion error	Inference accuracy
Purifier	0.0422	51.44%
Min-Max	0.0219	52.19%
MemGuard	0.0129	52.51%

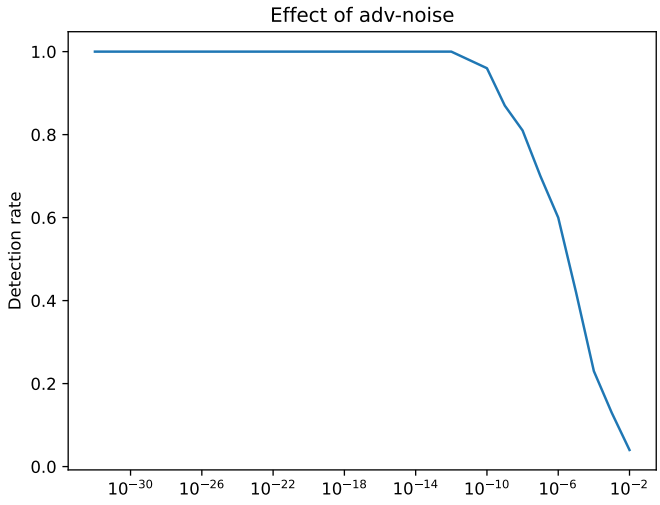


Fig. 10: The proportion of noise data that PURIFIER can detect under the FGSM attacks on the FaceScrub530 dataset.

As shown in Figure 10, PURIFIER can accurately detect the members with noise $\|\eta\|_\infty < 1e{-}10$ on FaceScrub530 dataset, which means that PURIFIER is robust to noisy members.

6 RELATED WORKS

Inference Attacks. The inference attacks against machine learning can be divided into model inference and data inference attacks. In model inference attacks [22], [20], [23], [21], an attacker could infer the parameters [22], hyperparameters [23], architecture [20] and functionality [21] of a target model.

We focus on data inference attacks in this paper. Xiao et al. [36] studied the adversarial reconstruction problem

where they aimed to prevent the latent representations from being decoded into the original input data. To this end, they regularized the encoder with an adversarial loss from a decoder. They studied 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 did not consider the adversarial scenario where the attacker has no access to the same data distribution as the original training data. Jia and Gong [37] proposed the adversarial formulation for privacy protection. They aimed 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. Membership inference attack is performed to determine whether a given data sample is part of a target dataset. Homer et al. [38] proposed one of the first membership inference attacks in the biomedical setting on genomic data. Some studies also performed membership inference attacks on other biomedical data such as MicroRNA [39] and DNA methylation [40]. Pyrgelis et al. [41], [42] further showed that it is possible to perform membership inference attack on location datasets as well. Shokri et al. [1] performed membership inference attack in the machine learning setting which is the same with our work.

Secure & Privacy-Preserving Machine Learning. A number of studies made 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 [43], [44], [45], [46], [47], [48] and secure computing using trusted hardware [49], [50]. Although these methods protect sensitive data from direct observation by the attacker, they do not prevent information leakage via the model computation itself which could be exploited by various inference attacks.

7 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 and efficient 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 *2017 IEEE Symposium on Security and Privacy, SP 2017, San Jose, CA, USA, May 22-26, 2017*, 2017, pp. 3–18.

[2] M. Fredrikson, S. Jha, and T. Ristenpart, "Model inversion attacks that exploit confidence information and basic countermeasures," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, Denver, CO, USA, October 12-6, 2015*, 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 *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, vol. 1, 2017, pp. 603–618.

[4] M. Nasr, R. Shokri, and A. Houmansadr, "Machine learning with membership privacy using adversarial regularization," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '18. ACM, 2018, pp. 634–646, event-place: Toronto, Canada.

[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 *Proceedings of the 26th Annual Network and Distributed System Security Symposium (NDSS 2019)*, 2018.

[6] S. Yeom, I. Giacomelli, M. Fredrikson, and S. Jha, "Privacy risk in machine learning: Analyzing the connection to overfitting," pp. 268–282, 2018.

[7] Z. Li and Y. Zhang, "Membership leakage in label-only exposures," in *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, 2021, pp. 880–895.

[8] J. Li, N. Li, and B. Ribeiro, "Membership inference attacks and defenses in classification models," in *Proceedings of the Eleventh ACM Conference on Data and Application Security and Privacy*, ser. CODASPY '21. Association for Computing Machinery, 2021, pp. 5–16. [Online]. Available: https://doi.org/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," *arXiv preprint arXiv:2101.01341*, 2021.

[10] N. Carlini, S. Chien, M. Nasr, S. Song, A. Terzis, and F. Tramer, "Membership inference attacks from first principles," in *2022 IEEE Symposium on Security and Privacy (SP)*. IEEE Computer Society, 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," 2022.

[12] C. Song and V. Shmatikov, "Overlearning reveals sensitive attributes," in *8th International Conference on Learning Representations, ICLR 2020*, 2020.

[13] Z. Yang, J. Zhang, E.-C. Chang, and Z. Liang, "Neural network inversion in adversarial setting via background knowledge alignment," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '19. Association for Computing Machinery, 2019, pp. 225–240, event-place: New York, NY, USA.

[14] M. Abadi, A. Chu, I. J. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, October 24-28, 2016*, 2016, pp. 308–318.

[15] J. Jia, A. Salem, M. Backes, Y. Zhang, and N. Z. Gong, "MemGuard: Defending against black-box membership inference attacks via adversarial examples," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security - CCS '19*. ACM Press, 2019, pp. 259–274, event-place: London, United Kingdom.

[16] X. Tang, S. Mahloujifar, L. Song, V. Shejwalkar, M. Nasr, A. Houmansadr, and P. Mittal, "Mitigating membership inference attacks by self-distillation through a novel ensemble architecture," 2021.

[17] C. A. Choquette-Choo, F. Tramer, N. Carlini, and N. Papernot, "Label-only membership inference attacks," in *Proceedings of the 38th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, M. Meila and T. Zhang, Eds., vol. 139. PMLR, 18–24 Jul 2021, pp. 1964–1974. [Online]. Available: https://proceedings.mlr.press/v139/choquette-choo21a.html

[18] X. Tang, S. Mahloujifar, L. Song, V. Shejwalkar, M. Nasr, A. Houmansadr, and P. Mittal, "Mitigating membership inference attacks by Self-Distillation through a novel ensemble architecture," in *31st USENIX Security Symposium (USENIX Security 22)*. Boston, MA: USENIX Association, Aug. 2022, pp. 1433–1450. [Online]. Available: https://www.usenix.org/conference/usenixsecurity22/presentation/tang

[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," pp. 17–32, 2014.

[20] S. J. Oh, M. Augustin, M. Fritz, and B. Schiele, "Towards reverse-engineering black-box neural networks," in *International Conference on Learning Representations*, 2018.

[21] T. Orekondy, B. Schiele, and M. Fritz, "Knockoff nets: Stealing functionality of black-box models," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4949–4958.

[22] F. Tramèr, F. Zhang, A. Juels, M. K. Reiter, and T. Ristenpart, "Stealing machine learning models via prediction APIs," in *25th USENIEX Security Symposium, USENIX Security 16, Austin, TX, USA, August 10-12, 2016.*, pp. 601–618.

[23] B. Wang and N. Z. Gong, "Stealing hyperparameters in machine learning," in *2018 IEEE Symposium on Security and Privacy (SP)*, 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," 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 *Proceedings of the 34th Annual Computer Security Applications Conference*, ser. ACSAC '18. Association for Computing Machinery, pp. 393–406, event-place: San Juan, PR, USA.

[26] X. Wu, M. Fredrikson, S. Jha, and J. F. Naughton, "A methodology for formalizing model-inversion attacks," in *2016 IEEE 29th Computer Security Foundations Symposium (CSF)*, 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 *2017 15th Annual Conference on Privacy, Security and Trust (PST)*.

[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."

[29] J. Li, N. Li, and B. Ribeiro, "Membership inference attacks and defenses in classification models," in *Proceedings of the Eleventh ACM Conference on Data and Application Security and Privacy*, 2021, pp. 5–16.

[30] D. Chen, N. Yu, and M. Fritz, "Relaxloss: Defending membership inference attacks without losing utility," 2022.

[31] Z. Zhang, Y. Song, and H. Qi, "Age progression/regression by conditional adversarial autoencoder," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 5810–5818.

[32] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

[33] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 2261–2269.

[34] M. Azure, "https://azure.microsoft.com/en-us/services/cognitive-services/face/," 2022.

[35] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014.

[36] T. Xiao, Y.-H. Tsai, K. Sohn, M. Chandraker, and M.-H. Yang, "Adversarial learning of privacy-preserving and task-oriented representations," 2019.

[37] J. Jia and N. Z. Gong, "AttriGuard: A practical defense against attribute inference attacks via adversarial machine learning," in *27th USENIX Security Symposium (USENIX Security 18)*. {USENIX} Association, 2018, pp. 513–529, event-place: Baltimore, MD.

[38] N. Homer, S. Szelinger, M. Redman, D. Duggan, W. Tembe, J. Muehling, J. V. Pearson, D. A. Stephan, S. F. Nelson, and D. W. Craig, "Resolving individuals contributing trace amounts of DNA to highly complex mixtures using high-density SNP genotyping microarrays," vol. 4, no. 8, p. e1000167.

[39] M. Backes, P. Berrang, M. Humbert, and P. Manoharan, "Membership privacy in MicroRNA-based studies," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '16. Association for Computing Machinery, pp. 319–330, event-place: Vienna, Austria.

[40] I. Hagestedt, Y. Zhang, M. Humbert, P. Berrang, H. Tang, X. Wang, and M. Backes, "MBeacon: Privacy-preserving beacons for DNA methylation data," in *Proceedings 2019 Network and Distributed System Security Symposium*. Internet Society, event-place: San Diego, CA.

[41] A. Pyrgelis, C. Troncoso, and E. D. Cristofaro, "Knock knock, who's there? membership inference on aggregate location data," in *Proceedings 2018 Network and Distributed System Security Symposium*. Internet Society, event-place: San Diego, CA.

[42] A. Pyrgelis, C. Troncoso, and E. De Cristofaro, "Under the hood of membership inference attacks on aggregate location time-series."

[43] J. Liu, M. Juuti, Y. Lu, and N. Asokan, "Oblivious neural network predictions via MiniONN transformations," no. 2017, pp. 619–631, 2017.

[44] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical secure aggregation for privacy-preserving machine learning," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security - CCS '17*, 2017, pp. 1175–1191.

[45] L. T. Phong, Y. Aono, T. Hayashi, L. Wang, and S. Moriai, "Privacy-preserving deep learning via additively homomorphic encryption," vol. 13, no. 5, pp. 1333–1345, 2018.

[46] 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 *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ser. ICML'16. JMLR.org, 2016, pp. 201–210, event-place: New York, NY, USA.

[47] P. Mohassel and Y. Zhang, "SecureML: A system for scalable privacy-preserving machine learning," in *2017 IEEE Symposium on Security and Privacy, SP 2017, San Jose, CA, USA, May 22-26, 2017*, 2017, pp. 19–38.

[48] C. Dwork and V. Feldman, "Privacy-preserving prediction," in *Proceedings of the 31st Conference On Learning Theory*, ser. Proceedings of Machine Learning Research, S. Bubeck, V. Perchet, and P. Rigollet, Eds., vol. 75. PMLR, 2018, pp. 1693–1702.

[49] O. Ohrimenko, F. Schuster, C. Fournet, A. Mehta, S. Nowozin, K. Vaswani, and M. Costa, "Oblivious multi-party machine learning on trusted processors," in *25th USENIX Security Symposium, USENIX Security 16, Austin, TX, USA, August 10-12, 2016.*, 2016, pp. 619–636.

[50] C. Juvekar, V. Vaikuntanathan, and A. Chandrakasan, "Gazelle: A low latency framework for secure neural network inference," in *27th USENIX Security Symposium (USENIX Security 18)*, 2018.