Inf²Guard: An Information-Theoretic Framework for Learning Privacy-Preserving Representations against Inference Attacks

Sayedeh Leila Noorbakhsh and Binghui Zhang, Illinois Institute of Technology; Yuan Hong, University of Connecticut; Binghui Wang, Illinois Institute of Technology

https://www.usenix.org/conference/usenixsecurity24/presentation/noorbakhsh
Inf²Guard: An Information-Theoretic Framework for Learning Privacy-Preserving Representations against Inference Attacks

Sayeedeh Leila Noorbakhsh¹,*, Binghui Zhang¹,*, Yuan Hong², Binghui Wang¹
¹Illinois Institute of Technology, ²University of Connecticut, *Equal contribution

Abstract

Machine learning (ML) is vulnerable to inference (e.g., membership inference, property inference, and data reconstruction) attacks that aim to infer the private information of training data or dataset. Existing defenses are only designed for one specific type of attack and sacrifice significant utility or are soon broken by adaptive attacks. We address these limitations by proposing an information-theoretic defense framework, called Inf²Guard, against the three major types of inference attacks. Our framework, inspired by the success of representation learning, posits that learning shared representations not only saves time/costs but also benefits numerous downstream tasks. Generally, Inf²Guard involves two mutual information objectives, for privacy protection and utility preservation, respectively. Inf²Guard exhibits many merits: it facilitates the design of customized objectives against the specific inference attack; it provides a general defense framework which can treat certain existing defenses as special cases; and importantly, it aids in deriving theoretical results, e.g., inherent utility-privacy tradeoff and guaranteed privacy leakage. Extensive evaluations validate the effectiveness of Inf²Guard for learning privacy-preserving representations against inference attacks and demonstrate the superiority over the baselines.¹

1 Introduction

Machine learning (ML) models (particularly deep neural networks) are vulnerable to inference attacks, which aim to infer sensitive information about the training data/dataset that are used to train the models. There are three well-known types of inference attacks on training data/dataset: membership inference attacks (MIAs) [12, 58, 73], property inference attacks (PIAs) (also called distribution inference attacks) [6, 20, 63], and data reconstruction attacks (DRAs) (also called model inversion attacks) [7, 28]. Given an ML model, in MIAs, an adversary aims to infer whether a particular data sample was in the training set, while in PIAs, an adversary aims to infer statistical properties of the training dataset used to train the targeted ML model. Furthermore, an adversary aims to directly reconstruct the training data in DRAs. Leaking the data sample or information about the dataset raises serious privacy issues. For instance, by performing MIAs, an adversary is able to identify users included in sensitive medical datasets, which itself is a privacy violation [30]. By performing PIAs, an adversary can determine whether or not machines that generated the bitcoin logs were patched for Meltdown and Spectre attacks [20]. More seriously, DRAs performed by an adversary leak all the information about the training data.

To mitigate the privacy risks, various defenses have been proposed against MIAs [35, 45, 54, 56, 58, 59, 61, 71] and DRAs [21, 25, 38, 48, 55, 62, 69, 81]². However, there are two fundamental limitations in existing defenses: 1) They are designed against only one specific type of attack; 2) Provable defenses (based on differential privacy [3, 18]) incur significant utility losses to achieve reasonable defense performance against inference attacks [33, 56] since the design of such randomization-based defenses did not consider specific inference attacks (also see Section 5); and empirical defenses are soon broken by stronger/adaptive attacks [9, 16, 59].

We aim to address these limitations and consider the question: 1) Can we design a unified privacy protection framework against these inference attacks, that also maintain utility? 2) Under the framework, can we further theoretically understand the utility-privacy tradeoff and the privacy leakage against the inference attacks? To this end, we propose an information-theoretic defense framework, termed Inf²Guard, against inference attacks through the lens of representation learning [11]. Representation learning has been one of the biggest successes in modern ML/AI so far (e.g., it plays an important role in today’s large language models such as ChatGPT [1] and PaLM2 [2]). Particularly, rather than training large models from scratch, which requires huge computational

¹Source code and the full version at: https://github.com/leilynoorbakhsh/Inf2Guard.

²To our best knowledge, there exist no effective defenses against PIAs. [26] analyzes sources of information leakage to cause PIAs, but their solutions are difficult to be tested on real-world datasets due to lack of generality.
costs and time (e.g., GPT-3 has 175 billion parameters), learning shared representations (or pretrained encoder)\(^3\) presents an economical alternative. For instance, the shared representations can be directly used or further fine-tuned with different purposes, achieving considerable savings in time and cost.

More specifically, we formulate Inf\(^2\)Guard via two mutual information (MI)\(^4\) objectives in general, for privacy protection and utility preservation, respectively. Under this framework, we can design customized MI objectives to defend against each inference attack. For instance, to defend against MIAs, we design one MI objective to learn representations that contain as less information as possible about the membership of the training data—thus protecting membership privacy, while the other one to ensure the learnt representations include as much information as possible about the training data labels—thus maintaining utility. However, directly solving the MI objectives for each inference attack is challenging, since calculating an MI between arbitrary variables is often infeasible [49]. To address it, we are inspired by the MI neural estimation [4, 10, 15, 29, 47, 50], which transfers the intractable MI calculations to the tractable variational MI bounds. Then, we are capable of parameterizing each bound with a (deep) neural network, and train neural networks to approximate the true MI and learn representations against the inference attacks. Finally, we can derive theoretical results based on our MI objectives: we obtain an inherent utility-privacy tradeoff, and guaranteed privacy leakage against each inference attack.

We extensively evaluate Inf\(^2\)Guard and compare it with the existing defenses against the inference attacks on multiple benchmark datasets. Our experimental results validate that Inf\(^2\)Guard obtains a promising utility-privacy tradeoff and significantly outperforms the existing defenses. For instance, under the same defense performance against MIAs, Inf\(^2\)Guard has a 30% higher testing accuracy than the DP-SGD [3]. Our results also validate the privacy-utility tradeoffs obtained by Inf\(^2\)Guard\(^5\).

Our main contributions are summarized as below:

- **Algorithm**: We design the first unified framework Inf\(^2\)Guard to defend against the three well-known types of inference attacks via information theory. Our framework can instantiate many existing defenses as special cases, e.g., AdvReg [45] against MIAs (See Section 3.1) and Soteria [62] against DRAs (See Section 3.3).

- **Theory**: Based on our formulation, we can derive novel theoretical results, e.g., the inherent tradeoff between utility and privacy, and guaranteed privacy leakage against all the considered inference attacks.

- **Evaluation**: Extensive evaluations verify the effectiveness of Inf\(^2\)Guard for learning privacy-preserving representations against inference attacks.

2 Background and Problem Definition

**Notations**: We use \(s, s, S, \) and \(S\) to denote (random) scalar, vector, matrix, and space, respectively. Accordingly, \(Pr(s), \) \(Pr(s), \) and \(Pr(S)\) are the probability distribution over \(s, s,\) and \(S, I(x;r)\) and \(H(x,r)\) are the mutual information and cross entropy between a pair of random variables \((x,r)\), respectively, and \(H(x) = I(x;x)\) as the entropy of \(x, KL(p||q)\) is the KL-divergence between two distributions \(p\) and \(q\).

We denote \(D\) as the underlying distribution that data are sampled from. A data sample is denoted as \((x,y) \sim D, \) where \(x\) is data features, \(y \in Y\) is the label, and \(X\) and \(Y\) are the data space and label space, respectively. We further denote a dataset as \(D = \{(x_i,y_i)\},\) that consists of a set of data samples \((x_i,y_i) \sim D,\) and will interchangeably use \(D\) and \(\{X,Y\}.\) We let \(u \in U\) be the private attribute within the attribute space \(\mathcal{U}.\) For instance, in MIAs, \(u \in \mathcal{U} = \{0,1\}\) means a binary-valued private membership; in PIAs, \(u \in \mathcal{U} = \{1,2,\cdots,K\}\) indicates a \(K\)-valued private dataset property; and \(u \in \mathcal{U} = X\) indicates the private data itself in DRAs. The composition function of two functions \(f\) and \(g\) is denoted as \((g \circ f)(\cdot) = g(f(\cdot)).\)

2.1 Formalizing Privacy Attacks

We denote a classification model\(^6\) \(F_\theta : X \rightarrow Y\) as a function, parameterized by \(\theta,\) that maps a data sample \(x \in X\) to a label \(y \in Y.\) Given a training set \(D \sim D,\) we denote \(F \leftarrow \mathcal{T}(D)\) as learned by running a training algorithm \(\mathcal{T}\) on the dataset \(D.\)

**Formalizing MIAs**: Assume a data sample \((x,y) \sim \mathcal{D}\) with a private membership \(u\) that is chosen uniformly at random from \(\{0,1\},\) where \(u = 1\) means \((x,y)\) is a member of \(D,\) and 0 otherwise. An MIA \(A_{MIA}\) has access to \(\mathcal{D}\) and \(F,\) takes \((x,y)\) as input, and outputs a binary \(A_{MIA}^{D,F}(x,y)\). We omit \(\mathcal{D},F\) for notation simplicity. Then, the attack performance of an MIA \(A_{MIA}\) is defined as \(Pr_{(x,y)}(A_{MIA}(x,y) = u).\)

**Formalizing PIAs**: PIAs define a private property on a dataset. Given a dataset \(D_u \sim \mathcal{D}\) with a private property \(u\) chosen uniformly at random from \(\{1,2,\cdots,K\}.\) A PIA \(A_{PIA}\) has access to \(\mathcal{D}\) and \(F,\) and outputs a \(K\)-valued \(A_{PIA}(D_u)\). Then, the attack performance of a PIA \(A_{PIA}\) is defined as \(Pr_{(D_u)}(A_{PIA}(D_u) = u).\)

**Formalizing DRAs**: Given a random data \((x,y) \in \mathcal{D},\) DRAs aim to reconstruct the private \(x.\) A DRA \(A_{DRA}\) has access to \(\mathcal{D}\) and \(F,\) and outputs a reconstructed \(\hat{x} = A_{DRA}(x,y).\) The DRA performance is measured by the similarity/difference between \(\hat{x}\) and \(x.\) For instance, [7] introduces the \((\eta,\gamma)-\)reconstruction metric defined as \(Pr_{(x,y)}(\|\hat{x} - x\|_2 \leq \eta) \geq \gamma,\) where a smaller \(\eta\) and a larger \(\gamma\) imply a more severe DRA.

\(^3\)Pretrained encoder as a service has been widely deployed by industry, e.g., OpenAI’s GPT-4 API [1] and Clarifai’s Embedding API [17]. We will interchangeably use the pretrained encoder and learnt representations.

\(^4\)In information theory, MI is a measure of shared information between random variables, and offers a metric to quantify the “amount of information” obtained about one random variable by observing the other random variable.

\(^5\)A recent work [53] formulates defenses against inference attacks under a privacy game framework, but it does not propose concrete defense solutions.

\(^6\)In this paper, we focus on classification models for simplicity.
2.2 Threat Model and Problem Formulation

We have three roles: task learner, defender, and attacker. The task learner (i.e., data owner) aims to learn an accurate classification model on its training data. The defender (e.g., data owner or a trusted service provider) aims to protect the training data privacy—it designs a defense framework by learning shared data representations that are robust against inference attacks. The attacker can arbitrarily use data representations to perform the inference attack. The attacker is also assumed to know the underlying data distribution, but cannot access the internal encoder (e.g., deployed as an API [1, 17]).

Formally, we denote \( f_\Theta : X \to Z \) as the encoder, parameterized by \( \Theta \), that maps a data sample \( x \in X \) (or a dataset \( X \in X \)) to its representation vector \( r = f(x) \in Z \) (or representation matrix \( R = f(X) \in Z \)), where \( Z \) is the representation space. Moreover, we let \( C : Z \to Y \) be the classification model on top of the representation \( r \) or encoder \( f \), which predicts the data label \( y \) (or dataset labels \( y \)). We further let \( A : Z \to U \) be the inference model, which infers the private attribute \( u \) using the learnt representations \( r \) or \( R \). Then, our defense goals are:

- **Defend against MIAs:** Given a random sample \((x, y, u) \in D \), we expect to learn \( f \) such that the MIA performance \( \Pr(A_{\text{IA}}(f(x), y) = u) \) is low, and the utility loss/risk, i.e., \( \text{Risk}_{\text{MIA}}(C \circ f) = \Pr(C \circ f(x) \neq y) \), is also small.

- **Defend against PIAs:** Given a random dataset \((X, y, u) \in D \), we expect to learn \( f \) with low PIA performance \( \Pr(A_{\text{PIA}}(f(X), y) = u) \), and also a small utility loss/risk, i.e., \( \text{Risk}_{\text{PIA}}(C \circ f) = \frac{1}{|Y|} \sum_{(x,y) \in X \times Y} \Pr(C \circ f(x) \neq y) \).

- **Defend against DRAs:** Given a random sample \((x, y) \in D \), we expect to learn \( f \) with low DRA performance, i.e., \( \Pr(A_{\text{DRA}}(f(x), y) = y) \geq \gamma \) with a large \( \gamma \) (flipping the inequality direction on \( \eta \) for DRAs), and also a small utility risk \( \text{Risk}_{\text{DRA}}(C \circ f) = \Pr(C \circ f(x) \neq y) \).

3 Design of Inf\(^2\)Guard

3.1 Inf\(^2\)Guard against MIAs

3.1.1 MI objectives

Given a data sample \( x \sim D \), from the training set \( D \) (i.e., \( u = 1 \)) or not (i.e., \( u = 0 \)), the defender learns the representation \( r = f(x) \) that satisfies the following two goals:

- **Goal 1: Membership protection.** \( r \) contains as less information as possible about the private membership \( u \). Ideally, when \( r \) does not include information about \( u \) (i.e., \( r \perp u \)), it is impossible to infer \( u \) from \( r \). Formally, we quantify the membership protection using the MI objective as follows:

  \[
  \min_f \Phi_f(r; u),
  \]

  where we minimize such MI to maximally reduce the correlation between \( r \) and \( u \).

- **Goal 2: Utility preservation.** \( r \) should be effective for predicting the label \( y \) of the training data (i.e., \( u = 1 \)), thus preserving utility. Formally, we quantify the utility preservation using the below MI objective:

  \[
  \max_f I(y; r | u = 1),
  \]

  where we maximize such MI to make \( r \) accurately predict the training data label \( y \) during training.

3.1.2 Estimating MI via tractable bounds

The key challenge of solving the above two MI objectives is that calculating an MI between two arbitrary random variables is likely to be infeasible [49]. Inspired by the existing MI neural estimation methods [4, 10, 15, 29, 47, 50], we convert the intractable exact MI calculations to the tractable variational MI bounds. Specifically, we first obtain an MI upper bound for membership protection and an MI lower bound for utility preserving via introducing two auxiliary posterior distributions, respectively. Then, we parameterize each auxiliary distribution with a neural network, and approximate the true MI by minimizing the upper bound and maximizing the lower bound through training the involved neural networks. We emphasize we do not design novel MI neural estimators, but adopt existing ones to assist our MI objectives for learning privacy-preserving representations. Note that, though the estimated MI bounds may not be tight (due to the MI estimators or auxiliary distributions learnt by neural networks) [15, 29], they have shown promising performance in practice. It is still an active research topic to design better MI estimators that lead to tighter MI bounds (which is orthogonal to this work).

Minimizing the upper bound MI in Equation (1). We adapt the variational upper bound proposed in [15]. Specifically, \( I(r; u) \leq I_{\text{KL}}(r; u) = \mathbb{E}_{p(r|u)} \log q(u|r) - \mathbb{E}_{p(r|u)} \log q(u|r) \), where \( q(u|r) \) is an auxiliary posterior distribution of \( p(u|r) \) needing to satisfy the below condition on KL divergence: \( KL(p(r|u)||q(u|r)) \leq KL(p(r)p(u)||q(u|r)) \). To achieve this, we thus minimize:

\[
\min_{\Psi} KL(p(r|u)||q(u|r)) = \min_{\Psi} KL(p(r|u)||q(u|r))
\]

\[
= \min_{\Psi} \mathbb{E}_{p(r|u)} \left[ \log q(u|r) - \mathbb{E}_{p(r|u)} \log q(u|r) \right]
\]

\[
\iff \max_{\Psi} \mathbb{E}_{p(r|u)} \left[ \log q(u|r) \right],
\]

where we note that \( \mathbb{E}_{p(r|u)} \log q(u|r) \) is irrelevant to \( \Psi \). [15] proved when \( q(u|r) \) is parameterized by a neural network with high expressiveness (e.g., deep neural network), the condition is satisfied almost surely by maximizing Equation (3). Finally, our Goal 1 for privacy protection is reformulated as solving the below \( \min\)-\( \max \) objective function:

\[
\min_f \Phi_f(r; u) \iff \min_{\Psi} \max_f \mathbb{E}_{p(r|u)} \left[ \log q(u|r) \right]
\]
Remark. Equation (4) can be interpreted as an adversarial game between an adversary \( q_{\Psi} \) (i.e., a membership inference classifier) who aims to infer the membership \( u \) from \( r \); and the encoder \( f \) who aims to protect \( u \) from being inferred.

Maximizing the lower bound MI in Equation (2). We adopt the MI estimator proposed in [46] to estimate the lower bound of Equation (2). Specifically, we have

\[
\begin{align*}
I(y; r | u = 1) &= H(y | u = 1) - H(y | r, u = 1) \\
&= H(y | u = 1) + \sum_{p(x,u)} \log p(y | r, u = 1) \\
&= H(y | u = 1) + \sum_{p(x,u)} \log q_{\Omega}(y | r, u = 1) \\
&\quad + \sum_{p(x,u)} [KL(p(y | r, u = 1) | q_{\Omega}(y | r, u = 1))],
\end{align*}
\]

where \( q_{\Omega} \) is an arbitrary auxiliary posterior distribution that aims to accurately predict the training data label \( y \) from the representation \( r \). Hence, our Goal 2 for utility preservation can be rewritten as the following: the max-max objective function:

\[
\max_f \min_{\Omega} \max x \log q_{\Omega}(y | r, u = 1) \quad (5)
\]

Remark. Equation (5) can be interpreted as a cooperative game between the encoder \( f \) and \( q_{\Omega} \) (e.g., a label predictor) that aims to preserve the utility collaboratively.

**Objective function of Inf\(^2\)Guard against MIAs.** By combining Equations (4) and (5), our objective function of learning privacy-preserving representations against MIAs is:

\[
\max_f \left( \lambda \min_{\Psi} \sum_{p(x,u)} \log q_{\Psi}(u | f(x)) \right) \\
+ (1 - \lambda) \max_{\Omega} \sum_{p(x,u)} \log q_{\Omega}(y | f(x), u = 1),
\]

(6)

where \( \lambda \in [0, 1] \) tradeoffs privacy and utility. That is, a larger \( \lambda \) indicates a stronger membership privacy protection, while a smaller \( \lambda \) indicates a better utility preservation.

### 3.1.3 Implementation in practice

In practice, we solve Equation (6) via training three parameterized neural networks (i.e., encoder \( f \), membership protection network \( g_{\Psi} \) associated with the posterior distribution \( q_{\Psi} \), and utility preservation network \( h_{\Omega} \) associated with the posterior distribution \( q_{\Omega} \)) using data samples from the underlying data distribution. Specifically, we first collect two datasets \( D_1 \) and \( D_0 \) from a larger dataset, and they include the members and non-members, respectively. Then, \( D_1 \) is used for training the utility network \( h_{\Omega} \) (i.e., predicting labels for training data \( D_1 \)) and the encoder \( f \); and both \( D_1 \) and \( D_0 \) are used for training the membership protection network \( g_{\Psi} \) (i.e., inferring whether a data sample from \( D_1/D_0 \) is a member or not) and the encoder \( f \). With it, we can approximate the expectation terms in Equation (6) and use them to train the neural networks.

**Training the membership protection network** \( g_{\Psi} \): We approximate the first expectation w.r.t. \( q_{\Psi} \) as

\[
\sum_{p(x,u)} \log q_{\Psi}(u | f(x)) \approx \sum_{(x_i, u_i) \in D_1 \cup D_0} H(u_i | g_{\Psi}(f(x_i))),
\]

where \( H(a,b) \) is the cross-entropy loss between \( a \) and \( b \). Take a single data \( x \) with private \( u \) for example. The above equation is obtained by: \( -H(u, g_{\Psi}(f(x))) = \log g_{\Psi}(u | f(x)) \), where \( g_{\Psi}(f(x)) \) indicates \( i \)-th entry probability, and \( q_{\Psi}(u | f(x)) \) means the probability of inferring \( x \)’s member \( u \). The adversary maximizes this expectation aiming to enhance the membership inference performance.

**Training the utility preservation network** \( h_{\Omega} \): We approximate the second expectation w.r.t. \( q_{\Omega} \) as:

\[
\sum_{p(x,u)} \log q_{\Omega}(y | f(x), u = 1) \approx \sum_{(x_i, y_i) \in D_1} H(y_i, h_{\Omega}(f(x_i))).
\]

We maximize this expectation to enhance the utility.

**Training the encoder** \( f \): With the updated \( g_{\Psi} \) and \( h_{\Omega} \), the defender performs gradient ascent on Equation (6) to update \( f \), which can learn representations that protect membership privacy and further enhance the utility.

We iteratively train the three networks until reaching predefined maximum rounds. Figure 1 illustrates our Inf\(^2\)Guard against MIAs. Algorithm 1 in Appendix details the training.

**Connection with AdvReg [45]**. We observe that AdvReg is a special case of Inf\(^2\)Guard. Specifically, the objective function of AdvReg can be rewritten as:

\[
\max_f \left( \lambda \min_{\Psi} \sum_{(x_i, u_i) \in D_1 \cup D_0} H(u_i, g_{\Psi}(f(x_i))) \right) \\
- \lambda \sum_{(x_i, y_i) \in D_1} H(y_i, h_{\Omega}(f(x_i))),
\]

where \( f : X \to [0, 1]^{|Y|} \) now outputs a sample’s probabilistic confidence score and \( g_{\Psi} \) is a membership inference model aiming to distinguish between members and non-members.

### 3.2 Inf\(^2\)Guard against PIAs

Different from MIAs, PIAs leak the training data properties at the dataset-level. To align this, instead of using a random sample \( (x, y) \), we consider a random dataset \( (X, y) \) in PIAs. Specifically, let \( X = \{x_i\} \) consist of a set of independent data samples and \( y = \{y_i\} \) the corresponding data labels that are sampled from the underlying data distribution \( D \); and \( X \) is associated with a private (dataset) property \( u \).

\[^7\]We omit the sample size \( |D_1|, |D_0| \) for description brevity.
3.2.1 MI objectives

Given a dataset \( X \sim \mathcal{D} \) with a property \( u \), the defender learns a dataset representation \( R = f(X) \) that satisfies two goals\(^8\):

- **Goal 1: Property protection.** \( R \) contains as less information as possible about the private dataset property \( u \). Ideally, when \( R \) does not include information about \( u \) (i.e., \( R \perp u \)), it is impossible to infer \( u \) from \( R \). Formally, we quantify the property protection using the below MI objective:

\[
\min_f I(R;u). 
\tag{7}
\]

- **Goal 2: Utility preservation.** \( R \) includes as much information as possible about predicting \( y \). Formally, we quantify the utility preservation using the MI objective as below:

\[
\max_f I(y;R). 
\tag{8}
\]

3.2.2 Estimating MI via tractable bounds

We estimate the bounds of Equations 7 and 8 as below.

**Minimizing the upper bound MI in Equation (7).** Following membership protection, **Goal 1** is reformulated as solving the below min-max objective function:

\[
f \approx \min \max_q \mathbb{E}_{p(R,u)} \left[ \log q(u|R) \right], \tag{9}
\]

where \( q(u|R) \) is a posterior distribution.

**Remark.** Similarly, Equation (9) can be interpreted as an adversarial game between a property inference adversary \( q \) who aims to infer \( u \) from the dataset representations \( R \) and the encoder \( f \) who aims to protect \( u \) from being inferred.

**Maximizing the lower bound MI in Equations (8).** Similarly, we adopt the MI estimator [46] to estimate the lower bound MI in our **Goal 2**, which can be rewritten as the following max-min objective function:

\[
f \approx \max \mathbb{E}_{p(R,y)} \left[ \log q_R(y|R) \right], \tag{10}
\]

where \( q_R(y|R) \) is an arbitrary posterior distribution that aims to predict each label \( y \in \mathcal{Y} \) from the data representation \( R \) and encoder \( f \) to preserve the utility collaboratively.

**Objective function of Inf\(^2\)Guard against PIAs.** By combining Equations (9) and (10), our objective function of learning privacy-preserving representations against PIAs is:

\[
\max_f \left( \lambda \min_p \mathbb{E}_{p(X,u)} \left[ \log q(u|f(X)) \right] + (1-\lambda) \max_{p(X,Y)} \mathbb{E}_{p(X,Y)} \left[ \log q(y|f(X)) \right] \right), \tag{11}
\]

where \( \lambda \in [0, 1] \) tradeoffs between privacy and utility. That is, a larger/smaller \( \lambda \) indicates less/more dataset property can be inferred through the learnt dataset representation.

\(^8\)For notation simplicity, we use the same \( f \) to indicate the encoder. Similar for subsequent notations such as \( g, h, q, h_2, h_3 \), etc.

3.2.3 Implementation in practice

Equation (11) is solved via three parameterized neural networks (i.e., the encoder \( f_0 \), the property protection network \( g_\Psi \) associated with \( q_\Psi \), and the utility preservation network \( h_2 \) associated with \( q_2 \)) using a set of datasets sampled from a data distribution. Specifically, we first collect a large reference dataset \( D_r \). Then, we randomly generate a set of small datasets \( \{D_j = \{X_j,y_j\}\}_j \) from \( D_r \). We denote the dataset property value for each \( D_j \) as \( u_j \). With it, we can approximate the expectation terms in Equation (11).

**Training the property inference network \( g_\Psi \):** We approximate the first expectation w.r.t. \( q_\Psi \) as

\[
\mathbb{E}_{p(X,u)} \log q_\Psi(u|f(X)) \approx - \sum_{\{X_j,y_j\}_j} H(u_j,g_\Psi(f(X_j))), \tag{12}
\]

where \( f(X_j) \) is the aggregated representation of a dataset \( X_j \), i.e., \( f(X_j) = \text{Agg}(\{f(x)\}_{x \in X_j}) \). We will discuss the aggregator \( \text{Agg}(\cdot) \) in Section 5.2.2. The adversary maximizes this expectation to enhance the property inference performance.

**Training the property protection network \( h_2 \):** Similarly, we approximate the second expectation w.r.t. \( q_2 \) as:

\[
\mathbb{E}_{p(X,Y)} \log q_2(y|f(X)) \approx - \sum_{\{X_j,y_j\}_j} \sum_{(x_i,y_i) \in D_j} H(y_i,h_2(f(x_i))), \tag{13}
\]

where we maximize this expectation to enhance the utility.

**Training the encoder \( f \):** The defender then performs gradient ascent on Equation (11) to update \( f \), which mitigates the PIA and further enhances the utility.

We iteratively train the three networks until reaching maximum rounds. Figure 2 illustrates our Inf\(^2\)Guard against PIAs. Algorithm 2 in Appendix details the training process.

![Inf\(^2\)Guard against PIAs.](image)

3.3 Inf\(^2\)Guard against DRAs

Different from MIAs and PIAs, DRAs aim to *directly* recover the training data from the learnt representations. A recent defense [62] shows perturbing the latent representations can somewhat protect the data from being reconstructed. However, this defense is broken by an advanced attack [9]. One key reason is the defense perturbs representations in a deterministic fashion for already trained models. We address the issues and propose an information-theoretic defense to learn...
randomized representations against the DRAs in an end-to-end learning fashion. Our core idea is to learn a deterministic encoder and a randomized perturbator that ensures learning the perturbed representation in a controllable manner.

3.3.1 MI objectives

Given a data sample \( x \sim D \) with a label \( y \), the defender learns a representation \( r = f(x) \) such that when \( r \) is perturbed by a certain perturbation (denoted as \( \delta \)), the shared perturbed representation \( r + \delta \) cannot be used to well recover \( x \), but is effective for predicting \( y \), from the information-theoretic perspective. Then we aim to achieve the following two goals:

- **Goal 1:** Data reconstruction protection. \( r + \delta \) contains as less information as possible about \( x \). Moreover, the perturbation \( \delta \) should be effective enough. Hence, we require \( \delta \) to cover all directions of \( x \), and force the entropy of \( \delta \) to be as large as possible. Formally, we quantify the data reconstruction protection using the below MI objective:

\[
\min_{f, p(\delta) \sim P} I(r + \delta; x) - H(\delta),
\]

(14)

- **Goal 2:** Utility preservation. To ensure \( r \) be useful, it should be effective for predicting the label \( y \). Further, as we will share the perturbed representation \( r + \delta \), it should be also effective for predicting \( y \). Formally, we quantify the utility preservation using the MI objective as follows:

\[
\max_{f, p(\delta) \sim P} I(r + \delta; y) + I(r; y),
\]

(15)

3.3.2 Estimating MI via tractable bounds

Minimizing the upper bound MI in Equation (14). Similarly, we adapt the variational upper bound in [15]. Our Goal 1 for data reconstruction protection can be reformulated as the below min-max objective function:

\[
\min_{f, p(\delta) \sim P} \max_{\Psi} I(r + \delta; x) - \alpha H(\delta),
\]

(16)

\[= \min_{f, p(\delta) \sim P} \left( \max_{\Psi} \mathbb{E}_{p(r, \delta, x)} \left[ \log q_{\Omega}(x|f(x) + \delta) \right] - \alpha H(\delta) \right)\]

Remark. Equation (16) can be interpreted as an adversarial game between an adversary \( q_{\Omega} \) (i.e., data reconstructor) who aims to infer \( x \) from \( r + \delta \), and the encoder \( f \) who aims to protect \( x \) from being inferred via carefully perturbing \( r \).

Maximizing the lower bound MI in Equation (15). Based on [50], we can produce a lower bound on the MI \( I(r; y) \) due to the non-negativity of the KL-divergence:

\[
I(r; y) = \mathbb{E}_{p(y|r)} \left[ \log q_{\Omega}(y|r) \right] + \mathbb{E}_{p(r)} \left[ KL(p(y|r)||q_{\Omega}(y|r)) \right]
\]

\[\geq \mathbb{E}_{p(y|r)} \left[ \log q_{\Omega}(y|r) \right] + H(y),
\]

(17)

where \( q_{\Omega} \) is an arbitrary posterior distribution that predicts the label \( y \) from \( r \) and the entropy \( H(y) \) is a constant.

We have a similar form for the MI \( I(r + \delta; y) \) as below

\[
I(r + \delta; y) \geq \max_{\Psi} \mathbb{E}_{p(\delta) \sim P} \left[ \log q_{\Omega}(y|r + \delta) \right] + H(y),
\]

(18)

where we use the same \( q_{\Omega} \) to predict the label \( y \) from the perturbed representation \( r + \delta \).

Then, our **Goal 2** for utility preservation can be rewritten as the following max-max objective function:

\[
\max_{f, p(\delta) \sim P} \left( I(r + \delta; y) + I(r; y) \right)
\]

(19)

\[\Leftrightarrow \max_{f, p(\delta) \sim P} \left( \max_{\Psi} \mathbb{E}_{p(y, \delta)} \left[ \log q_{\Omega}(y|r + \delta) \right] + \mathbb{E}_{p(r)} \left[ \log q_{\Omega}(y|r) \right] \right).
\]

Remark. Equation (19) can be interpreted as a cooperative game between the encoder \( f \) and the label prediction network \( q_{\Omega} \), who aim to preserve the utility collaboratively.

**Objective function of \( \text{Inf}^2 \text{Guard} \) against DRAs.** By combining Equations (16)-(19), our objective function of learning privacy-preserving representations against DRAs is:

\[
\max_{f, p(\delta) \sim P} \left( \lambda \left( \min_{\Psi} \mathbb{E}_{p(\delta)} \left[ \log q_{\Psi}(y|f(x) + \delta) \right] + \alpha H(\delta) \right) \right.
\]

\[+ (1 - \lambda) \left( \max_{\Omega} \mathbb{E}_{p(y, \delta)} \left[ \log q_{\Omega}(y|f(x) + \delta) \right] + \mathbb{E}_{p(r)} \left[ \log q_{\Omega}(y|f(x)) \right] \right),
\]

(20)

where \( \lambda \in [0, 1] \) trades off privacy and utility. A larger \( \lambda \) implies less data features can be inferred through the perturbed representation, while a smaller \( \lambda \) implies the shared perturbed representation is easier for predicting the label.

3.3.3 Parameterizing perturbation distributions

The key of our defense lies in defining the perturbation distribution \( p(\delta) \) in Equation (20). Directly specifying the optimal perturbation distribution is challenging. Motivated by variational inference [36], we propose to parameterize \( p(\delta) \) with trainable parameters, e.g., \( \Phi \). Then the optimization problem w.r.t. the perturbation \( \delta \) can be converted to be w.r.t. the parameters \( \Phi \), which can be solved via back-propagation.

A natural way to model the perturbation around a representation is using a distribution with an explicit density function. Here we adopt the method in [36] by transforming \( \delta \) such that the reparameterization trick can be used in training. For instance, when considering \( p(\delta) \) as a Gaussian distribution \( \mathcal{N}(\mathbf{\mu}, \sigma^2) \), we can reparameterize \( \delta \) (with a scale \( \epsilon \)) as:

\[
\delta = \epsilon \cdot \text{tanh}(u), \quad u \sim \mathcal{N}(\mathbf{\mu}, \text{diag}(\sigma^2)).
\]

(21)

That is, it first samples \( u \) from a diagonal Gaussian with a mean vector \( \mathbf{\mu} \) and standard deviation vector \( \sigma \), and \( \delta \) is obtained by compressing \( u \) to be \([-1, 1]\) via the \( \text{tanh}(\cdot) \) function and multiplying \( \mathbf{\epsilon} \). \( \mathbf{\Phi} = (\mathbf{\mu}, \sigma) \) are the parameters to be learnt.

3.3.4 Implementation in practice

We train three neural networks (i.e., the encoder \( f \), reconstruction protection network \( q_{\Psi} \), and utility preservation network \( h_{\Omega} \)) using data samples from certain data distribution. Suppose we are given a set of data samples \( D = \{x_j, y_j\} \).

\[\text{4.1}}\text{ USENIX Security Symposium}

\[\text{43rd USENIX Association}\]
Learning the encoder \( g_r \): As \( x \) and its representation \( r \) are often high-dimensional, the previous MI estimators are inappropriate in this setting. To address it, we use the Jensen-Shannon divergence (JSD) [29] specially for high-dimensional MI estimation. Assume we have updated \( \Phi \). We can approximate the expectation w.r.t. \( g_r \) as:

\[
E_{p(x), p(\delta)} \log g_r(x | f(x) + \delta) = I_{\Theta \Psi}^{(JSD)}(x : f_0(x) + \delta) \\
\approx \sum_{x_i \in D, \delta_i \sim p(\delta)} \left[ -sp(-h_{\Phi}(x_i, f_0(x_i) + \delta_i)) \right] \\
- \sum_{(x_i, y_i) \in D} \left[ sp(h_{\Psi}(x_i, f_0(x_i) + \delta_i)) \right],
\]

where \( x'_i \) is an independent and random sample from the same distribution as \( x_i \), and \( sp(z) = \log(1 + \exp(z)) \) is the softplus function. We maximize \( I_{\Theta \Psi}^{(JSD)} \) to update \( g_r \).

Learning the utility preservation network \( h_{\Omega} \): We first estimate the below expectation:

\[
E_{p(x)} \log q_u(y | f(x) + \delta) \approx - \sum_{(x_i, y_i) \in D} H(y, h_2(f(x_i) + \delta_i)).
\]

Similarly, we can approximate the third expectation as:

\[
E_{p(x)} \log q_u(y | f(x)) \approx - \sum_{(x_i, y_i) \in D} H(y, h_2(f(x_i))).
\]

We minimize the two cross entropy losses to update \( h_{\Omega} \).

Updating the distribution parameter \( \Phi \): Due to the parameterization trick, the gradient can be back-propagated from each \( \delta_i \) to the parameters \( \Phi \). For simplicity, we do not consider the JSD term in Equation (22) due to its complexity. Then we have the terms relevant to \( \Phi \) as below:

\[
E_{c \sim X(0,1), \epsilon \sim \tanh(\mu + \sigma z)} \sum_{y, y_i} H(y, h_2(f(x_i) + \epsilon \cdot \text{tanh}(\mu + \sigma z))) - \beta \cdot H(\epsilon \cdot \text{tanh}(\mu + \sigma z)),
\]

where \( \beta = \lambda \alpha / (1 - \lambda) \). The first term is the cross entropy loss, while the second term is the entropy. The gradient w.r.t. \( \Phi \) in each term can be calculated. In practice, we approximate the expectation on \( z \) with (e.g., 5) Monte Carlo samples, and perform the stochastic gradient descent to update \( \Phi \). Details on updating \( \Phi \) are in Algorithm 3. With \( \Phi \), we use it to generate \( \delta \) and add it to \( r \) to produce the perturbed representation.

Learning the encoder \( f \). Finally, after updating \( g_r, h_{\Omega} \), and \( \Phi \), we can perform gradient ascent to update \( f \).

We iteratively train the networks until reaching a predefined maximum round. Figure 3 illustrates Inf\(^2\)Guard against DRAs. Algorithm 4 in Appendix details the training process.

4 Theoretical Results

Due to limited space, we mainly show the guaranteed privacy leakage under Inf\(^2\)Guard. We also derive an inherent utility-privacy tradeoff of Inf\(^2\)Guard, which requires a binary classification task, and binary-valued dataset property in PIAs. Details and proofs are deferred to the full version.

Guaranteed privacy leakage of MIAs: Let \( A_{\text{MIA}} \) be the set of all MIAs \( A_{\text{MIA}} = \{ A_{\text{MIA}} : Z \rightarrow u \in \{0,1\} \} \) that have access to the representations \( r \) by querying \( f \) with data \( x \) from the distribution \( D \). The MIA accuracy is bounded as below:

**Theorem 1.** Let \( f \) be the learnt encoder by Equation (6) over a data distribution \( D \subset X \). For a random data sample \( x \sim D \) with the learnt representation \( r = f(x) \) and membership \( u \), we have \( Pr(A_{\text{MIA}}(r) = u) \leq 1 - \frac{H(u | r)}{2 \log_{64}(6 / H(u | r))} \), \( \forall A_{\text{MIA}} \in A_{\text{MIA}}. \)

**Remark.** Theorem 1 shows when \( H(u | r) \) is larger, the bounded MIA accuracy is smaller. Note \( H(u | r) = H(u) - I(u; r) \) and \( H(u) \) is a constant. Achieving a large \( H(u | r) \) implies obtaining a small \( I(u; r) \), which is our Goal 1 in Equation (1) does. In practice, once the encoder \( f \) is learnt on a dataset from \( D \), \( I(u; r) \) can be estimated, then the bounded MIA accuracy can be calculated. A better encoder \( f \) or/and better MI estimator of \( I(u; r) \) can yield a smaller MIA performance.

Guaranteed privacy leakage of PIAs: Let \( A_{\text{PIA}} \) be the set of all PIAs that have access to the representations \( R \) of a dataset \( X = \{ x \} \) sampled from the data distribution \( D \), i.e., \( A_{\text{PIA}} = \{ A_{\text{PIA}} : Z \rightarrow u \in \{0,1\} \} \). The PIA accuracy is bounded as:

**Theorem 2.** Let \( f \) be the learnt encoder by Equation (11) over a data distribution \( D \). For a random dataset \( X \sim D \) with the learnt representation \( r = f(X) \) and dataset property \( u \), we have \( Pr(A_{\text{PIA}}(R) = u) \leq 1 - \frac{H(u | R)}{2 \log_{64}(6 / H(u | R))} \), \( \forall A_{\text{PIA}} \in A_{\text{PIA}}. \)

**Remark.** Theorem 2 shows when \( H(u | R) \) is larger, the PIA accuracy is smaller, i.e., less dataset property is leaked. Also, a large \( H(u | R) \) indicates a small \( I(u; R) \)—This is exactly our Goal 1 in Equation (7) aims to achieve.

Guaranteed privacy leakage of DRAs: Let \( A_{\text{DRA}} \) be the set of all DRAs that have access to the perturbed data representations, i.e., \( A_{\text{DRA}} = \{ A_{\text{DRA}} : r + \delta \in Z \times D \rightarrow x \in X \} \). An \( \ell_p \)-norm ball centered at a point \( v \) with a radius \( \rho \) is denoted as \( B_p(v, \rho) \), i.e., \( B_p(v, \rho) = \{ v' : \| v' - v \|_p \leq \rho \} \). For a space \( S \), we denote its boundary as \( \partial S \), whose volume is denoted as \( \text{Vol}(\partial S) \). Then the reconstruction error (in terms of \( \ell_p \) norm difference) incurred by any DRA is bounded as below:

**Theorem 3.** Let \( f \) be the encoder learnt by Equation (20) over a data distribution \( D \subset X \) and \( \delta \) be the perturbation for a random sample \( x \sim X \). Then, \( Pr(\| A_{\text{DRA}}(r + \delta) - x \|_p \geq \eta) \geq 1 - \frac{H(x | r + \delta) + \log_2 \text{Vol}(\partial X(\eta))}{\log \text{Vol}(\partial D) - \log \text{Vol}(\partial D(\eta))} \), \( \forall A_{\text{DRA}} \in A_{\text{DRA}}, \) where \( \text{Vol}(\partial X(\eta)) = \max_{x \in X} \text{Vol}(\partial B_p(x, \eta) \cap X) \).

**Remark.** Theorem 3 shows a lower bound error achieved by the strongest DRA. Given an \( \eta \), when \( I(x; r + \delta) \) is smaller, the lower bound data reconstruction error is larger, meaning the privacy of the data itself is better protected. Moreover, minimizing \( I(x; r + \delta) \) is exactly our Goal 1 in Equation (14).
5 Evaluations

In this section, we will evaluate InfGuard against the MIAs, PIAs, and DRAs on benchmark datasets. InfGuard involves training the encoder, the privacy protection network, and the utility preservation network. The detailed dataset description and architectures of the networks are given in the full version.

5.1 Defense Results on MIAs

5.1.1 Experimental setup

Datasets: Following existing works [35, 45], we use the CIFAR10 [37], Purchase100 [45], and Texas100 [58] datasets, to evaluate InfGuard against MIAs.

Defense/attack training and testing: The training sets and test sets are listed in Table 9 in Appendix A. For instance, in CIFAR10, we use 50K samples in total and split it into two halves, where 25K samples are used as the utility training set ("members") and the other 25K samples as the utility test set ("non-members"). We select 80% of the members and non-members as the attack training set and the remaining members and non-members as the attack test set.

- Defense training: We use the utility training set and attack training set to train the encoder, utility preservation network, and membership protection network simultaneously. Then, the learnt encoder is frozen and published as an API.
- Attack training: To mimic the strongest possible MIA, we let the attacker know the exact membership protection network and attack training set used in defense training. Specifically, s/he feeds the attack training set to the learnt encoder to get the data representations and trains the MIA classifier (same as the membership protection network) on these representations to maximally infer the membership.
- Defense and attack testing: We use the utility test set to obtain the utility (i.e., test accuracy) via querying the trained encoder and utility network. Moreover, we use the attack test set to obtain the MIA performance.

Privacy metric: We measure the MIA performance via both the MIA accuracy and the true positive rate (TPR) vs. false positive rate (FPR), suggested by the SOTA LiRA MIA [12].

$$\lambda$$ Utility MIA Acc $$\lambda$$ Utility MIA Acc $$\lambda$$ Utility MIA Acc

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>78.9%</td>
<td>70.1%</td>
<td>0</td>
</tr>
<tr>
<td>0.25</td>
<td>78.2%</td>
<td>55.9%</td>
<td>0.25</td>
</tr>
<tr>
<td>0.5</td>
<td>78%</td>
<td>53.5%</td>
<td>0.5</td>
</tr>
<tr>
<td>0.75</td>
<td>77.2%</td>
<td>51.1%</td>
<td>0.75</td>
</tr>
<tr>
<td>1</td>
<td>20%</td>
<td>50%</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 1: InfGuard results against MIAs on the three dataset. $$\lambda = 0$$ means no privacy protection, while $$\lambda = 1$$ means no utility preservation. Random guessing MIA accuracy is 50%.

Specifically, the MIA accuracy is obtained by querying the trained encoder and trained MIA classifier with the attack test set. Moreover, we treat the representations and membership network learnt by InfGuard as the input data and target model for LiRA. We then train 16 white-box shadow models (i.e., assume LiRA uses the exact membership network in InfGuard) on the data representations of the utility training set, and report the TPR vs. FPR on the attack test set.

5.1.2 Experimental results

Utility-privacy results: According to Equation (6), $$\lambda = 0$$ indicates no privacy protection. Increasing $$\lambda$$’s value enhances InfGuard’s resilience against MIAs. $$\lambda = 1$$ means the maximum privacy protection without preserving utility. Table 1 shows the utility-MIA Accuracy results of InfGuard. We have the following observations: 1) The MIA accuracy is the largest when $$\lambda = 0$$, implying leaking the most membership privacy by MIAs. 2) When only protecting privacy ($$\lambda = 1$$), the MIA accuracy reaches to the optimal random guessing, but the utility is the lowest. 3) When $$0 < \lambda < 1$$, InfGuard obtains reasonable utility and MIA accuracy. Especially, when $$\lambda = 0.75$$, the utility loss is marginal (i.e., < 4%), while the MIA accuracy is close to random guessing. The results show the learnt privacy-preserving encoder/representations are effective against MIAs, and maintain utility as well.

Further, Figure 4 shows the TPR vs FPR of InfGuard against LiRA. Similarly, we observe that the TPR at low FPRs is relatively large (strong membership inference) in case of no privacy protection, but it can be largely reduced by increasing $$\lambda$$. This implies that InfGuard indeed learns the representations that can defend against LiRA to some extent.

![Figure 4: TPR vs FPR of InfGuard against LiRA on different $$\lambda$$’s.](image)
Figure 5: Inf²Guard against MIAs: 3D t-SNE embeddings results on the learnt representation of on CIFAR10.

Figure 6: Inf²Guard against MIAs: 3D t-SNE embeddings results on the learnt representation on Purchase100.

Figure 7: Inf²Guard against MIAs: 3D t-SNE embeddings results on the learnt representation on Texas100.

Visualizing the learnt representations: To better understand the learnt representations by Inf²Guard, we adopt the t-SNE algorithm [64] to visualize the low-dimensional embeddings of them. \( \lambda \) is chosen in Table 1 that achieves the best utility-privacy tradeoff. We also compare with the case without privacy protection. Figures 5-7 show the 3D t-SNE embeddings, where each color corresponds to a label in the learning task or (non)member in the privacy task, and each point is a data sample. We can observe the t-SNE embeddings of the learnt representations without privacy protection for members and non-members are separated to some extent, meaning the membership can be inferred via the learnt MIA classifier. On the contrary, the t-SNE embeddings of the learnt representations by our Inf²Guard for members and non-members are mixed—hence making it difficult for the (best) MIA classifier to infer the membership from these learnt representations.

Comparing with the existing defenses against MIAs: All empirical defenses are broken by stronger attacks [16, 59], except adversarial training-based AdvReg [45] (a special case of Inf²Guard). NeuGuard [71] is a recent empirical defense and shows better performance than, e.g., [35, 57]. Differential privacy is the only defense with privacy guarantees. We propose to use two DP variants, i.e., DP-SGD [3] and DP-encoder (details in Appendix A). The comparison results of these defenses are shown in Table 2 (more DP results in Table 12 in Appendix B) and Figure 8. From Table 2, we observe DP methods have bad utility when ensuring the same level of defense performance (w.r.t. MIA accuracy) as Inf²Guard. AdvReg and NeuGuard also perform worse than Inf²Guard.

Table 2: Comparing Inf²Guard with existing defenses against MIAs on the three datasets. DP methods are under the same/close defense performance as Inf²Guard.
5.2 Defense Results on PIAs

5.2.1 Experimental setup

Datasets: Following recent works [13, 63], we use three datasets (Census [63], RSNA [63], and CelebA [40]) and treat the female ratio as the private dataset property.

Defense/attack training and testing: We first predefine a (different) female ratio set in each dataset. For each female ratio, we generate a number of subsets from the training set and test set with different subset sizes. The generated training/test subsets and all data samples in these subsets are treated as the attack training/test set and the utility training/test set, respectively. More details are in Table 10 in Appendix A.

- Defense training: We use the utility training set and attack training set to train the encoder, utility preservation network, and property protection network simultaneously. Then, the learnt encoder is frozen and published as an API.

5.2.2 Experimental results

Utility-privacy results: Table 3 shows the utility-privacy results of \( \text{Inf}^2\text{Guard} \), where the encoder uses a mean-aggregation. \( \lambda = 0 \) means no privacy protection, while \( \lambda = 1 \) means no utility preservation. Random guessing PIA accuracy on the three datasets are 25%, 14.3%, and 9.1%, respectively.

Table 3: \( \text{Inf}^2\text{Guard} \) results against PIAs with a mean-aggregation. \( \lambda = 0 \) means no privacy protection, while \( \lambda = 1 \) means no utility preservation. Random guessing PIA accuracy on the three datasets are 25%, 14.3%, and 9.1%, respectively.

\[
\begin{array}{cccccc}
\text{Census} & \text{Utility} & \text{PIA Acc} & \text{RSNA} & \text{Utility} & \text{PIA Acc} & \text{CelebA} & \text{Utility} & \text{PIA Acc} \\
\hline
0 & 85\% & 68\% & 0 & 83\% & 52\% & 0 & 91\% & 50\% \\
0.25 & 80\% & 61\% & 0.25 & 82\% & 25\% & 0.25 & 91\% & 28\% \\
0.5 & 78\% & 52\% & 0.5 & 82\% & 24\% & 0.5 & 91\% & 17\% \\
0.75 & 76\% & 34\% & 0.75 & 80\% & 19\% & 0.75 & 89\% & 11\% \\
1 & 45\% & 26\% & 1 & 50\% & 15\% & 1 & 53\% & 10\% \\
\end{array}
\]

- Attack training: We mimic the strongest possible PIA, where the attacker knows the exact property protection network, the aggregator, and attack training set used in defense training. Specifically, s/he feeds each subset in the attack training set to the learn encoder to get the subset representation, applies the aggregator to obtained the aggregated representation, and trains the PIA classifier (same as the property protection network) on these aggregated representations to maximally infer the private female ratio.

- Defense/attack testing: We utilize the utility test set to obtain the utility via querying the trained encoder and utility network, and the attack test set to obtain the PIA accuracy via querying the trained encoder and trained PIA classifier.

Figure 8 shows the TPR vs. FPR of these defenses against LiRA under the results in Table 2. For DP methods, we also plot the TPR vs FPR when their utility is close to \( \text{Inf}^2\text{Guard} \). With an MIA accuracy close to random guessing (but low utility), we observe DP methods have the smallest TPR at a given low FPR. This means DP methods can most reduce the attack effectiveness of LiRA, which is also verified in [12]. However, if DP methods have a close utility as \( \text{Inf}^2\text{Guard} \), their TPRs are much higher than \( \text{Inf}^2\text{Guard}'s \) at a low FPR. Besides, \( \text{Inf}^2\text{Guard} \) has smaller TPRs than AdvReg and NeuGuard.

\[
\begin{align*}
\text{Census} &: \lambda = 0.25, \text{PIA Acc} = 82\%, \text{Utility} = 52\% \\
\text{RSNA} &: \lambda = 0.25, \text{PIA Acc} = 82\%, \text{Utility} = 25\% \\
\text{CelebA} &: \lambda = 0.5, \text{PIA Acc} = 82\%, \text{Utility} = 24\% \\
\end{align*}
\]

\[
\begin{align*}
\text{Census} &: \lambda = 0.75, \text{PIA Acc} = 79\%, \text{Utility} = 19\% \\
\text{RSNA} &: \lambda = 0.75, \text{PIA Acc} = 80\%, \text{Utility} = 17\% \\
\text{CelebA} &: \lambda = 1, \text{PIA Acc} = 53\%, \text{Utility} = 10\% \\
\end{align*}
\]

\[
\begin{align*}
\text{Census} &: \lambda = 0, \text{PIA Acc} = 85\%, \text{Utility} = 68\% \\
\text{RSNA} &: \lambda = 0, \text{PIA Acc} = 83\%, \text{Utility} = 52\% \\
\text{CelebA} &: \lambda = 0, \text{PIA Acc} = 91\%, \text{Utility} = 50\% \\
\end{align*}
\]
Table 4: \textsuperscript{Inf\textsuperscript{2}}Guard results against PIAs with a max-aggregation. Random guessing PIA accuracy on the three datasets are 25%, 14.3%, and 9.1%, respectively.

<table>
<thead>
<tr>
<th>( \lambda )</th>
<th>Census Utility PIA Acc</th>
<th>RSNA Utility PIA Acc</th>
<th>CelebA Utility PIA Acc</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>85%</td>
<td>65%</td>
<td>0</td>
</tr>
<tr>
<td>0.25</td>
<td>83%</td>
<td>51%</td>
<td>0.25</td>
</tr>
<tr>
<td>0.5</td>
<td>79%</td>
<td>49%</td>
<td>0.5</td>
</tr>
<tr>
<td>0.75</td>
<td>77%</td>
<td>37%</td>
<td>0.75</td>
</tr>
<tr>
<td>1</td>
<td>47%</td>
<td>30%</td>
<td>1</td>
</tr>
</tbody>
</table>

Visualizing the learnt representations: Figures 12-14 in Appendix B show the 3D t-SNE embeddings of the learnt representations with \( \lambda = 0.75 \). Similarly, we observe the t-SNE embeddings of the aggregated representations without privacy protection can be separated to a large extent, while those with privacy protection by \textsuperscript{Inf\textsuperscript{2}}Guard are mixed. This again verifies it is difficult for the (best) PIA to infer the private female ratio from the representations learnt by \textsuperscript{Inf\textsuperscript{2}}Guard.

**Impact of the aggregator used by the encoder:** In this experiment, we test the impact of the aggregator and choose a max-aggregator for evaluation, where we select the element-wise maximum value of the representations of each subset of data. Table 4 shows the results. We have similar conclusions as those with the mean-aggregator. In addition, \textsuperscript{Inf\textsuperscript{2}}Guard with the max-aggregator has slightly worse utility-privacy tradeoff, compared with the mean-aggregator. A possible reason could be the mean-aggregator uses more information of the subset representations than the max-aggregator.

**Comparing with the DP-based defense:** There exists no effective defense against PIAs, and [63] shows DP-SGD [3] does not work well. Here, we propose to use a DP variant called DP-encoder, similar to that against MIA. More details about DP-encoder are in Appendix A. The compared results are shown in Table 5. We can see that, with the same level privacy protection as \textsuperscript{Inf\textsuperscript{2}}Guard, DP has much worse utility.

<table>
<thead>
<tr>
<th>Defense</th>
<th>Census Utility PIA Acc</th>
<th>RSNA Utility PIA Acc</th>
<th>CelebA Utility PIA Acc</th>
</tr>
</thead>
<tbody>
<tr>
<td>DP-encoder</td>
<td>52%</td>
<td>34%</td>
<td>57%</td>
</tr>
<tr>
<td>\textsuperscript{Inf\textsuperscript{2}}Guard</td>
<td>76%</td>
<td>34%</td>
<td>80%</td>
</tr>
</tbody>
</table>

5.3 Defense Results on DRAs

5.3.1 Experimental setup

**Datasets:** We select two image datasets: CIFAR10 [37] and CIFAR100 [37], and one human activity recognition dataset Activity [51] to evaluate \textsuperscript{Inf\textsuperscript{2}}Guard against DRAs.

**Defense/attack training and testing:** Table 11 in Appendix B shows the statistics of the utility/attack training and test sets.

**Defense training:** We use the training set to train the encoder, utility preservation network, reconstruction protection network, and update the perturbation distribution parameters, simultaneously. Then, the learnt encoder and perturbation distribution are published.

Table 6: \textsuperscript{Inf\textsuperscript{2}}Guard results against DRAs. A smaller SSIM or PSNR indicates better defense performance (\( \lambda = 0.4 \)).

<table>
<thead>
<tr>
<th>Activity</th>
<th>Scale ( \epsilon )</th>
<th>Utility SSIM/PSNR</th>
<th>Scale ( \gamma )</th>
<th>Utility SSIM/PSNR</th>
</tr>
</thead>
<tbody>
<tr>
<td>CIFAR10</td>
<td>0</td>
<td>0.78</td>
<td>15.97</td>
<td>0</td>
</tr>
<tr>
<td>CIFAR100</td>
<td>0.75</td>
<td>0.42</td>
<td>12.09</td>
<td>0.75</td>
</tr>
<tr>
<td></td>
<td>1.25</td>
<td>0.21</td>
<td>11.87</td>
<td>1.25</td>
</tr>
<tr>
<td></td>
<td>1.75</td>
<td>0.17</td>
<td>11.21</td>
<td>1.75</td>
</tr>
<tr>
<td></td>
<td>2.25</td>
<td>0.11</td>
<td>10.62</td>
<td>2.25</td>
</tr>
</tbody>
</table>

- **Attack training:** We mimic the strongest DRA, where the attacker knows the reconstruction protection network, training set, and perturbation distribution. She feeds each training data to the learnt encoder + perturbation distribution to get the perturbed representation. Then the attacker trains the reconstruction network (using the pair of input data and its perturbed representation) to infer the training data.

- **Defense/attack testing:** We use the utility test set to obtain the utility via querying the encoder and utility network; and use the attack test set to obtain the DRA performance by querying the trained encoder and reconstruction network.

**Privacy metric:** For image datasets, we use the common Structural Similarity Index Measure (SSIM) and PSNR metrics [27]. A larger SSIM (or PSNR) between two images indicate they look more similar. An effective attack aims to achieve a large SSIM (or PSNR), while the defender does the opposite. For human activity dataset, we use the mean-square error (MSE) between two samples to measure similarity. A smaller/larger MSE indicates a more effective attack/defense.

5.3.2 Experimental results

**Utility-privacy results:** Table 6 shows the defense results of \textsuperscript{Inf\textsuperscript{2}}Guard with the Gaussian perturbation distribution, where \( \lambda = 0.4 \) in Equation (20). We can observe \( \epsilon \) acts a utility-privacy tradeoff. A larger \( \epsilon \) implies adding more perturbation to the representation during defense training. This makes the DRA more challenging, but also sacrifice the utility more.

We also test the impact of \( \lambda \) and the results are shown in Table 7. We can see \( \lambda \) also acts as a tradeoff—a larger \( \lambda \) can protect data privacy more, while having larger utility loss.

**Comparing with the DP-based defense:** All empirical defenses against DRAs are broken by an advanced attack [9]. A few papers [7, 53] show if a randomized algorithm satisfies DP, it can defend against DRAs with provable guarantees. We compare \textsuperscript{Inf\textsuperscript{2}}Guard with DP and Table 8 shows the DP results. Viewing with results in Table 6, we see \textsuperscript{Inf\textsuperscript{2}}Guard obtains better utility-privacy tradeoffs than DP-SGD.
Visualizing data reconstruction results: Figure 9 and Figure 10 show the reconstruction results on some CIFAR10 and CIFAR100 images, respectively. We see that, without defense, the attacker can accurately reconstruct the raw images. With a similar utility, visually, InfGuard can better defend against image reconstruction than DP. Figure 11 summarizes the reconstruction results on 50 samples in Activity, where we report the difference between each reconstructed feature by InfGuard and that by DP to the true feature. A (larger) positive value implies InfGuard is (more) dissimilar than DP to the true feature. We can see InfGuard has better defense results than DP in most (413 out of 516) of the features.

6 Discussion and Future Work

InfGuard and DP: Essentially, InfGuard and DP are two different provable privacy mechanisms, and they complement each other. First, DP mainly measures the user or sample-level privacy risks in the worst case while InfGuard can accurately measure the average privacy risks at the dataset level with the derived bounds. Second, DP has been shown to provide some resilience transferability across some inference attacks [53] (but not all of them). It is also interesting to study the resilience transferability for the proposed InfGuard, which we will explore in the future. More importantly, our InfGuard can complement DP. For instance, we can use the learnt (deterministic) data representations by InfGuard as input to DP-SGD or add (Gaussian) noise to the representations to ensure DP guarantees against MIAs.

Task-agnostic representation learning: Our current MI formulation for utility preservation knows the labels of the learning task (e.g., see Equation (2)). A more promising solution would be task-agnostic, i.e., learning task-agnostic representations that can benefit many (unknown) downstream tasks. We note that our framework can be easily extended to this scenario. For instance, in MIAs, we now require the learnt representation \( r \) includes as much information about the training sample \( x \) as possible (i.e., \( u = 1 \)). Intuitively, when \( r \) retains all information about \( x \), the model trained on \( r \) will have the same performance as trained on the raw \( x \), despite the learning task. Formally, the MI objective becomes max \( I(x; r|u = 1) \).

Defending against multiple inference attacks simultaneously: We design the customized MI objectives to defend against each inference attack in the paper. A natural solution to defend against multiple inference attacks is unifying their training objectives (by summarizing them with tradeoff hyperparameters). While this is possible, we emphasize that the learnt encoder is weak against all attacks. This is because the encoder should balance the defense effectiveness among these attacks, and cannot be optimal against all of them.

Generalizing our theoretical results: Our theoretical results assume the learning task is binary classification and dataset property is binary-valued. We will generalize our theoretical results to multiclass classification and other types of learning such as regression and multi-valued dataset property.
Generalizing our framework against security attacks: In our current framework, each privacy protection task is formalized via an MI objective. An important future work would be generalizing our framework to design customized MI objectives to learn robust representations against security attacks such as evasion, poisoning, and backdoor attacks.

7 Related Work

7.1 MIAs and Defenses

MIAs [12, 14, 16, 31, 39, 54, 58–60, 70, 73, 76]. Existing MIAs can be classified as training based [12, 14, 16, 39, 52, 54, 58, 60, 72, 73] and non-training based [16, 59]. Given a (non)training sample and its output by a target ML model, training based MIAs use the (sample, output) pair to train a binary classifier, which is then used to determine whether a testing sample belongs to the training set or not. For instance, [58] introduces multiple shadow models to perform training. In contrast, non-training based MIAs directly use the samples’ predicted score/label to make decisions. For instance, [59] designs a metric prediction correctness, which infers the membership based on whether a given sample is correctly classified by the target model or not. Overall, an MIA that has more information is often more effective than that has less information.

Defenses [35, 45, 54, 56, 58, 59, 61, 71]. They can be categorized as training time based defense (e.g., dropout [54], L2 norm regularization [58], model stacking [54], adversarial regularization [45], loss variance deduction [71], DP [3, 32, 75], early stopping [59], knowledge distillation [56]) and inference time based defense (e.g., MemGuard [35]). Almost all of them are empirical and broken by stronger attacks [16, 59]. DP is only defense offering privacy guarantees. Its main idea is to add noise to the gradient [3, 75] or objective function [32] during training. The main drawback of current DP methods is that they have significant utility losses [33, 56].

7.2 PIAs and Defenses

PIAs [5, 6, 13, 20, 24, 41, 42, 63, 66, 77, 79]. Ateniese et al. [6] are the first to describe the problem of the PIA (against support vector machines and hidden Markov models), where the attack is performed in the the white-box setting and consists of training a meta-classifier on top of many shadow models. Ganju et al. [20] extend PIAs to neural networks, particularly fully connected neural networks (FCNNs). Zhang et al. [77] propose PIAs in the black-box setting and train a meta-classifier based on shadow models. Mahloujifar et al. [41] observe that data poisoning attacks can be incorporated into training the shadow model and increase the effectiveness of PIAs. Suri and Evans [63] are the first to formally formalize PIAs as a cryptographic game, inspired by the way to formalize MIAs [73]. They also extend the white-box attack on FCNNs [20] to convolutional neural networks (CNNs). Zhou et al. [79] develop the first PIA against generative models, i.e., generative adversarial networks (GANs) [23], under the black-box setting. Chaudhari et al. [13] propose a data poisoning strategy to perform the efficient private property inference.

Defenses. To our best knowledge, there exist no known effective defenses against PIAs. DP cannot mitigate PIAs since it obfuscates individual samples, while PIAs care about the entire datasets [63]. [63] also shows that DP does not work as a potential defense (also verified in Section 5).

7.3 DRAs and Defenses

DRAs [7–9, 19, 22, 27, 28, 34, 67, 68, 74, 78, 81]. Existing DRAs mainly reconstruct the training data from the model parameters or representations. They are formulated as an optimization problem that minimizes the difference between gradient from the raw training data and that from the reconstructed data. For instance, Zhu et al. [81] proposed a DLG attack method which relies entirely on minimization of the difference of gradients. Furthermore, several methods [22, 28, 34, 67, 74] propose to incorporate prior knowledge (e.g., total variation regularization) into the training data, or introduce an auxiliary dataset to simulate the training data distribution (e.g., via GANs [23]). A few works [22, 80] derive close-formed solutions to reconstruct the data, by constraining the neural networks to be fully connected [22] or convolutional [80].

Defenses [21, 25, 38, 48, 55, 62, 69, 81]. Most of these defenses have none/little privacy guarantees. For instance, Zhu et al. [81] propose to prune model parameters with smaller magnitudes. Sun et al. [62] propose to obfuscate the gradient for a single layer (called defender layer) such that the reconstructed data and the original data are dissimilar. Gao et al. [21] propose to generate augmented images that, when they are used to train the network, produce non-invertible gradients. These defenses are broken by an advanced attack based on Bayesian learning. Only defenses based on DP-SGD [3], a version of SGD with clipping and adding Gaussian noise, provide formal privacy guarantees.

8 Conclusion

We propose a unified information-theoretic framework, dubbed Inf2Guard, to learn privacy-preserving representations against the three major types of inferences attacks (i.e., membership inference, property inference, and data reconstruction attacks). The framework formalizes the utility preservation and privacy protection against each attack via customized mutual information objectives. The framework also enables deriving theoretical results, e.g., inherent utility-privacy tradeoff, and guaranteed privacy leakage against each attack. Extensive evaluations verify the effectiveness of Inf2Guard for learning privacy-preserving representations and show the superiority over the compared baselines.
Acknowledgement

We thank all the anonymous reviewers and our shepherd for the valuable feedback and constructive comments. Wang is partially supported by the National Science Foundation (NSF) under grant Nos. ECCS-2216926, CNS-2241713 and CNS-2339686. Hong is partially supported by the National Science Foundation (NSF) under grant Nos. CNS-2302689, CNS-2308730, CNS-2319277 and CMMI-2326341. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the funding agencies.

References


Algorithm 1 InfGuard against MIAs

<table>
<thead>
<tr>
<th>Input</th>
<th>Dataset $D_1$ of members and dataset $D_0$ of non-members, tradeoff hyperparameter $\lambda \in [0, 1]$, learning rates $lr_1, lr_2, lr_3$; #local gradients $I$, #global rounds $T$.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output</td>
<td>Network parameters: $\Theta, \Psi, \Omega$.</td>
</tr>
<tr>
<td>1</td>
<td>Initialize $\Theta, \Psi, \Omega$ for the encoder $f$, membership protection network $g_{\Psi}$, and utility preservation network $h_{\Omega}$.</td>
</tr>
<tr>
<td>2</td>
<td>for $t = 1$ to $T$ do</td>
</tr>
<tr>
<td>3</td>
<td>$L_1 = \sum_{(x_i, y_i) \in D_1} H(g_{\Psi}(f(x_i)))$;</td>
</tr>
<tr>
<td>4</td>
<td>$L_2 = \sum_{(x_i, y_i) \in D_0} H(y_i, h_{\Omega}(f(x_i)))$;</td>
</tr>
<tr>
<td>5</td>
<td>for $i = 1$ to $I$ do</td>
</tr>
<tr>
<td>6</td>
<td>$\Psi \leftarrow \Psi - lr_1 \cdot \frac{\partial L_1}{\partial \Psi}$;</td>
</tr>
<tr>
<td>7</td>
<td>$\Omega \leftarrow \Omega - lr_2 \cdot \frac{\partial L_2}{\partial \Omega}$;</td>
</tr>
<tr>
<td>8</td>
<td>$\Theta \leftarrow \Theta + lr_3 \cdot \frac{\partial L_1}{\partial \Theta} + (1 - \lambda) \frac{\partial L_2}{\partial \Theta}$;</td>
</tr>
</tbody>
</table>

Algorithm 2 InfGuard against PIAs

<table>
<thead>
<tr>
<th>Input</th>
<th>$N$ datasets ${D_j}_{j=1}^N$ sampled from a reference dataset $D_1$ with each $D_j$ having a property value $a_j$, tradeoff hyperparameter $\lambda \in [0, 1]$, learning rates $lr_1, lr_2, lr_3$; #local gradients $I$, #global rounds $T$.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output</td>
<td>Network parameters: $\Theta, \Psi, \Omega$.</td>
</tr>
<tr>
<td>1</td>
<td>Initialize $\Theta, \Psi, \Omega$ for the encoder $f$, property protection network $g_{\Psi}$, and utility preservation network $h_{\Omega}$.</td>
</tr>
<tr>
<td>2</td>
<td>for $r = 1$ to $T$ do</td>
</tr>
<tr>
<td>3</td>
<td>$L_1 = \sum_{(x_i, y_i) \in D_1} H(a_j, g_{\Psi}(f(x_i)))$;</td>
</tr>
<tr>
<td>4</td>
<td>$L_2 = \sum_{(x_i, y_i) \in D_0} H(y_i, h_{\Omega}(f(x_i)))$;</td>
</tr>
<tr>
<td>5</td>
<td>for $i = 1$ to $I$ do</td>
</tr>
<tr>
<td>6</td>
<td>$\Psi \leftarrow \Psi - lr_1 \cdot \frac{\partial L_1}{\partial \Psi}$;</td>
</tr>
<tr>
<td>7</td>
<td>$\Omega \leftarrow \Omega - lr_2 \cdot \frac{\partial L_2}{\partial \Omega}$;</td>
</tr>
<tr>
<td>8</td>
<td>$\Theta \leftarrow \Theta + lr_3 \cdot \frac{\partial L_1}{\partial \Theta} + \frac{1}{\lambda} \frac{\partial L_2}{\partial \Theta}$;</td>
</tr>
</tbody>
</table>

Algorithm 3 Update perturbation distribution parameter $\Phi$

<table>
<thead>
<tr>
<th>Input</th>
<th>$K$ Monte Carlo samples, the encoder $f_{\theta}$ in the previous round, objective function Eqn (20), learning rate $lr_1$; #epochs $h_1$.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output</td>
<td>Perturbation distribution parameters $\Phi$.</td>
</tr>
<tr>
<td>1</td>
<td>Initialize $\Phi = (\mu, \sigma)$.</td>
</tr>
<tr>
<td>2</td>
<td>for $i = 1$ to $I$ do</td>
</tr>
<tr>
<td>3</td>
<td>for $j = 1$ to $K$ do</td>
</tr>
<tr>
<td>4</td>
<td>Sample $z_j$ from $N(0, 1)$ and compute $\delta_j = \mu + \sigma z_j$;</td>
</tr>
<tr>
<td>5</td>
<td>Calculate the gradient $g_{\delta}$ of Eqn (25) w.r.t. $\Phi$;</td>
</tr>
<tr>
<td>6</td>
<td>Update $\Phi$ by: $\Phi \leftarrow \Phi - lr_1 \cdot g_{\delta}$.</td>
</tr>
</tbody>
</table>

Algorithm 4 InfGuard against DRAs

<table>
<thead>
<tr>
<th>Input</th>
<th>A dataset $D = {x_i, y_i}$, hyperparameters $\lambda \in [0, 1]$, learning rates $lr_1, lr_2, lr_3$; #local gradients $I$, #global rounds $T$.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output</td>
<td>Network parameters: $\Theta, \Psi, \Omega$.</td>
</tr>
<tr>
<td>1</td>
<td>Initialize $\Theta, \Psi, \Omega$ for the encoder $f$, data reconstruction network $g_{\Psi}$, utility preservation network $h_{\Omega}$, and perturbation distribution parameter.</td>
</tr>
<tr>
<td>2</td>
<td>for $r = 1$ to $T$ do</td>
</tr>
<tr>
<td>3</td>
<td>for each batch $bs \subset D$ do</td>
</tr>
<tr>
<td>4</td>
<td>Update $\Phi$ via Algorithm 3;</td>
</tr>
<tr>
<td>5</td>
<td>Update $g_{\Psi}$ (given $\Theta$ and ${\delta_j}$): Calculate $\frac{\partial L_{\text{JS}}}{\partial \Psi}$ on $bs$ with ${\delta_j}$ via Eqn (22); $\Psi \leftarrow \Psi + lr_1 \cdot \frac{\partial L_{\text{JS}}}{\partial \Psi} / \partial \Psi$;</td>
</tr>
<tr>
<td>6</td>
<td>Update $h_{\Omega}$ (given $\Theta$ and ${\delta_j}$): Calculate CE loss $L_1$ on $bs$ with ${\delta_j}$ via Eqn (23); Calculate CE loss $L_2$ on $bs$ with clean data via Eqn (24); $\Omega \leftarrow \Omega - lr_2 \cdot \frac{\partial L_1}{\partial \Omega} + \frac{1}{\lambda} \frac{\partial L_2}{\partial \Theta}$;</td>
</tr>
<tr>
<td>7</td>
<td>Update $f_{\theta}$ (given $\Psi, \Omega$, and ${\delta_j}$): $\Theta \leftarrow \Theta - lr_3 \cdot \frac{\partial L_1}{\partial \Theta} + (1 - \lambda) \frac{\partial L_2}{\partial \Theta}$;</td>
</tr>
</tbody>
</table>
A  More Experimental Setup

Training and testing: Table 9-Table 11 show the utility training/test and attack training/test sets on the three datasets.

Differential Privacy (DP) against MIAs: DP provides an upper bound on the success of any MIA. We can add noise in several ways (e.g., input data, model parameters, gradients, latent features, output scores) to ensure DP. Note that there exists an inherent trade-off between utility and privacy: a larger added noise often leads to a higher level of privacy protection, but incurs a larger utility loss. Here, we propose to use the below two ways.

- **DP-SGD [3]:** 1) **DP-SGD training:** It clips gradients (with a gradient norm bound) and adds Gaussian noise to the gradient in each SGD round when training the ML model (i.e., encoder + utility network). More details can be seen in Algorithm 1 in [3]. After training, the model ensures DP guarantees and the encoder is published. 2) **Attack training:** The attacker obtains the representations of the attack training data via querying the trained encoder and uses these representations to train the MIA classifier. 3) **Defense/attack testing:** The utility test set is used to obtain the utility via querying the trained ML model; and the attack test set to obtain the MIA accuracy via querying the trained encoder and trained MIA classifier.

We used the Opacus library (https://opacus.ai/), a PyTorch extension that enables training models with DP-SGD and dynamically tracks privacy budget and utility. In the experiments, we tried ε in DP-SGD from 0.5 to 16.

- **DP-encoder:** 1) **Normal training:** It first trains the encoder + utility network using the (utility) training set. The encoder is then frozen and can be used to produce data representations when queried by data samples. 2) **Defense via adding noise to the representations:** We add Gaussian noise to the representations by querying the encoder with the attack training data to produce the noisy representations. Notice that, since the Gaussian noises are injected to the matrix outputs (data representations), if needed, the actual DP guarantee (i.e., privacy bounds) can be derived via Rényi Differential Privacy [43], similar to the theoretical studies in [44, 65]. We skip the details here since this work does not focus on the derivation for the privacy bounds of DP-encoder. 3) **Attack training:** The attacker uses the noisy representations of attack training data to train the MIA classifier. 4) **Defense/attack testing:** We use the utility test set to obtain the utility via querying the trained encoder and utility network; and use the attack test set to obtain the MIA accuracy on the trained encoder and trained MIA classifier. We call this **DP-encoder** as we add noise to the representations outputted by the well-trained encoder.

DP-encoder against PIAs: We follow the strategy in DP against MIAs and choose the DP-encoder, as DP-SGD is ineffective in this setting [63]. The only difference is that we now add Gaussian noise to the mean-aggregated representation of a subset, instead of the individual representation.

Data reconstruction attack/defense on shallow encoder: As shown in [27], when the encoder is deep, it is difficult for the attacker to reconstruct the input data from the representation. To ensure DRAs be effective, we use a shallow 2-layer encoder. As a result, this makes the defense more challenging.
Figure 12: \textsuperscript{Inf\textsuperscript{2}Guard} against PIAs: 3D t-SNE embeddings results on the learnt representation on Census Income. Each point in (b) and (d) is an aggregated representation of a dataset.

Figure 13: \textsuperscript{Inf\textsuperscript{2}Guard} against PIAs: 3D t-SNE embeddings results on the learnt representation on RSNA Bone Age.

Figure 14: \textsuperscript{Inf\textsuperscript{2}Guard} against PIAs: 3D t-SNE embeddings results on the learnt representation on CelebA.

B More Experimental Results

More results on defending against MIAs: Table 12 shows more DP results (vs varying $\varepsilon$’s) against MIAs. We can see \textsuperscript{Inf\textsuperscript{2}Guard} obtains higher utility than DP methods under the same privacy protection performance.

More results on defending against PIAs: Figure 12–Figure 14 shows the t-SNE embeddings of \textsuperscript{Inf\textsuperscript{2}Guard} against PIAs. Table 13 shows \textsuperscript{Inf\textsuperscript{2}Guard} results against PIAs, where the attacker does not the true (mean) aggregator and use a substitute one (i.e., max-aggregator). We can see the attack performance is less effective and \textsuperscript{Inf\textsuperscript{2}Guard} can yield (close to) random guessing attack performance (when $\gamma = 0.75$), with a slight utility loss. This implies the aggregator plays a critical role in designing effective PIAs against \textsuperscript{Inf\textsuperscript{2}Guard}.

Table 14: \textsuperscript{Inf\textsuperscript{2}Guard} results against DRAs with uniform perturbation distribution. A smaller SSIM or PSNR indicates better defense performance ($\lambda = 0.4$).

<table>
<thead>
<tr>
<th>Scale $\varepsilon$</th>
<th>Utility</th>
<th>SSIM/PSNR</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>89.5%</td>
<td>0.78 / 15.97</td>
</tr>
<tr>
<td>1.25</td>
<td>85.7%</td>
<td>0.28 / 13.23</td>
</tr>
<tr>
<td>2.25</td>
<td>77.7%</td>
<td>0.24 / 12.34</td>
</tr>
<tr>
<td>3.25</td>
<td>64.9%</td>
<td>0.20 / 12.93</td>
</tr>
</tbody>
</table>

More results on defending against DRAs: Table 14 shows the \textsuperscript{Inf\textsuperscript{2}Guard} results against DRAs, where the perturbation distribution is uniform distribution. We observe similar utility-privacy tradeoff in terms of the noise scale $\varepsilon$. 