Purifier: Defending Data Inference Attacks via Transforming Confidence Scores

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 score vectors 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. We set up experiments on a large number of widely adopted datasets. 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 has been widely adopted in various applications, ranging from face recognition to intelligent medicine. While providing convenience for people's life, machine learning also takes use of a lot of private information, leaving hidden danger for data leakage.

Usually, users have access to the APIs from service providers, which return a confidence score vector or a label by the models. Many studies indicate that the predicted information can be used for *data inference attack*, which aims at inferring secret information about the data that the machine learning model operates on.

Data inference attack could be largely divided into three categories. The first is *membership inference attack*, aiming at inferring whether a sample is a member of the model's training dataset. The second is *model inversion attack*, aiming at inferring the input data from the output data of the model. The third is *attribute inference attack*, aiming at inferring the hidden information of a sample. We focus out study on membership inference attack.

The major cause of membership inference attack is that the prediction results are distinguishable for members and non-members. For example, when a model overfits on the training data, it behaves more confidently on predicting the training data (members) than predicting the testing data (non-members). Many studies take use of the prediction differences between members and non-members to perform membership inference attack. In general, the prediction differences exist in the following 3 aspects.

1) *Individual shape*. The confidence scores on members and non-members differ in their individual shape. 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 [1], [2]
2) *Statistical distribution*. Confidence scores on 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 [3] exploits such statistical difference to infer membership by comparing the distance variation of the confidence scores of two generated datasets.
3) *Prediction label*. The confidence scores on members and non-members are different in the prediction label. Member samples have a higher probability of being correctly predicted than the non-member samples, which leads to the difference in classification accuracy. Various label-only attacks exploit such distinguishability [4], [5], [6].

In the paper, we propose a method, namely PURIFIER, to defend membership inference attack. It takes the confidence scores predicted by the target model as input, and output transformed confidence scores, which behave indistinguishable in *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 model named *confidence reformer*. The confidence reformer is trained on non-members' confidence scores predicted by the target model, which enables the model to learn individual shape of non-members data. Transformed by confidence reformer, the members' confidence scores are indistinguishable from the non-members'.
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

- *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.

confidence scores, thus members and non-members become indistinguishable in statistical distribution.

3) To purify prediction label, we propose a mechanism named *label swapper*. Attack by prediction label is known as *label-only attack*, which takes use of classification accuracy gap between members and non-members. The label swapper modifies members' prediction label to second confident label at a specially designed rate. In consideration of efficiency, we introduce KNN into label swapper.

In our conference version, confidence scores are processed by confidence reformer first, and then label swapper. But according to our further research, we put label swapper before confidence reformer, which makes the PURIFIER more reasonable.

Experiments shows that PURIFIER has the ability to defend various data inference attacks. We set up experiments on various widely used datasets, including cifar10, cifar100, facescrub530, UTKFace, purchase100, texas and location, and we use PURIFIER to defend a number of membership inference attacks. It turns out that the PURIFIER has a good defense effect. For example, the inference accuracy of NSH attack on cifar100 decrease from 76.98% to 50.01%, and the inference accuracy of Mlleak attack on cifar100 decrease from 73.78% to 50.15%. Meanwhile, although PURIFIER is designed to defend the membership inference attacks, it turns out to be also effective in defending the model inversion attack and the attribute inference attack.

We also set up comparative experiments on other defense methods, which shows that PURIFIER behave effective and efficient in most case. For example, while SELENA performs 50.32% inference accuracy on NSH attack, PURIFIER decreases the accuracy to 50.01%. And the training time of PURIFIER is only 0.423 time of target model, while SELENA needs 22.42 time.

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 3 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 achieve indistinguishability between members and non-members in the above 3 aspects.
3) On the basis of our great many of experiments, PURIFIER outperforms other defense methods in both effectiveness and efficiency.

2 RELATED WORK

2.1 Membership inference attack

Membership inference attack is first proposed by Shokri in his paper. By training shadow models, attackers can judge whether a sample is in the training data set or not. Up to this time, various effective membership inference methods are proposed.
NSH attack.
Mlleaks attack.
Adaptive attack.
BlindMI attack.
Label-only attack.
First principle attack.
Enhanced attack.

2.2 Membership inference defense

A number of membership inference defense methods are proposed.
Min-Max.
MemGaurd.
Model Stacking.
MMD Defense.
SELENA.
One-Hot Encoding.

3 PROBLEM STATEMENT

We focus our study on classification models of neural network. More specifically, machine learning classifier F is trained on the training dataset D_{train}, and tested on the test dataset D_{test}. Each element in D_{train} and D_{test} is a sample and its corresponding class. The classifier F maps a given input x to a confidence scores vector $F(x)$, and the corresponding class of the maximal score in $F(x)$ is the prediction label.

We assume the attacker has a black access to the classifier F, where the attacker can only input x and get the prediction result $F(x)$. Also, the attacker can collect data samples which has a similar distribution with the classifier's D_{train} and D_{test} to help his attack.

We conduct our defense research with the purpose of not affecting classifier's structure. This means the defender neither need to adjust the model's layers, nor retrain the classifier. However, compared with attackers, we defender has access to the training dataset D_{train}.

4 APPROACH: PURIFIER

We propose PURIFIER as a defense of data inference attacks. The main idea is to transform the confidence score vector $F(x)$ that they appear indistinguishable on members and non-members. PURIFIER consists of a *label swapper* H and a *confidence reformer* G, as shown in Figure 1.

4.1 Focus of PURIFIER

We focus on reducing the three distinguishabilities of confidence scores between members and non-members: *individual shape*, *statistical distribution* and *prediction label*.
Individual shape. Distinguishability of individual shape means a member sample and a non-member sample differ in the shape of their confidence scores vector. For example, the classifier usually predicts the class of member data with a high degree of confidence, which means the value of the maximal scores in the vector would be larger for member sample than non-member sample. This can be proved by Figure 2 (a), where we calculate the frequency of maximal score's confidence of members and non-members respectively. It can be seen that non-members are more than members at low confidence, reversely at high confidence.

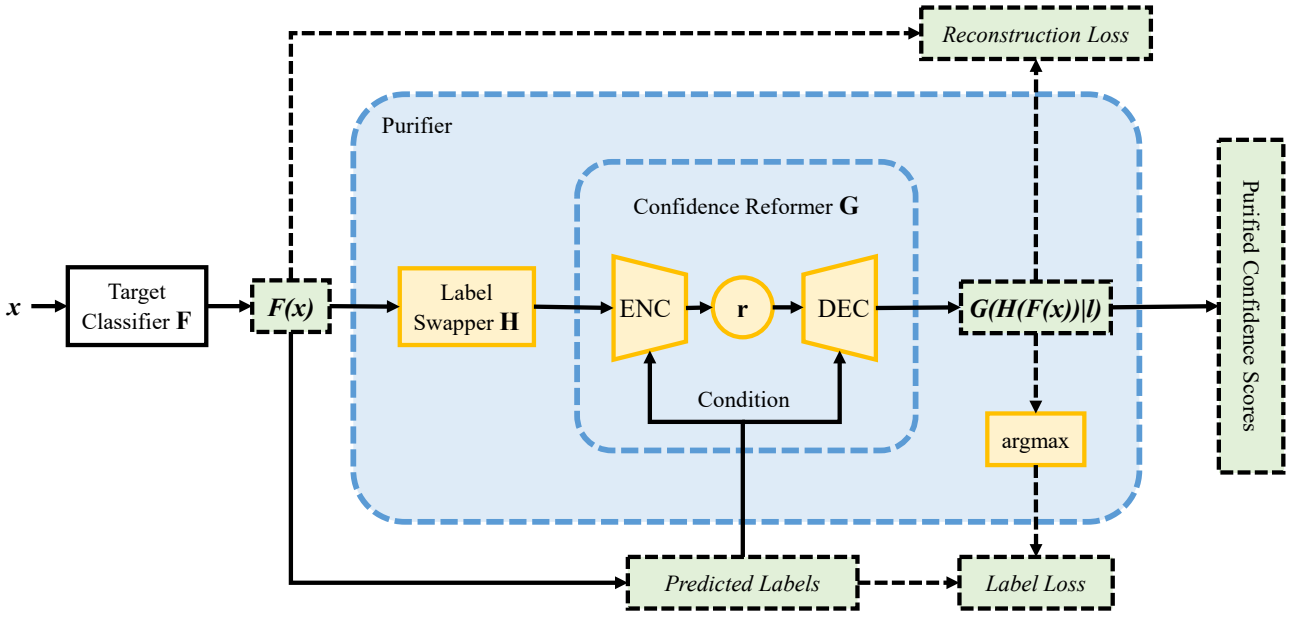
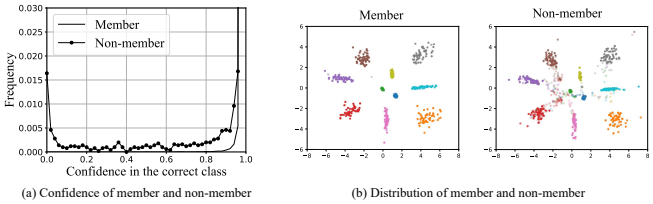


Fig. 1: Architecture of PURIFIER. PURIFIER consists of a *label swapper* H and a *tconfidence reformer* G. H can reduce the gap of classification accuracy between members and non-members by modifying the predicted labels of specific training data. G is a Conditional Variational Auto-encoder(CVAE), with 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.



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

Fig. 2: Individual and statistical distinguishability between members and non-members in cifar10.

Statistical distribution. Distinguishability of statistical distribution means that members' confidence score vectors of samples in the same class are more similar than non-members. We reduce the dimension of the high-dimensional vectors to a plane, as shown in Figure 2 (b). We can see that members data in the same class are more closely clustered than non-members.

Prediction label. The prediction label for a sample is the class corresponded with the maximal score in confidence vector. The distinguishability of prediction label directly leads to the difference between training accuracy and testing accuracy, where the former is higher in general. For example, training accuracy for the classifier on cifar10 is 99.99%, while the testing accuracy is 95.92%, as shown in Table 2.

4.2 Design of Confidence Reformer

The PURIFIER reforms the confidence scores with the *confidence reformer* G, which is a CVAE.

The training process of confidence reformer goes as Algorithm 1. G takes the confidence score vector $F(x)$ as input, with the corresponding label l being the condition.

Algorithm 1: Training process of confidence reformer.

Input: The reference dataset D_{ref}, the target classifier 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**
    $c_{r_j} \leftarrow F(\vec{x}_{ref_j})$;
    $l_{r_j} \leftarrow$ **onehot**$((c_{r_j}))$;
    $g \leftarrow \nabla_\theta \frac{1}{q} \sum_{j=1}^{q} \mathcal{R}(G_\theta(c_{r_j}|l_{r_j}), c_{r_j}) + \lambda \mathcal{L}(G_\theta(c_{r_j}|l_{r_j}), l_{r_j})$;
    $\theta \leftarrow$ **updateParameters**(η, θ, g)
  end
end
return G_θ

$F(x)$ first goes through the encoder, where it is mapped to the encoded latent space $\vec{r}$. The decoder then maps the confidence score back from the latent space $\vec{r}$, and the reformed confidence score $G(F(x)|l)$ is obtained. G is trained on the confidence scores predicted by F on the defender's reference dataset D_{ref}, which consists of non-member samples. Moreover, we should also preserve the classification accuracy. As a result, we also take the label loss into consideration when training G. Formally, G is trained to minimize the following objective function.

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

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

By the training process, G not only learns the pattern of *individual shape* on non-member samples, but also preserve the classification accuracy. As a result, 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 are similar to the non-member ones on D_{ref} (non-members) with the same label l. However, noises introduced in the latent space $\vec{r}$ will increase the reconstruction error. As a result, G 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 breakdown 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.

4.3 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 usually doesn't change which of the scores is maximal when transforming confidence vector. This leads that the confidence reformer can't cope with the distiguishability 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 select training samples to replace their predicted labels with the second largest predicted labels 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 swap members data. At this swap rate p_{swap}, the training accuracy can decrease to testing accuracy, achieving indistiguishability of prediction label.

We can use the label swapper mechanism as follows: When input a vector, judge whether it is a vector of members. If so, swap its label with probability p_{swap}. But this leads to the problem that the prediction label of a member sample may be swapped, or may not, so attacker can get two results if he input it several times, which is a more severe distinguishability between members and non-members.

So we need a pre-prosess for label swapper, as Algorithm 2 goes. we 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 c_j of the sample $(\vec{x}_{train_j}, y_{train_j}) \in D_{swap}$. The original confidence score c_j is added to the prediction indexing set P_{index} and later used by the label swapper to achieve indistinguishability of prediction label. Moreover, the setting of P_{index} also helps to improve efficiency. It is challenging for the label swapper to efficiently store and index the member information in D_{train}.

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 ← ∅;
D_train ← shuffle(D_train);
D_swap ← {(x⃗_{train_j}, y_{train_j})}_{j=1}^{t} ⊂ D_train;
for each (x⃗_{train_j}, y_{train_j}) ∈ D_swap do
    c_j ← F(x⃗_{train_j});
    P_index ← P_index ∪ {c_j};
end
return P_index
```

P_{index} has much smaller dimension and size than D_{train}, which is acceptable.

Simply swapping labels in P_index is not enough to achieve a good defense effect. We should note that small perturbations may be added to the input data by attackers to indirectly infer membership of a target member sample $x \in D_{swap}$. So we can't just simply match whether a input confidence vector is in P_{index}. Instead, H uses k nearest neighbor (kNN) to identify these suspicious noisy members and also swaps their labels.

4.4 Defence process of PURIFIER

It is worth discussing that whether to place label swapper in front of confidence reformer, or reversely. In our conference version, we place confidence reformer in the front, and label swapper serves as a supplement. According to our further study, placing label swapper in the front is a better choice. The transformation effect of confidence reformer appears to be softening. If we swap label later, the softening effect may be destroyed, which can leads to a difference between members and non-members. Our experiment shows the result (TBD...).

Now we can use PURIFIER n defense. In the inference stage, given an input sample $\vec{x}$, we first query the target classifier F to get the confidence score c and the predicted label l. Then, we input c into the *label swapper* H. H checks if c has a match in P_{index} using kNN and swaps the label as l' if c is matched. At this stage, c is indistinguishable in prediction label. Then, we input c into the *confidence reformer* G, with l' (l if not swapped in the previous step) being the condition, to get the purified confidence vector p. This ensures indistinguishability in terms of individual shape and statistical distribution. Finally, PURIFIER returns the purified confidence scores p.

5 EXPERIMENTS

5.1 Datasets And Model Settings in Following Experiments

5.1.1 Datasets

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 [7], [1], [5]. 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 [7], [2], [1], [5]. This dataset is based on Kaggle's "acquired valued shopper" challenge. [1] We used the preprocessed and simplified version of this dataset [7]. It is composed of 197,324 data records and each data record has 600 binary features. The dataset is clustered into 100 classes.

FaceScrub530 [8]. This dataset consists of URLs for 100,000 images of 530 individuals. We obtained the preprocessed and simplified version of this dataset from [8] which has 48,579 facial images and each image is resized to 64×64.

UTKFace. This dataset consists of URLs for 22000 images of individuals. We train the classifier to classify the genders and use the race as the sensitive attribute in our experiments.

CIFAR100. 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 classes has 600 images.

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

Location. 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.

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 reference set D_2 and the test set D_3.

5.1.2 Target Classifier Setting

We use the same model architectures as in previous work. That is,

1) For CIFAR10 and CIFAR100 datasets, we use DenseNet121. We train our classifier with SGD optimizer for 350 epochs with the learning rate 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).
2) For Purchase100 dataset, we use the same model and training strategy as in previous work to train the target classifier. It is a 4-layer fully connected neural network.
3) For FaceScrub530 dataset, we use the same conventional neural network and the same training strategy as in previous work to train the target classifier.
4) 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.
5) For Texas dataset, we use a 4-layer fully connected neural network with the Tanh as the activation function.
6) For Location dataset, we use a 3-layer fully connected neural network with the Tanh as the activation function.

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

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 CIFAR10.

5.2 Effectiveness of PURIFIER

5.2.1 Effectiveness in Membership Inference

Table 2 presents the defense performance of PURIFIER against different membership inference attacks. For each classification task, PURIFIER decreases the attack accuracy as well as preserves the the classification accuracy. PURIFIER reduces the accuracy of NSH attack significantly for different datasets. For instance, it reduces the accuracy of NSH attack from 69.34% to 51.56% in FaceScrub530 dataset. As for Mlleaks attack, the model defended with PURIFIER reduces the attack accuracy to nearly 50%. Comparing with the pure 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% in 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 a more powerful shadow model. As Table 3 shown, 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 PURIFIER is able to mitigate stronger attacks which having more information (e.g., an attacker has more data to train shadow models) than that we assumed in the main paper.

5.2.2 Effectiveness in Adversarial Model Inversion

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 dataset. Although PURIFIER is designed to protect models from membership inference attacks, it turns out that the

TABLE 2: Defense performance of PURIFIER against various attacks. Results of Transfer attack and Boundary attack are reported in AUC. Note that the N.A. means that 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			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	1.4357
	Purifier	95.93%	95.92%	51.65%	50.26%	50.23%	50.64%	50.84%	0.4974	0.4949	1.4939
Purchase100	None	100.00%	84.36%	70.36%	64.43%	N.A.	69.82%	57.82%	0.5431	N.A.	0.1426
	Purifier	84.46%	84.37%	51.71%	50.09%	50.13%	50.96%	51.68%	0.4978	N.A.	0.1520
FaceScrub530	None	100.00%	77.68%	69.34%	75.04%	N.A.	62.61%	61.16%	0.5869	0.7739	0.0114
	Purifier	77.34%	76.11%	51.56%	51.04%	50.00%	50.00%	50.02%	0.4983	0.6185	0.0454
CIFAR100	None	100.00%	69.98%	76.98%	73.78%	N.A.	/	/	0.5980	0.6668	/
	Purifier	66.35%	66.36%	50.01%	50.15%	51.02%	/	/	0.5120	0.4975	/
Texas	None	79.17%	50.91%	66.37%	58.93%	N.A.	/	/	0.5431	N.A.	/
	Purifier	50.99%	47.88%	51.29%	50.00%	51.18%	/	/	0.5028	N.A.	/
Location	None	100.00%	60.44%	82.37%	84.00%	N.A.	/	/	0.5893	N.A.	/
	Purifier	60.50%	59.44%	51.75%	50.31%	51.41%	/	/	0.5015	N.A.	/

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

Dataset	Defense	NSH Attack	Mlleaks Attack
CIFAR10	None	58.46%	70.20%
	Purifer	50.67%	50.01%
Purchase100	None	88.27%	68.50%
	Purifer	52.74%	51.00%
FaceScrub530	None	70.30%	73.46%
	Purifer	50.22%	50.66%

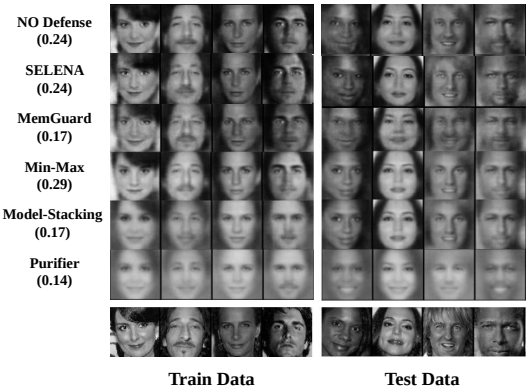


Fig. 3: Model inversion attack against the FaceScrub530 classifier defended by different approaches.

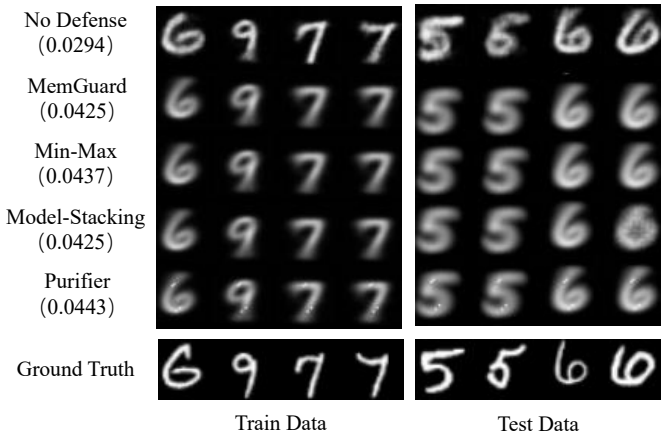


Fig. 4: Model inversion attack against the MNIST classifier defended by different approaches. Numbers on the left indicate inversion errors

TABLE 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	84.20%	82.78%	20.94%

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 [9], which is shown on left side of Figure 3. The less the number is,the less similarity reconstructed samples share with the original samples. Similarly, Figure 4 shows the results against model inversion attack on MNIST. We report the inversion error as the left of Figure 4 to qualify 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 one, only representing a blurred image of their classes.

We report all the inversion error under previous three datasets 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 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 though without any defense.

5.2.3 *Effectiveness in Attribute Inference*

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 20.94% (almost random guessing) after applying PURIFIER.

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

Dataset	Defense	Training acc.	Test acc.	NSH Attack	Mlleaks Attack	BlindMI Attack	Gap Attack	Inversion Error
CIFAR10	Purifier	97.60%	95.92%	**51.65%**	50.26%	**50.64%**	**50.84%**	**1.4939**
	Min-Max	99.40%	94.38%	53.97%	52.93%	53.52%	52.51%	1.4770
	MemGuard	99.99%	95.92%	53.63%	52.24%	52.03%	52.04%	1.4439
	Model-Stacking	95.80%	92.12%	51.93%	51.01%	52.69%	51.84%	1.4723
	MMD Defense	99.99%	87.44%	59.50%	57.60%	58.92%	56.28%	1.4414
	SELENA	98.40%	93.90%	52.14%	52.35%	51.08%	52.25%	1.4350
	One-Hot Encoding	99.99%	95.92%	52.17%	**50.00%**	51.88%	52.04%	1.4414
	Random Noise	99.99%	95.92%	55.97%	50.01%	51.69%	52.04%	1.4342
Purchase100	Purifier	86.59%	83.23%	**51.71%**	50.09%	**50.96%**	**51.68%**	0.1520
	Min-Max	99.89%	82.03%	65.13%	63.95%	57.39%	58.93%	0.1428
	MemGuard	100.00%	84.36%	62.28%	57.86%	61.35%	57.82%	0.1426
	Model-Stacking	81.84%	69.68%	61.16%	55.53%	60.36%	56.08%	0.1472
	MMD Defense	100.00%	82.65%	69.48%	69.89%	66.62%	58.67%	0.1439
	SELENA	83.24%	79.53%	51.90%	52.97%	53.04%	51.83%	0.1440
	One-Hot Encoding	100.00%	84.36%	57.65%	**50.00%**	57.67%	57.82%	**0.1524**
	Random Noise	100.00%	84.36%	60.06%	50.02%	54.44%	57.82%	0.1409
FaceScrub530	Purifier	77.58%	77.52%	**51.56%**	51.04%	**50.00%**	**50.03%**	**0.0454**
	Min-Max	98.99%	68.31%	65.56%	69.84%	66.16%	65.34%	0.0182
	MemGuard	100.00%	77.68%	62.48%	60.06%	62.72%	61.16%	0.0117
	Model-Stacking	86.30%	57.05%	62.00%	51.86%	60.62%	64.63%	0.0417
	MMD Defense	100.00%	77.38%	64.88%	67.95%	63.55%	61.31%	0.0111
	SELENA	81.06%	72.05%	51.68%	51.23%	54.05%	50.50%	0.0131
	One-Hot Encoding	100.00%	77.68%	57.87%	**50.00%**	61.23%	61.16%	0.0420
	Random Noise	100.00%	77.68%	56.85%	50.04%	60.83%	61.16%	0.0175
CIFAR100	Purifier	70.02%	69.98%	**50.01%**	50.15%	/	/	/
	SELENA	78.00%	62.10%	50.32%	**50.42%**	/	/	/
Texas	Purifier	51.01%	50.91%	**51.29%**	50.00%	/	/	/
	SELENA	77.90%	55.25%	54.40%	**51.00%**	/	/	/
Location	Purifier	60.45%	60.43%	**51.75%**	50.31%	/	/	/
	SELENA	77.90%	55.25%	54.40%	**51.00%**	/	/	/

5.3 Comparison With Other Defenses

We compare PURIFIER with following defenses. ①Min-Max [2]. ②MemGuard [10]. ③Model-Stacking [1]. ④MMD Defense [6]. ⑤SELENA [11]. ⑥One-Hot Encoding.

Table 5 shows the defense performance of PURIFIER and other defense methods against membership inference attacks under different datasets. PURIFIER achieves the best defense performance against most of the attacks, including the NSH attack, the BlindMI attack and the gap attack compared to other methods on all of the six datasets. For the Mlleaks Attack, PURIFIER can achieve the second best performance only 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 intolerable reduction in model's test accuracy. For One-Hot Encoding and Random Noise, their transformation on confidence vectors leads to a large degree of semantic information loss. MemGuard reaches acceptable defense performance with negligible decline in test accuracy. However, its defense performance is not as good as that of PURIFIER.

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 not protect the model from adversaries recovering small details of the original image. It 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 can protect the target model against adversarial model inversion attack effectively.

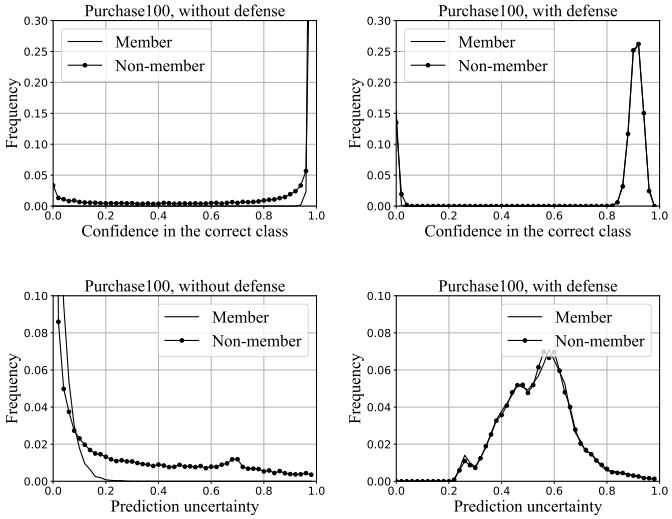


Fig. 5: Distribution of the target classifier's confidence in predicting the correct class and the prediction uncertainty on members and non-members of training set.

5.4 Indistinguishability in Purified Scores

In this subsection, we analyze how the purified confidence scores affect membership inference attacks by evaluating three indistinguishabilities: individual, statistical and label.

5.4.1 *Individual Indistinguishability in Purified Scores*

PURIFIER reshapes the input confidence score vectors according to the pattern of the learned non-member samples. We

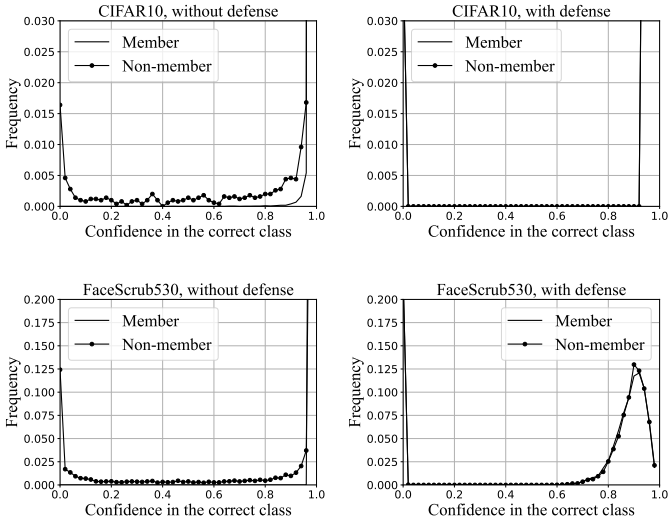


Fig. 6: Distribution of the target classifier's confidence in predicting the correct class on members and non-members of its training set.

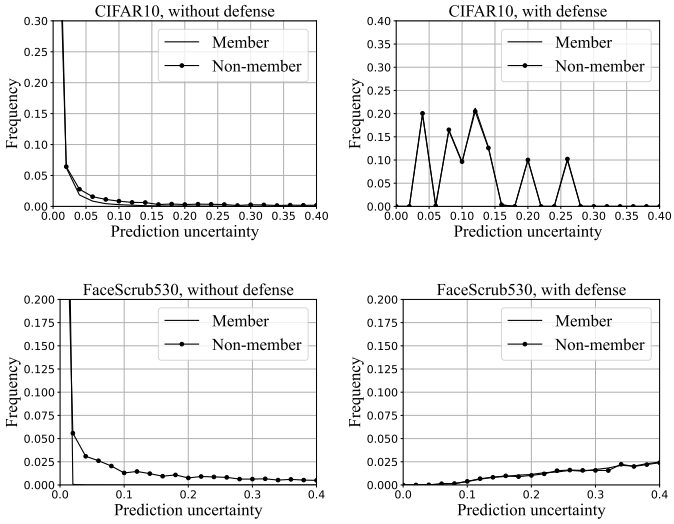


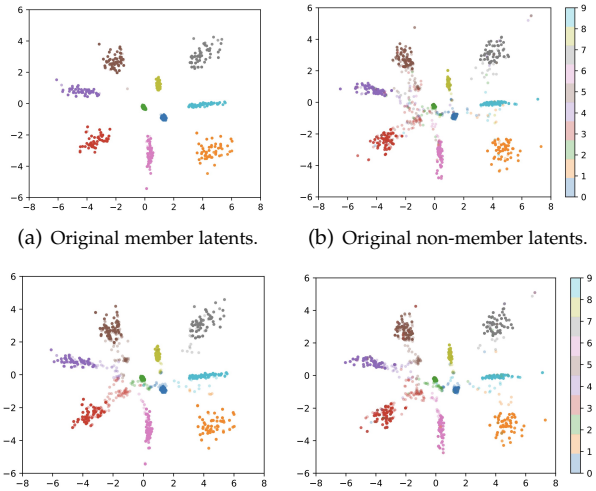
Fig. 7: Distribution of the target classifier's prediction uncertainty on members and non-members of its training set. The uncertainty is measured as the normalized entropy of the confidence score vector.

examine the indistinguishability of the confidence scores on members and non-members by plotting 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 can reduce the gap between the two curves. Similar curves 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 6. 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.

TABLE 6: 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



(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.

5.4.2 Statistical Indistinguishability in Purified Scores

We present the statistical distribution of confidence score vectors in the encoder latent space of the *confidence reformer*. Figure 8 visibly displays the differences on the CIFAR10 dataset between members and non-members in the latent space. 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 the confidence score vectors, making 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 predicted labels of members.

label swapper incurs negligible reduction of test accuracy. At the same time, swapping the labels of the member samples

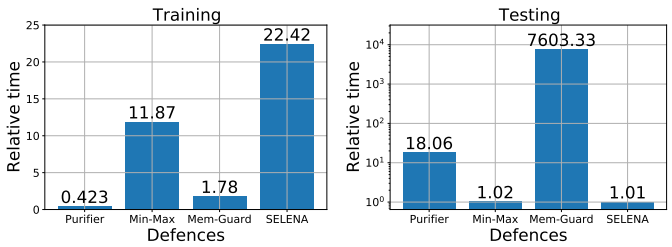


Fig. 9: Efficiency of different defense methods.

reduces the training accuracy so that the gap between the accuracy of member and non-members 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 purified member confidence vectors are less distinguishable from those of the non-members in terms of label.

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 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 all the other methods. 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 Experiment About New And Old PURIFIER

TBD

6 DISCUSSION

Inference about the reference data. 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. Similarly, our approach uses it to train the PURIFIER. 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 7. 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. However, there might be opposing views on whether such reference datasets should be considered as members.

Effect of PURIFIER's training data. We also investigate the effect of the PURIFIER's training data by using different in/out-distribution data to train PURIFIER. Specifically, for in-distribution data, we vary the size of D_2 and also replace D_2 with D_1. For out-of-distribution data, we use CIFAR10 data to train the PURIFIER for the FaceScrub530 classifier, and use randomly generated data to train the PURIFIER for the Purchase100 classifier.

TABLE 7: 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	0.0435	51.23%
Min-Max	0.0222	52.07%
MemGuard	0.0116	52.20%

TABLE 8: Effect of the PURIFIER's in-distribution training data on the defense performance. The numbers are reported on the Purchase100 dataset.

Training set	Classification acc	NSH	Mlleaks	Adaptive
D_2 (5,000)	83.85%	52.63%	50.09%	50.62%
D_2 (10,000)	83.47%	51.72%	50.12%	50.14%
D_2 (20,000)	83.23%	51.71%	50.09%	50.13%
D_2 (40,000)	81.39%	51.93%	50.09%	50.20%
D_2 (60,000)	81.75%	51.84%	50.09%	50.10%
D_1 (20,000)	84.23%	52.29%	50.12%	50.07%

We present the effect of the in-distribution training data in Table 8. 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 of 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 9 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: Effect of the PURIFIER's out-of-distribution training data on the defense performance.

Classifier	Purifier	Classification acc	NSH	Mlleaks	Adaptive
FaceScrub530	CIFAR10	40.05%	54.54%	50.11%	50.50%
Purchase100	Random	7.76%	51.55%	50.20%	50.71%

Effect of PURIFIER to detect noisy members. Furthermore, we investigate the effectiveness of PURIFIER to detect noisy members, which means the members added noise by attackers intentionally. We use the adversarial attack methods

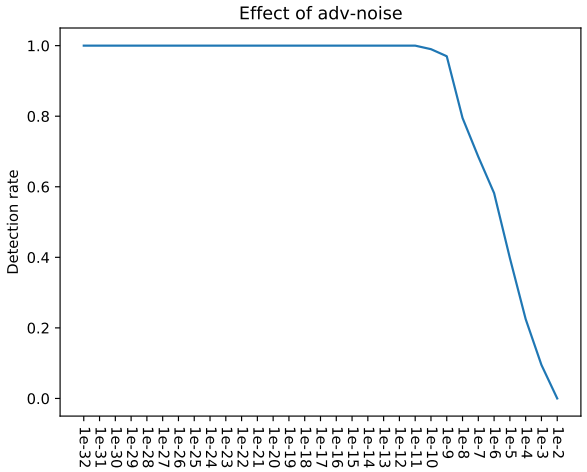


Fig. 10: The proportion of noise data that PURIFIER can detect under the FGSM attacks on the FaceScrub530 dataset.

TABLE 10: Ablation study on the Label Swapper.

Dataset	Defense	Transfer Attack
CIFAR10	None	0.5048
	CVAE	0.5045
	CVAE + Label Swapper	0.4974
CIFAR100	None	0.6668
	CVAE	0.6632
	CVAE + Label Swapper	0.4975

(FGSM)[2] to create noisy members. As shown in Figure 10 that PURIFIER can accurately detect the members with noise $\|\eta\|_\infty < 1e-10$ on FaceScrub530 dataset.

Ablation study on the Label Swapper. We conduct the ablation study on the label swapper to investigate the effectiveness of it in our defense mechanism. Given our design of label swapper to mitigate the attackers using only the hard labels to perform attacks, we verify its efficiency under label only transfer attack. Table 10 shows a significant AUC drop in label only transfer attack with the help of label swapper compared by those mechanisms without it. Additionally, the result indicates that label only transfer attack can obtain approximately the same AUC among models without any defense and PURIFIER without the label swapper, which means 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 PURIFIER to mitigate the label only attacks.

Cost of PURIFIER in real-world setting. We discuss the latency and the memory cost introduced by PURIFIER at this section. We report that the training time of the CIFAR10 model is 3.18h, and the extra cost introduced by PURIFIER in it is 1.34h. Compared with 71.30h SELENA has introduced, PURIFIER contributes to less latency time. Besides the training cost, PURIFIER will spend extra 282.18s to predict 50k images in the inference time. We speculate that the latency of PURIFIER is mostly caused by kNN. Hence we expect to replace kNN by a more efficient component in the future work. For the memory cost, PURIFIER costs extra space for swap set. The sizes of it with different datasets are respectively 78.23K for CIFAR10, 1.14M for Purchase and 13.5M for Facescrub530. We believe the extra memory cost by PURIFIER is negligible compared to the original training set.

[2]ICLR 2015 Explaining and Harnessing Adversarial Examples

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 the input confidence score vectors to this pattern. It makes members' confidence score vectors indistinguishable from non-members' in terms of individual shape, statistical distribution and prediction label. Our 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] 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.

[2] 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.

[3] 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.

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

[5] 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.

[6] 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

[7] 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.

[8] 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.

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

[10] 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.

[11] 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

PLACE
PHOTO
HERE