Defending Model Inversion and Membership Inference Attacks via Prediction Purification

Ziqi Yang  
National University of Singapore  
yziqi07@gmail.com

Bin Shao  
Zhejiang University  
shaobin_zju@zju.edu.cn

Bohan Xuan  
Zhejiang University  
xuanbohan@zju.edu.cn

Ee-Chien Chang  
National University of Singapore  
changec@comp.nus.edu.sg

ABSTRACT

Neural networks are susceptible to data inference attacks such as the model inversion attack and the membership inference attack, where the attacker could infer the reconstruction and the membership of a data sample from the confidence scores predicted by the target classifier. In this paper, we propose a common approach, namely purification framework, to defend data inference attacks. It purifies the confidence score vectors predicted by the target classifier, with the goal of removing redundant information that could be exploited by the attacker to perform the inferences. Specifically, we design a purifier model which takes a confidence score vector as input and reshapes it to meet the defense goals. It does not retrain the target classifier. The purifier can be used to mitigate the model inversion attack, the membership inference attack or both attacks. We evaluate our approach on deep neural networks using benchmark datasets. We show that the purification framework can effectively defend the model inversion attack and the membership inference attack, while introducing negligible utility loss to the target classifier (e.g., less than 0.3% classification accuracy drop). Moreover, we also empirically show that it is possible to defend data inference attacks with negligible change to the generalization ability of the classification function.

1 INTRODUCTION

Machine learning has been widely adopted in a variety of applications, transforming many aspects of our daily life such as handling users’ sensitive data. Machine learning itself is also been provided as a service, e.g., machine-learning-as-a-service, by many platforms. Users access these models through prediction APIs which return a prediction score vector. Such vector is a probability distribution over the possible classes and each score indicates the confidence in predicting the corresponding class. The class with the largest confidence is predicted as the label of the input data. In this paper, we are interested in data inference attacks, notably membership inference and model inversion that exploit such prediction scores to threaten the privacy and security of machine learning.

A series of studies has indicated that the prediction scores of black-box machine learning models could be exploited to perform data inference attacks to get useful information about the data on which the machine learning model operates [14, 20, 24, 35, 36, 47, 49, 60]. For examples, membership inference attack [24, 35, 47, 49] and model inversion attack [14, 20, 60] are two of the most important and exemplary ones. In a membership inference attack, the adversary is asked to determine whether a given data sample is in the target model’s training data or not according to its confidence scores predicted by the target model. Specifically, the adversary trains a binary classifier which takes the prediction scores as input and predicts whether the data sample is a member or non-member of the target model. In a model inversion attack, the adversary aims at inferring information about the query data from the prediction scores such as the sensitive attributes [15, 19, 58] or the reconstruction of the sample [14, 20, 60]. Recently, Yang et al. [60] proposed an effective black-box model inversion attack where the attacker leverages auxiliary knowledge to construct an inversion model which can reconstruct the original input sample from the prediction scores with high accuracy.

The main reason of why such data inference attacks work is that the prediction scores contain not only confidence of classifying the query data but also unwanted redundant information, i.e., membership information and inversion information, which could be exploited to infer useful information about the query data. For instance, the major cause of membership inference attack is that the prediction scores of the target model are distinguishable for members and non-members of the training data. Such distinguishability leaks membership information to the attacker. Overfitting is believed to be one of the major reasons causing the distinguishability, but is shown to be not the only one [49].

A number of approaches have been proposed to mitigate membership inference attack in the literature. These approaches mainly leverage various regularization techniques to reduce overfitting, such as $L_2$ regularizer [49], dropout [47], model-stacking [47] and min-max regularization [35]. However, they do not impose a direct reduction of the distinguishability. Moreover, these defenses require retraining the target model which is considered less efficient especially for complicated neural networks. Jia et al. [24] recently proposed MemGuard which, instead of reducing overfitting, transforms the prediction scores to an adversarial example to evade the attacker’s membership classification. Although it does not retrain the target model, the defense effectiveness is dependent on the transferability [41] of adversarial examples, which might not generally reduce the distinguishability of prediction scores. Another set of defenses use differential privacy mechanisms [1, 23, 48].
These approaches can provide a theoretical guarantee of privacy but impose a significant classification accuracy loss [35].

Most previous defense approaches of data inference attacks focused on the membership inference attack. Unfortunately, little has been studied about the defense of model inversion attack.

In this paper, we propose a common purification framework to defend data inference attacks by “purifying” the prediction scores. The framework takes the prediction scores of a trained target model as input and produces a purified version to satisfy one or both of these defense goals: (I) preventing model inversion attack and (II) preventing membership inference attack. The intuition is to remove the redundant information hidden in the prediction scores which could be exploited by the attacker. Specifically, in the model inversion attack, the attacker exploits the redundant information to infer a reconstruction of an input sample, while in the membership inference attack, the attacker exploits the distinguishability between the prediction scores on members and non-members to infer the membership.

We achieve the purification framework by training a purifier, which takes the confidence scores of the target model as input and purifies them. An additional adversarial model is trained for Defense Goal I (i.e., preventing model inversion), and an additional discriminator is trained for Defense Goal II (i.e., preventing membership inference). The adversarial model and the discriminator can be discarded after training. The purifier and the original classifier will work as a black box to classify the query data. The purifier has a similar structure as autoencoder [4] and thus can learn a latent representation of a dataset. It is trained to optimize two loss functions: a reconstruction loss of the original confidence scores and the purified version, and a cross entropy loss of the purified version and the originally predicted label. Such training process leads to a purifier that not only captures the latent representation (features) of the original confidence scores but also imposes minimized distortion to them and the classification accuracy loss.

To achieve the Defense Goal I, we train the purifier by also minimizing the redundant information in the prediction scores that is useful for the attacker to infer the reconstruction of the input data. Specifically, we anticipate an additional adversarial model which adaptively performs model inversion attack against the purifier and the purifier keeps updating the prediction scores to minimize the inversion accuracy. Eventually, the adversarial model evolves to be the supposedly strongest inversion model and the purifier purifies the prediction scores with minimized useful information for the mode inversion attack. We formulate such training as a min-max game between the purifier and the adversarial model, and jointly train them. The training data could be the same training set of the target model or another set drawn from the same data distribution.

The purifier achieves the Defense Goal II by purifying the confidence scores on members or non-members as if they were predicted on non-member data or a reconstruction, and the purifier learns to fool the discriminator to make mistakes. Eventually, the purifier improves the reconstructed confidence scores.

The purifier can concurrently attain both defense goals when we use the validation set to jointly train the purifier, the adversarial model and the discriminator. The result of the joint training is that, at the equilibrium point, the purifier can not only purify the confidence scores as if they were predicted on non-members but also minimize the inversion accuracy of the corresponding strongest inversion model. Besides the defenses, the purifier is trained to minimize the distortion to the confidence scores as well as the classification accuracy loss.

Our experimental results show that the purification framework can effectively decrease the model inversion accuracy and the membership inference accuracy at the same time. For example, our experimental results on the FaceScrub classifier [60] show that the individual-specific facial features in the inverted facial images are largely reduced. The defense performance against the membership inference attack outperforms existing approaches on most of the evaluated datasets. Moreover, our approach introduces negligible utility loss. For example, the classification accuracy drop is within 0.3% on all evaluated datasets. We also empirically show that our framework is effective because it removes redundant information useful for the attacker, rather than changing the generalization error of the target model.

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

- We propose an effective adversarial training approach to mitigate the model inversion attack on classification models, by minimizing the useful information for the attacker to infer the reconstruction of the input sample.
- We propose a defense approach of membership inference attack by transforming the confidence score vector on any data to behave as if it is predicted on non-member data. This decreases the distinguishability of the prediction on members vs. non-members.
- We propose a common purification framework to defend data inference attacks including the model inversion attack and membership inference attack. It purifies a confidence score vector by removing the redundant information useful for the attacker.
- We empirically show that it is possible to effectively mitigate data inference attacks with negligible change to the generalization error of the classification function.

2 INFERENCES ON MACHINE LEARNING

It has been shown that machine learning models are vulnerable to various inference attacks [15, 49, 60, 61], 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 [37, 40, 54, 55]. Data inference, on the contrary, focuses on extracting information about the data on which the target model operates [2, 14, 15, 49, 57, 58, 60, 61]. In this paper, we concentrate on two of the most important and exemplary data inference attacks, notably membership inference attack [49] and model inversion attack [14, 60]. In this section, we first introduce these two data inference attacks and then introduce existing defenses. Finally, we analyze the limitations of existing defense mechanisms.

2.1 Data Inference Attacks

Membership inference and model inversion attacks are two 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 [24, 30, 31, 36, 47, 49].

Confidence-based Attack [47, 49], Shokri et al. [49] introduced 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. Salem et al. [47] further showed that it is sufficient to train only one shadow model to replicate the target model for membership inference attack. Besides, they also showed that ranking the elements in the confidence score vectors before inputting them to the attack model could improve the inference accuracy. For example, their experimental results show that only the top one/three highest values in the confidence vector are sufficient to result in effective membership inference. These results indicate that the assumptions of membership inference attack could be largely relaxed which might lead to more practical threats.

Confidence & Label-based Attack [35], Nasr et al. [35] extended the attack model by also taking the label information as input. Their attack model is composed of three neural networks. The first two networks operate on the confidence score vector and the one-hot encoded label respectively. They have the same size of input dimension, i.e., the number of classes of the target model. The third network operates on the concatenation of the output of the first two networks and predicts the membership. They assume that the attacker has a subset of the members and non-members of the target model’s training data, and thus they do not train shadow models.

More settings of membership inference attack have been studied in the literature. For examples, Nasr et al. [36] proposed membership inference attack in the white-box setting, where the attacker computes the gradients of the white-box target model with respect to the given data sample as features for membership inference. There are also research efforts on membership inference attack in federated learning [32, 36] and against generative models [18]. In this paper, we consider membership inference attack in the black-box setting against standalone centralized classification models.

Model Inversion Attack. Model inversion aims to reconstruct the input data from its confidence scores predicted by the target model. Fredrikson et al. [14] 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. [60] recently 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. Their experimental results showed significant improvement of the inversion accuracy over previous works. They also performed accurate inversion attacks against real-world commercial facial recognition services.

Some other methods were proposed to infer sensitive attributes [15, 19, 58, 63] or statistical information [2] about the training data instead of reconstructing the specific input data. There are also studies [20] of inversion attack in the setting of federated learning where the attacker has white-box access to the global model. In this paper, we consider on model inversion attack that aims to reconstruct the input data in the black-box setting against centralized classification models.

2.2 Defenses against Data Inference Attacks

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

Overfitting, i.e., machine learning models behave more confidently on their training data (members) than others, is believed to be one of the major reasons that make the confidence scores distinguishable for members and non-members [49]. When a model overfits on training data (i.e., members), the confidence scores of the model on members and non-members exhibit significantly different patterns which makes it easier for the attacker to distinguish them. In line of this, a number of studies make use of various regularization techniques and ensemble learning to reduce overfitting.

- **L2-regularizer** [49]. The L2 regularizer is an L2 norm of the model parameters added as a weighted penalty term to the original loss function. In [49], the authors showed that using L2-regularizer to train the target model can help mitigate membership inference attack.

- **Dropout** [47]. Dropout is another technique used to regularize neural networks [53]. It works by dropping a neuron with a certain probability during the network training. Salem et al. [47] used dropout to mitigate the membership inference attack.
• **Min-Max Game** [35]. Nasr et al. [35] proposed to add an adversarial regularizer to the original loss function of the target model such that the target model is trained to minimize the prediction loss and to also maximize the membership privacy. The training process is formulated as a min-max optimization problem.

• **Model Stacking** [47]. Model stacking is essentially an ensemble approach which combines multiple simple classifiers as a complicated one to make the final prediction [39, 44]. It is often used as a way of reducing overfitting [51]. Salem et al. [47] leveraged model stacking to mitigate membership inference attack.

**MemGuard** [24]. While most existing defenses focus on reducing overfitting to mitigate membership inference attack, Jia et al. [24] studied 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 his own “attack model” which works similarly as the attacker’s attack model, and thus he can craft the adversarial example against his attack model in a white-box manner. Such adversarial example is likely to also evade the membership classification of the attacker’s attack model due to the transferability of adversarial examples [9, 41, 42].

**Differential Privacy.** Differential privacy [13] is a widely used privacy-preserving technique. A number of studies have explored differential privacy to mitigate membership inference attack. For example, some methods add noise to the objective function of the model [8, 23, 26], while others add noise to the gradient of the model during minimizing the objective function [1, 5, 52, 56, 62]. Differential privacy is able to provide theoretical privacy guarantee but at the cost of significant loss of classification accuracy [28, 35].

### 2.3 Limitations of Existing Defenses

Previous studies of defense mechanisms against the membership inference attack did not discuss their impact on the 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 both membership inference and membership attacks is available.

It is shown that overfitting is not the only reason that causes membership inference attack [49]. 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 [49]. As what existing defense mechanisms already did, reducing overfitting contributes to the decrease of such distinguishability. However, such defense methods could be more effective if the distinguishability can be directly reduced.

The confidence scores of a machine learning classifier tell more useful information about the query data beyond the predicted label. Therefore, one defense mechanism should introduce negligible distortion to them. Most of previous defenses retrain the target model which not only disorganizes the confidence scores of the target model but also becomes inefficient especially for large and complicated neural networks. MemGuard [24] is designed to have a hyper-parameter to control the distortion of the confidence scores. However, the effectiveness of this defense method is dependent on the transferability of adversarial examples. Besides, turning the confidence score vector into an adversarial example might not lead to a reduction of the general distinguishability between members and non-members. Crafting an adversarial example for each query data is considered less efficient compared to other defense mechanisms [28] in the testing phase.

Differential privacy, though provides a theoretical guarantee of privacy, has a significant impact on the classification accuracy loss of the target model compared with other defense mechanisms [35]. Although differential privacy prevents an attacker from gaining additional information by including or excluding an individual data record, the information leaked from the released prediction scores (through which an attacker can perform model inversion attack) is not discussed in previous studies.

### 3 PROBLEM FORMULATION

We focus on the supervised learning, more specifically, on training classification models (classifiers) using neural networks [27]. We have three parties in our problem, namely **model owner**, **attacker** and **defender**. The model owner trains a machine learning classifier on its private training dataset. We refer to this classifier as **target classifier**. The attacker aims to launch data inference attacks against the target classifier. The defender aims to provide protection to the target classifier against the attacker.

#### 3.1 Model Owner

The model owner trains a machine learning classifier $F$ on his private training dataset $D_{\text{train}}$. He also has a validation dataset $D_{\text{val}}$ to test whether $F$ functions properly. Both $D_{\text{train}}$ and $D_{\text{val}}$ are drawn from the same underlying data distribution $p_{\text{D}}(x)$. The classifier $F$ is trained with the goal of making predictions on unseen data which we refer to as test dataset $D_{\text{test}}$. Let $x$ represent the data drawn from $p_{\text{D}}$, and $y$ be the vectorized class of $x$. The training objective is to find a function $F_{\text{w}}$ to well approximate the relation between each data point $(x, y)$. Formally, we have:

$$F : x \mapsto y$$

The training process is to optimize an objective function $L(F)$ and terminates typically when the classification accuracy on the validation set $D_{\text{val}}$ achieves the best [7]. 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 $x \in D_{\text{test}}$ through the prediction APIs. The classifier $F$ returns a prediction score vector $F(x)$ to the users. The prediction score vector is a probability distribution of the classifier’s confidence over all the possible classes. For example, the $i$-th element $F(x)_i$ is the probability of the data $x$ belonging to class $i$. We usually take the class with the maximum probability to be the predicted label of the data $x$. 


3.2 Attacker
The attacker aims at performing data inference attacks against the target classifier \( F \). We consider that the classifier \( F \) works as a black-box “oracle” to the attacker, i.e., the attacker can only query \( F \) with his data sample \( x \) and obtain the prediction scores \( F(x) \). The attacker is also assumed to have auxiliary information \( \mathcal{A} \) such as the ground-truth label of \( x \) and a set of data samples drawn from a similar data distribution \( p_x \) as the target classifier’s training data distribution \( p_x \). Formally, given a prediction vector \( F(x) \) on some victim data point \( x \), the attacker wants to find an attack function \( A(F(x)), O(F), \mathcal{A} \) in the following:
\[
A(F(x), O(F), \mathcal{A}) = \begin{cases} 
  m \in \{0, 1\}, & \text{membership inference} \\
  \hat{x}, & \text{model inversion}
\end{cases} \tag{1}
\]
where \( O(F) \) represents the attacker’s black-box access to the oracle classifier \( F \), \( m \) denotes the membership of the victim data \( x \), and \( \hat{x} \) is the reconstruction of \( x \). The membership \( m \) has two possible values: 0 and 1, where \( m = 1 \) indicates that \( x \) is a member of the target classifier’s training data while \( m = 0 \) means non-member.

3.3 Defender
The defender protects the target classifier from data inference attacks. It could be the model owner or a trusted third party who has access to the target classifier’s prediction score vectors and the validation dataset \( D_{val} \). For any query to the target classifier from users, the defender modifies the prediction score vector of the target classifier with the goal of removing redundant information that the attacker could exploit to perform data inference attacks before returning it to users. The attacker has access only to the modified prediction scores from the defender. In particular, the defender wants to achieve the following goals:

- **Defense Goal**. The defender aims to achieve one or both of these defense goals. **(Defense Goal I)** The defender wants to make the error of \( A \) on reconstructing the input data \( x \) large enough such that the attacker is unable to infer anything specific about \( x \) from the \( \hat{x} \). **(Defense Goal II)** The defender wants to make the attack function \( A \) inaccurate at inferring the membership of a given data sample.

- **Utility Goal**. The defender aims to introduce insignificant distortion to the confidence score vectors of the target classifier as well as negligible classification accuracy loss.

- **Efficiency Goal**. The defense mechanism should introduce negligible time overhead to deploy and use.

4 APPROACH: PURIFICATION FRAMEWORK
We propose a purification framework to defend data inference attacks, i.e., model inversion attack and membership inference attack, by purifying the confidence scores of the target classifier such that the attacker could not exploit the purified confidence scores to infer inversion and membership information about the data. The purification framework does not tamper with the training process of the target classifier.

The purification framework is designed towards achieving the defense, utility and efficiency goals. The architecture of the purification framework for each defense goal is shown in Figure 1. The common component of the purification framework is a purifier \( G \) for all defense goals. It takes the confidence score vector \( F(x) \) of the target classifier \( F \) as input and modifies it such that the redundant membership and inversion information is removed while introducing negligible distortion to \( F(x) \) and negligible classification accuracy loss. An additional adversarial model \( H \) is trained for the Defense Goal I and an additional discriminator \( I \) is trained for the Defense Goal II. After training, \( H \) and \( I \) can be discarded. The purifier \( G \) and the target classifier \( F \) will work as a black box to classify the query data.

4.1 Base of Purification: Purifier \( G \)
The purifier \( G \) is the base of the purification framework, which preserves the utility of the target classifier \( F \) and can be extended to support the defense of data inference attacks with the help of \( H \) or (and) \( I \). Specifically, \( G \) has a similar architecture as autoencoder [4] as shown in Figure 2. It is able to learn to copy its input to its output, in which process it learns the latent representation (features) for a set of confidence score vectors. In order to minimize the distortion to \( F(x) \) and the classification accuracy loss caused by \( G \), we train \( G \) to optimize the following objective function.
\[
\min_G L(G) = \mathbb{E}_{x \sim p_x} [R(G(F(x)), F(x))] + \alpha L(G(F(x), \arg \max F(x))) \tag{2}
\]
where \( R \) is a reconstruction loss function (we use \( L_2 \) norm) and \( L \) is the cross entropy loss function.

Throughout the joint optimization of the two loss functions, the purifier \( G \) is encouraged to minimize the distortion introduced by \( G(F(x)) \) to \( F(x) \) as indicated in \( R \). On the other hand, \( G(F(x)) \) will maximally preserve the original classification accuracy because \( G \) is trained to predict the label predicted by \( F \) as indicated in \( L \). The parameter \( \alpha \) controls the balance of the two loss functions during optimization. As a result, the purifier \( G \) achieves our utility goal.

4.2 Defense Goal I: Joint \( G \) and \( H \)
When we jointly train the purifier \( G \) and an additional adversarial model \( H \) in an adversarial training process, the purifier \( G \) can be extended to prevent the model inversion attack, i.e., achieving the Defense Goal I. The training data could be a dataset drawn from \( p_x(x) \) such as \( D_{train} \) or \( D_{val} \). Figure 1(a) presents the architecture of the purification framework.

In the model inversion attack, the attack function \( A \) could be constructed in any unexpected ways with the goal of bypassing a particular defense mechanism. In response to this, we propose to anticipate a supposedly strongest inversion function \( H \) and train \( G \) to minimize the inversion accuracy of \( H \). Intuitively, the attacker could not design a better attack function \( A \) than \( H \) which is already prepared during the training of \( G \). We model such training process as a min-max adversarial game between \( G \) and \( H \) which is a similar way as many adversarial processes for machine learning [11, 16, 33, 35]. Formally, \( H \) is trained to optimize the following training objective.
\[
\max_H L(H) = \mathbb{E}_{x \sim p_x} [-R(x, H(G(F(x))))] \tag{3}
\]
where \( R \) is the reconstruction loss function which is \( L_2 \) norm in our work. This is exactly the same training objective of the attacker’s
The purifier $G$ will work as a black box (i.e., the yellow part) to classify query data. It has a similar structure as autoencoder as shown in Figure 2 and thus can be used to learn the distribution of the confidence score vector $F(x)$. The target classifier $F$ is fixed during training $G$. After training, $G$ and $F$ will work as a black box (i.e., the yellow part) to classify query data.

### 4.3 Defense Goal II: Joint $G$ and $I$

When we jointly train the purifier $G$ and an additional discriminator $I$ on the defender’s validation dataset $D_{val}$ (i.e., non-member data), the purifier $G$ can be extended to prevent the membership inference attack, i.e., achieving the Defense Goal II. Figure 1(b) presents the architecture of the purification framework.

The intuition of defending the membership inference attack is to, for every query data $x$, make the purified $G(F(x))$ behave as if it is predicted on a non-member sample. Hence, the attacker could not find an attack function $A$ to effectively distinguish its membership. To this end, we train $G$ on $D_{val}$ which are non-members of $F$’s training data. $G$ is able to learn the latent representation of $F$’s confidence scores on these non-members because of its autoencoder structure. Any $F(x)$ is expected be purified to $G(F(x))$ behaving as if predicted on non-member data. To further make $G(F(x))$ more real to evade the attacker’s membership classification, we jointly train $G$ with a discriminator $I$ in an adversarial training process. Specifically, $I$ is trained to distinguish real $F(x)$ and fake $G(F(x))$ by optimizing the following objective function.

\[
\max_{I} \mathbb{E}_{x \sim p_c(x)} \left[ \log I(F(x)) + \log(1 - I(G(F(x)))) \right] \tag{6}
\]

where $p_c(x)$ represent the conditional probability of $x$ for samples in $D_{val}$. The purifier $G$ aims to fool $I$ to misclassify $G(F(x))$ as real. Formally, we have the following objective function to train $G$.

\[
\min_{G} L(G)_3 = \mathbb{E}_{x \sim p_c(x)} \left[ L(G)_1 + \beta \log(1 - I(G(F(x)))) \right] \tag{7}
\]

where $\beta$ is a parameter controlling the importance of the loss functions during training.

We formalize the two objective functions (i.e., function 6 and 7) in one min-max optimization problem to jointly train $G$ and $I$ to
Table 1: Data allocation. A dataset is divided into training set $D_1$ of the target classifier, validation set $D_2$ and test set $D_3$. In membership inference attacks, 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| 10,000| 8,000  | 4,000  |

find the equilibrium point.

\[
\min_{G} \max_{I} L(G, I) = \mathbb{E}_{x \sim p_{\text{data}}(x)} \left[ L(G) + \beta \log I(F(x)) \right] + \beta [\log (1 - I(G(F(x))))] \tag{8}
\]

Similarly, the purifier $G$ and the discriminator $I$ are trained alternatively in one mini-batch to find the best responses against each other. Note that, the discriminator $I$ is used to distinguish real or fake confidence scores and the function $8$ encourages $G$ to generate real confidence scores as if they were predicted on non-members. This is different from the min-max game in [35] where the inference model is used to classify the membership from the target classifier’s predictions and the target classifier is retrained on the original $D_{\text{train}}$ and a reference set.

4.4 Defense Goal I & II: Joint $G$, $H$ and $I$

The purifier $G$ can be extended to defend both model inversion attack and membership inference attack if we jointly train $G$, $H$ and $I$ on the validation dataset $D_{\text{val}}$ (i.e., non-member data). Figure 1(c) presents the architecture. We formalize the joint training of the three models in a min-max-max optimization problem.

\[
\min_{G} \max_{H} \max_{I} L(G, H, I) = \mathbb{E}_{x \sim p_{\text{data}}(x)} \left[ L(G) - \lambda R(x, H(F(x))) \right] + \beta [\log (1 - I(G(F(x))))] \tag{9}
\]

The result of this optimization is a purifier $G$ that will purify $F(x)$ to $G(F(x))$ such that the redundant information useful for the attacker to perform data inferences is largely reduced. Moreover, the classification accuracy loss and the distortion to $F(x)$ is minimized. The time overhead for each query is introduced by the computation of $G$ which is a single forward pass of a much smaller neural network than the original $F$. The time overhead is considered negligible and thus we achieve the efficiency.

5 EXPERIMENTS

In this section, we evaluate our purification framework in defending model inversion attack and membership inference attack. We implement our framework using PyTorch\footnote{https://pytorch.org/}. We also compare our method with existing methods from the defense, utility and efficiency perspectives.

5.1 Experimental Setup

Datasets

We use 3 datasets which are widely adopted in previous works on model inversion [60] and membership inference [24, 47, 49].

CIFAR10 [28, 47, 49]. 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 [28, 35, 47, 49]. This dataset is based on Kaggle’s “acquired valued shopper” challenge\footnote{https://www.kaggle.com/c/acquire-valued-shoppers-challenge/data}. We used the preprocessed and simplified version of this dataset [49]. It is composed of 197,324 data records and each data record has 600 binary features. The dataset is clustered into 100 classes.

FaceScrub530 [60]. This dataset consists of URLs for 100,000 images of 530 individuals. We obtained the preprocessed and simplified version of this dataset from [60] which has 48,579 facial images and each image is resized to $64 \times 64$.

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 attacks, we assume that the attacker has access to a subset $D^A$ of $D_1$ and a subset $D'^A$ of $D_3$. Nonetheless, whether the attacker has knowledge of the membership labels of $D^A$ and $D'^A$ depends on whether the attacker trains shadow models. Specifically, if the attacker trains shadow models, he is assumed to know the membership labels, and use half of $D^A$ and $D'^A$ to train them. The attack model is trained on the whole $D^A$ and $D'^A$ with the shadow models’ training data labeled as members. Otherwise, the attacker is assumed to have the membership labels and can directly query the target classifier to get the confidence scores for members and non-members. In the model inversion attack, for the FaceScrub530 classifier, the attacker uses a CelebA dataset as the auxiliary dataset to train the inversion model, which is the same setting as in [60]. For other classifiers, the attacker samples 80% from $D_1$, $D_2$ and $D_3$ respectively to form the auxiliary dataset and use the other 20% data to test the inversion accuracy.

Target Classifier

For the CIFAR10 dataset, we use the DenseNet121 architecture [22], which is also used in [28, 35]. We train our classifier with Stochastic gradient descent(SGD) optimizer for 350 epochs with 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 L2 regularization(5e-4). For the Purchase100 dataset, we use the same model and the same training strategy as in [35] to train the target classifier. It is a 4-layer fully connected neural network. For the FaceScrub530 dataset, we use the same convolutional neural network and the same training strategy as in [60] to train the target classifier.

Purification Framework

The purification framework consists of the purifier, the adversarial model and the discriminator.

Purifier. We use an autoencoder structure to implement the purifier. It has the layer size $[10, 7, 4, 7, 10]$ for CIFAR10, $[100, 50, 20, 10, 20, 50, 100]$ for Purchase100 and $[530, 200, 530]$ for FaceScrub530. We use the ReLU activation function and batch normalization in every hidden layer of all the purifier models. We train the Purchase100 purifier for 200 epochs and others for 50 epochs. We use
the Adam optimizer with learning rate 0.01 for CIFAR10, 0.0001 for Purchase100 and 0.0005 for FaceScrub530.

Adversarial Model. We use different adversarial models for different datasets. For FaceScrub530, we use the same model as [60], which consists of 5 transposed CNN blocks. For CIFAR10, we reduce 1 transposed CNN blocks from the FaceScrub530 adversarial model and the model consists of 4 transposed CNN blocks. For Purchase100, we use a multi-layer perceptron of size [100, 512, 1024, 600]. Each hidden layer has a ReLU activation function and the output layer has a Sigmoid activation function. We use the Adam optimizer with learning rate 0.0002 for all the adversarial models.

Discriminator. We use a similar structure as the attack model in [35] to build the discriminator. Specifically, it consists of three neural networks. The first neural network operates on the confidence score vector and has the size of \([d, 1024, 512, 64]\) where \(d\) is the input dimension. The second neural network takes the one-hot encoded label as input and has the size of \([d, 512, 64]\). The third network takes the concatenation of the output of the first two networks as input and produces a single value indicating whether the input confidence score vector and label are real or fake. We use the Adam optimizer with learning rate 0.0002 to train the discriminator model.

Data Inference Attacks
In our experiments, we consider the following model inversion attack and membership inference attacks as introduced in Section 2.1. Adversarial model inversion attack [60]. We adopt the recently proposed black-box model inversion attack [60], where the attacker trains an inversion model to infer the reconstruction of the input sample. We use the same model architecture as in [60] to train the inversion model for FaceScrub530 dataset. The inversion model was trained on the CelebA dataset which is the same auxiliary dataset used in [60].

Mleaks attack [47]. This is a confidence-based membership inference attack, where the attacker has no knowledge of the membership labels of \(D_A\) and \(D^A\) and thus has to train a shadow model to replicate the target classifier and then trains the attack model based on the confidence scores of the shadow model. To consider the strongest attack, we assume that the shadow model has the same architecture as the target classifier. We use a multilayer perceptron with a 128-unit hidden layer and a sigmoid output layer to train the attack model. We use the Adam optimizer with learning rate 0.001. The number of training epochs is set to 50 for each dataset.

Mleaks-a attack [47]. This is an adaptive version of the Mleaks attack, where the attacker is assumed to know the defender’s purification framework. Hence, he can train his own purification framework on \(D_2\) to increase the attack accuracy.

NSH attack [35]. This is a confidence & label-based membership inference attack proposed by Nasr, Shokri and Houmansadr [35]. The attacker is assumed to have the knowledge of the membership labels of \(D_A\) and \(D^A\), and thus can directly query the target classifier to get the confidence score vectors of members and non-members without training the shadow model. We use the same attack model as in [35] to implement this attack. During the training of the attack model, we make sure every training batch has the same number of member and non-member instances to prevent the attack model to be biased toward one side as [35] did.

Existing Defenses
We compare our approach with the following existing defenses as introduced in Section 2.1, which represent the state-of-the-art defense methods in the literature.

Min-Max [35]. We use the open-source code of [35] to implement the Min-Max defense. The number of training epochs for both the classification model and the inference model is the same as the number we use to train the target classifier.

MemGuard [24]. We adopt the open-source code of [24] to implement this defense method. Specifically, the defense classifier used in this method consists of three hidden layers [256, 128, 64] and uses ReLU in hidden layers and Sigmoid in the output layer.

Model-Stacking [47]. This method ensembles two layers of models. The first layer combines two models, where we use the same architecture as the target classifier as the first model and use random forest for Purchase100 and VGG19 [50] for FaceScrub530 and CIFAR10 as the second model. The second layer is a logistic regression model for Purchase100 and FaceScrub530, and a neural network with a single hidden layer of size 128 for CIFAR10.

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

Classification Accuracy. It is measured on the training set \(D_1\) and the test set \(D_3\) of the target classifier. It reflects how good the target classifier is at the classification task.

Confidence Score Distortion. We measure the confidence score distortion introduced by a defense method by computing the \(L_1\) norm of the distance between the original confidence score vector predicted by the target classifier and the new confidence score vector after the defense method is applied.

Membership Inference Accuracy. This is the classification accuracy of the attacker’s attack model in predicting the membership of input samples. It is measured on \(D_1 - D^A\) (i.e., members) and \(D_3 - D^A\) (i.e., non-members).

Inversion Error. We measure the inversion error by computing the mean squared error between the original input sample and the reconstructed sample. For the FaceScrub530 classifier, it is measured on \(D_1\) and \(D_3\). For other classifiers, it is measured on the 20% of \(D_1\) and \(D_3\) respectively.

Time Overhead. We measure the efficiency of a defense method by reporting the extra time introduced by deploying and using the defense method. It includes the additional training time of any models introduced by the method and the additional test time when classifying a query data.

5.2 Experimental Results
To compare the performance of our approach and existing approaches, we present the utility, defense performance, efficiency of the original target classifier without any defense in Table 2 (1st row in each dataset).

Defense Goal I: Preventing Model Inversion Attack
We apply the purification framework in Figure 1(a) to mitigate the adversarial model inversion attack against the FaceScrub530
Table 2: Comprehensive results of the utility, defense performance and efficiency of evaluated defense methods.

| Dataset          | Defense          | Utility | Model Inversion | Membership Inference | Efficiency |
|------------------|------------------|---------|-----------------|----------------------|------------|
|                  | Train acc.       | Test acc. | Conf. dist. | Recon. error | NSH | Mlleaks | Mlleaks-a | Train (h) | Test (s) |
| CIFAR10          | None             | 99.99%   | 95.92%        | 0          | 1.4438 | 56.03% | 56.26%    | 0        | 0       |
|                  | Purifier(1,1,1)  | 100.00%  | 95.64%        | 0.1110     | 1.4937 | 52.89% | 51.12%    | 51.72%   | 0.92     | 6.90e-4 |
|                  | Min-Max          | 97.43%   | 88.78%        | 0.3396     | 1.4652 | 55.05% | 52.77%    | 62.71    | 5.88e-4 |
|                  | MemGuard         | 99.99%   | 95.92%        | 0.0418     | 1.4397 | 51.71% | 50.86%    | 4.89     | 2.57    |
|                  | Model-Stacking   | 95.80%   | 92.12%        | 0.2230     | 1.4723 | 51.93% | 51.01%    | 6.44     | 1.38e-3 |
| Purchase100      | None             | 99.96%   | 84.36%        | 0          | 0.1426 | 70.36% | 64.43%    | 0        | 0       |
|                  | Purifier(1,1,1)  | 100.00%  | 84.19%        | 0.1842     | 0.1518 | 58.81% | 51.40%    | 52.74%   | 1.73     | 1.50e-4 |
|                  | Min-Max          | 94.12%   | 79.90%        | 0.4873     | 0.1416 | 56.85% | 55.85%    | 16.00    | 1.80e-5 |
|                  | MemGuard         | 100.00%  | 84.36%        | 0.1444     | 0.1426 | 59.53% | 52.77%    | 0.44     | 2.46    |
|                  | Model-Stacking   | 81.84%   | 69.68%        | 0.8519     | 0.1472 | 61.16% | 55.53%    | 0.09     | 2.70e-4 |
| FaceScrub530     | None             | 100.00%  | 77.68%        | 0          | 0.0114 | 69.34% | 75.04%    | 0        | 0       |
|                  | Purifier(1,1,1)  | 100.00%  | 77.60%        | 0.2558     | 0.0447 | 58.91% | 60.84%    | 60.80%   | 0.92     | 4.25e-3 |
|                  | Min-Max          | 83.43%   | 54.91%        | 0.9821     | 0.0215 | 61.75% | 61.94%    | 1.82     | 1.79e-4 |
|                  | MemGuard         | 100.00%  | 77.68%        | 0.1115     | 0.0115 | 61.97% | 67.29%    | 1.54     | 3.69    |
|                  | Model-Stacking   | 86.30%   | 57.03%        | 1.2635     | 0.0417 | 62.00% | 51.86%    | 1.98     | 9.18e-4 |

Figure 3: Effect of different defenses on defending model inversion attack against the FaceScrub530 classifier. Row 2-6 show the inversion results using our defense approach.

We jointly train the purifier and the adversarial model with $\alpha = 1$ and $\beta = 1$ on $D_1$ and $D_2$ respectively. We choose two architectures $H$ and $H'$ to train the adversarial model. Compared to $H$, $H'$ has an extra transposed convolutional layer with batch normalization and Tanh activation function. Figure 3 (row 1-5) shows the inversion results on the FaceScrub530 dataset with and without defense. We can see that, without defense, the attacker can infer a very accurate reconstruction of the facial images. When the purification framework is applied, the inversion results are more like “average” faces and the useful information about the specific individuals is significantly reduced. Table 4 shows the quantified inversion error and the utility loss introduced by purifier. The inversion error is significantly increased by a factor of almost 4 compared to the inversion error without defense. Besides, the maximum classification accuracy loss is only 0.18% and the average confidence score distortion 0.2743 is considered negligible. From the results, it is clear that our approach can effectively prevent the model inversion attack and introduce negligible utility loss.

**Defense Goal II: Preventing Membership Inference Attack**

The purification framework can be used to prevent the membership inference attack when we jointly train the purifier and the discriminator on the validation dataset $D_2$ as shown in Figure 1(b). We set $\alpha = 1$ and $\beta = 1$. We perform the NSH attack, Mlleaks attack and Mlleaks-a attack against the target classifier with and without our defense. Table 5 presents the results of these attacks against our framework. The classification accuracy is dropped less than 0.3%, and the membership inference accuracy of these attacks is significantly reduced. For instances, the membership inference accuracy of NSH attack is dropped by around 12% for the Purchase100 classifier, by 9% for the FaceScrub530 classifier and to almost 50% for the CIFAR10 classifier. The inference accuracy of Mlleaks attack is dropped by around 15% for the CIFAR10 and FaceScrub530 classifiers and by 12% for the Purchase100 classifier. Interestingly, even though the attacker knows that the target classifier is protected by the purification framework and performs the adaptive attack, i.e., Mlleaks-a attack, the inference accuracy is still equivalent to that of the non-adaptive attack for CIFAR10 and FaceScrub530 classifiers and is increased less than 3% for the Purchase100 classifier. These results demonstrate that our approach can significantly mitigate the membership inference attack with negligible utility loss.

**Defense Goal I & II: Preventing Both Attacks**

To prevent both the mode inversion attack and the membership inference attack, we jointly train the purifier, the adversarial model
Goal I (preventing model inversion attack).

Next, we evaluate the effect of the size of the validation set $D_2$ as shown in Figure 1(c). We present the defense performance against both attacks in Table 2. For all datasets, we set $\alpha = 1$, $\beta = 1$ and $\lambda = 1$. We can see that the classification accuracy, i.e., training accuracy and test accuracy is dropped within 0.3% on the three target classifiers. The introduced confidence distortion is reasonably small, especially when compared with existing defense methods. The defense performance against both attacks is equivalent to the performance when the purification framework is applied to prevent each individual attack. For instance, the inversion error is significantly increased, for example, by a factor of 4 for the FaceScrub530 classifier, which can be visually justified in Figure 3 (6th row), where the attacker cannot infer anything useful about the specific individuals. These results show that the purification framework is effective in defending both the model inversion attack and the membership inference attack with equivalent performance when applied to prevent each individual attack.

Next, we evaluate the effect of the size of the validation set $D_2$ on the defense performance. We use Purchase100 as an illustration example. We vary the size of $D_2$ from 5,000 to 40,000 and present the defense performance of the purification framework in Table 3. We set $\alpha = 1$, $\beta = 1$, $\lambda = 1$ for all $D_2$ sizes. It is worth noting that, when the $D_2$ size increases, both the utility loss and the NSH attack accuracy decrease. This is because more $D_2$ contributes more classification and non-member information to the purifier and the discriminator to learn such that the purified confidence scores behave more like non-member data with less utility loss. However, such effect on the Mleaks attacks is not significant because the attacker has to train a shadow model to replicate the target classifier, which introduces some level of randomness to the attack accuracy. Another finding is that the inversion error slightly decreases as $D_2$ size increases. We believe that it is because the parameters ($\alpha = 1$, $\beta = 1$, $\lambda = 1$) might not fit the larger $D_2$ size. By increasing $\lambda$, the inversion error is expected to increase.

Data Inference and Generalization

We further study the relation between data inference and the generalization ability of the model. We plot the cumulative distribution function of the target classifier’s generalization error over different classes in Figure 4. The curves that lean towards left have a smaller generalization error. We compare the generalization error of the target classifier with and without defense. As it is shown, our approach introduces insignificant change to the generalization error of the target classifier. The same conclusion can be reached in Table 2, where the training accuracy and test accuracy are preserved after a purifier is applied. This result demonstrates that it is possible to effectively mitigate data inference attacks without significantly changing the generalization ability of the target model. Our purification framework, rather than reducing the generalization error of the target classifier (i.e., reducing overfitting) as existing defense methods did, purifies the confidence scores by removing the unwanted redundant information that the attacker could exploit to infer useful information about the query data.

Reduction of Redundant Information in Confidence Scores

The attacker exploits the distinguishable statistical information to infer the membership of a query data [49], and exploits the redundant information that is sensitive to the input sample to infer the reconstruction of the query data [60]. Therefore, we measure how much the two types of information are reduced in the confidence scores by our approach such that the attacker could not infer useful information from them.

We investigate 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 in Figure 6 and plotting the histogram of the prediction uncertainty in Figure 7. The prediction uncertainty is measured as the normalized entropy $\sum_i \hat{y}_i \log(\hat{y}_i)$ of the confidence vector $y = F(x)$, where $k$ is the number of classes. In both figures, the gap between member curves and non-member curves represents the degree of the indistinguishability. The larger the gap is, the more distinguishable the confidence scores on members and non-members are. We report the maximum gap and the average gap between the curves (i.e., without versus with defense) in the following.

- Figure 6 maximum gap: CIFAR10 (0.103 vs. 0.057), Purchase100 (0.412 vs. 0.164) and FaceScrub530 (0.416 vs. 0.264).
- Figure 6 average gap: CIFAR10 (0.004 vs. 0.002), Purchase100 (0.016 vs. 0.007) and FaceScrub530 (0.017 vs. 0.010).
- Figure 7 maximum gap: CIFAR10 (0.114 vs. 0.019), Purchase100 (0.201 vs. 0.058) and FaceScrub530 (0.418 vs. 0.129).

| Table 3: Effect of the size of the validation set $D_2$ on the defense performance of our approach. The numbers are reported on Purchase100 dataset. |
| $D_2$ size | Classification | Conf. dist. | Inver. error | NSH | Mleaks | Mleaks-a |
|------------|----------------|-------------|--------------|-----|--------|----------|
| 5,000 | 69.50% | 0.494224 | 0.152091 | 59.39% | 52.46% | 54.33% |
| 10,000 | 83.36% | 0.195250 | 0.152222 | 59.21% | 52.80% | 54.85% |
| 20,000 | 84.19% | 0.184203 | 0.151805 | 58.81% | 51.40% | 52.74% |
| 40,000 | 84.24% | 0.167419 | 0.151398 | 58.33% | 52.81% | 54.17% |

| Table 4: Inversion results of using different architectures to train the adversarial model on the FaceScrub530 dataset when the purification framework is used to achieve Defense Goal I (preventing model inversion attack). |
| Model | Train Set | Classification Acc. | Conf. dist. | Inversion error |
|-------|---------|---------------------|-------------|----------------|
| None  |         | -                   | -           | -              |
| $H$   | $D_1$   | 77.65%              | 0.289042    | 0.043507       |
| $H'$  | $D_2$   | 77.54%              | 0.256243    | 0.043517       |
| $H''$ | $D_3$   | 77.65%              | 0.276290    | 0.042705       |

| Table 5: Membership inference results on different datasets when the purification framework is used to achieve Defense Goal II (preventing membership inference attack). |
| Dataset | Classification | Conf. dist. | NSH | Mleaks | Mleaks-a |
|---------|----------------|-------------|-----|--------|----------|
| CIFAR10 | 95.64%         | 0.056833    | 52.65% | 51.79% | 52.00%   |
| Purchase100 | 84.10% | 0.170126    | 59.88% | 52.52% | 55.32%   |
| FaceScrub530 | 77.44% | 0.280228    | 60.51% | 59.07% | 60.18%   |
Figure 4: The empirical CDF of the generalization error of the target classifier with and without defense across different classes. The generalization error is measured as the difference between the training and test accuracy of the target classifier. The y-axis is the fraction of classes that have less generalization error than x-axis.

Figure 5: Distribution of the confidence score vectors of the target classifier on the training data and test data of class 29 in the Purchase100 dataset. Each color represents one data record.

- Figure 7 average gap: CIFAR10 (0.005 vs. 0.002), Purchase100 (0.015 vs. 0.003) and FaceScrub530 (0.017 vs. 0.005).

These results quantitatively and visibly show that our approach can significantly reduce both maximum and average gaps between the target classifier’s confidence in the correct class and prediction uncertainty on members versus non-members. This, to some extent, improves the indistinguishability of confidence scores.

Figure 5 visibly presents the confidence score vectors of the target classifier on the training and test data from class 29 without and with our defense. Without defense, the target classifier produces a very high probability for class 29 on the training data. On the test data, besides class 29, the probabilities for class 54 and 70 are also similarly high. This is where the target classifier makes most mistakes on the test data. Besides, the target classifier spreads the prediction probability across many classes which means the prediction is sensitive to the input data. When our defense is applied, the prediction probabilities on training data and test data show similar patterns. For example, besides class 29, they are both high on class 15, 24, 35 and 38. The probability of the test data is still high on class 54 and 70 which is in line with our finding that our defense introduces negligible classification accuracy loss. On the other hand, by reshaping the confidence scores to concentrate on a
When the attacker performs model inversion attack, our approach introduces negligible utility loss over the three datasets. Our approach makes them less sensitive to input data, which is helpful in mitigating the model inversion attack.

### 5.3 Comparison with Existing Defenses

Table 2 presents the comparison of different defense methods from the utility, defense performance and efficiency aspects. We didn’t perform the model inversion attack against the FaceScrub530 classifier protected by MemGuard because it took too long to train the inversion model on the auxiliary CelebA dataset in limited time.

As it is shown, our approach introduces negligible utility loss to all the three target classifiers, which outperforms Min-Max and Model-Stacking. MemGuard is designed to guarantee no classification accuracy loss and minimal confidence score distortion. Therefore, it slightly outperforms our approach in terms of utility loss. When the attacker performs model inversion attack, our approach outperforms all the other defense methods with the largest inversion error. Figure 3 (row 6-8) shows the inversion results against the FaceScrub530 classifier protected by different defense methods. We can see that the reconstructed facial images against our approach are more vague with detailed facial features not recovered. For membership inference attacks, our approach outperforms other approaches in defending NSH attack on the FaceScrub530 dataset and defending Mleaks attack on the Purchase100 dataset. In other attacks, the defense performance of our approach is comparable with the best result among other approaches.

**Efficiency.** In our experiments, we train models on a PC equipped with four Titan XP GPUs with 12GBBytes of graphic memory, 128 GBytes of memory and an Intel Xeon E5-2678 CPU. We compute the efficiency of different defense methods by measuring the training time overhead and the test time overhead. Table 2 shows the comparison results. Our approach achieves significantly less training time than other defense methods on the CIFAR10 and FaceScrub530 classifiers, which are relatively large and complicated neural networks and thus the efficiency of our approach has practical meaning. The test time overhead of our approach is introduced by the computation of a single forward pass of the purifier model. The average overhead 1.70e-3 seconds per sample over the three datasets is considered insignificant.

### 6 RELATED WORK

**Inference Attacks.** The inference attacks against machine learning can be divided into model inference and data inference attacks. In model inference attacks, an attacker could infer the parameters [54], hyper-parameters [55], architecture [37] and functionality [40] of a target model. In data inference attacks, the attacker aims at inferring information about the data that the target model operates on. These attacks include membership inference attack [49], model inversion attack (input inference) [14, 60], attribute inference [15, 58], statistics inference [2] and side-channel attack [57].

In this paper, we concentrate on data inference attacks, notably membership inference attack and model inversion attack. Existing defense methods mainly focused on membership inference attacks [24, 28, 35, 49]. Little has been studied about the model inversion attack. Xiao et al. [59] studied the adversarial reconstruction problem where they aim 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 classification models whose output is constrained by a probability distribution wherein the 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.

**General Membership Inference Attack.** Membership inference attack is performed to determine whether a given data sample is part of a target dataset. It is not limited to machine learning models. Homer et al. [21] 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 [3] and DNA methylation [17]. Pyrgelis et al. [45, 46] further showed that it is possible to perform membership inference attack on location datasets as well. Shokri et al. [49] performed membership inference attack in the machine learning setting which is the same setting in this paper.

**Secure & Privacy-Preserving Machine Learning.** The untrusted access of machine learning models in the training or testing phase is a direct threat to the security and privacy of machine learning. A number of studies made use of trusted hardware and...
We propose a purification framework to defend data inference attacks against machine learning including the model inversion attack and the membership inference attack. The framework purifies a dataset which could be exploited by various inference attacks. In our extensive experiments, we show that the purification framework is effective in mitigating data inference attacks and imposes negligible utility loss.

7 CONCLUSION

We propose a purification framework to defend data inference attacks against machine learning including the model inversion attack and the membership inference attack. The framework purifies a dataset which could be exploited by various inference attacks. In our extensive experiments, we show that the purification framework is effective in mitigating data inference attacks and imposes negligible utility loss.

REFERENCES

[1] Martin Abadi, Andy Chu, Ian J Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. 2016. Deep Learning with Differential Privacy. In Proceedings of the 2016 [ACM] SIGSAC Conference on Computer and Communications Security, Vienna, Austria, October 24-28, 2016. 308–318. https://doi.org/10.1145/2976749.2978318

[2] Giuseppe Ateniese, Luigi V Mancini, Angelo Spognardi, Antonio Villani, Domenico Vitali, and Giovanni Felici. 2015. Hacking Smart Machines with Smarter Ones: How to Extract Meaningful Data from Machine Learning Classifiers. International Journal of Security and Networks 10, 3 (Sept. 2015), 137–150. https://doi.org/10.1504/IJSN.2015.071829

[3] Michael Backes, Pascal Berrang, Mathias Humbert, and Praveen Manoharan. 2016. Membership Privacy in Micro-RNA-Based Studies. In Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (CCS ’16). Association for Computing Machinery, Vienna, Austria, 319–330. https://doi.org/10.1145/2976749.2978355

[4] Pierre Baldi. 2012. Autoencoders, Unsupervised Learning, and Deep Architectures. In Proceedings of ICMC Workshop on Unsupervised and Transfer Learning, 37–49.

[5] Raf Baskil, Adam Smith, and Abhradeep Thakurta. 2014. Private empirical risk minimization: Efficient algorithms and tight error bounds. In Proceedings - Annual IEEE Symposium on Foundations of Computer Science, FOCS (Proceedings - Annual IEEE Symposium on Foundations of Computer Science, FOCS). IEEE Computer Society, United States, 464–473. https://doi.org/10.1109/FOCS.2014.56

[6] Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H. Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. 2017. Practical Secure Aggregation for Privacy-Preserving Machine Learning. In Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security - CCS ’17. 1175–1191. https://doi.org/10.1145/3133956.3133982

[7] Rich Caruana, Steve Lawrence, and C Lee Giles. 2001. Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping. In Advances in neural information processing systems. 402–408.

[8] Kamalika Chaudhuri, Claire Monteleoni, and Anand D Sarwate. 2011. Differentially Private Empirical Risk Minimization. Journal of Machine Learning Research 12, Mar (2011), 1069–1109.

[9] Ambra Demontis, Marco Melis, Maura Pintor, Mathew Jagielski, Shuichi Katsumata, Shinsaku Kiyomoto, and Giovanni Felici. 2015. Hacking Smart Machines with Smarter Ones: How to Extract Meaningful Data from Machine Learning Classifiers. International Journal of Security and Networks 10, 3 (Sept. 2015), 137–150. https://doi.org/10.1504/IJSN.2015.071829

[10] Nathan Dowlon, Ran Gilad-Bachrach, Kim Laine, Kristin Lauter, Michael Naehrig, and John Wernsing. 2016. 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 (ICML ’16). JMLR.org, New York, NY, USA, 201–210.

[11] Vincent Dumoulin, Ishmael Belghazi, Ben Poole, Alex Lamb, Martin Arjovsky, Olivier Mastropietro, and Aaron C Courville. 2017. Adversarially Learned Inference. In ICLR 2017, Vol. abc1066.0.
