Inf^2Guard: 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

This paper is included in the Proceedings of the
33rd USENIX Security Symposium.

August 14–16, 2024 • Philadelphia, PA, USA

978-1-939133-44-1

Open access to the Proceedings of the
33rd USENIX Security Symposium
is sponsored by USENIX.

Inf[2]Guard: An Information-Theoretic Framework for Learning Privacy-Preserving Representations against Inference Attacks

Sayedeh Leila Noorbakhsh[1,*], Binghui Zhang[1,*], Yuan Hong[2], Binghui Wang[1]
[1]*Illinois Institute of Technology,* [2]*University 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 `Inf`^{2}`Guard`, 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, `Inf`^{2}`Guard` involves two mutual information objectives, for privacy protection and utility preservation, respectively. `Inf`^{2}`Guard` 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 `Inf`^{2}`Guard` 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 `Inf`^{2}`Guard`, 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

[1]Source code and the full version at: https://github.com/leilynourbakhsh/Inf2Guard.

[2]To 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 `Inf²Guard` 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 `Inf²Guard` and compare it with the existing defenses against the inference attacks on multiple benchmark datasets. Our experimental results validate that `Inf²Guard` obtains a promising utility-privacy tradeoff and significantly outperforms the existing defenses. For instance, under the same defense performance against MIAs, `Inf²Guard` has a 30% higher testing accuracy than the DP-SGD [3]. Our results also validate the privacy-utility tradeoffs obtained by `Inf²Guard`[5].

Our main contributions are summarized as below:

- **Algorithm:** We design the first unified framework `Inf²Guard` 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 `Inf²Guard` for learning privacy-preserving representations against inference attacks.

2 Background and Problem Definition

Notations: We use s, $\mathbf{s}$, $\mathbf{S}$, and $\mathcal{S}$ to denote (random) scalar, vector, matrix, and space, respectively. Accordingly, $\Pr(s)$, $\Pr(\mathbf{s})$, and $\Pr(\mathbf{S})$ are the probability distribution over s, $\mathbf{s}$, and $\mathbf{S}$. $I(\mathbf{x};\mathbf{r})$ and $H(\mathbf{x},\mathbf{r})$ are the mutual information and cross entropy between a pair of random variables $(\mathbf{x},\mathbf{r})$, respectively, and $H(\mathbf{x}) = I(\mathbf{x};\mathbf{x})$ as the entropy of $\mathbf{x}$. $KL(p||q)$ is the KL-divergence between two distributions p and q. We denote $\mathcal{D}$ as the underlying distribution that data are sampled from. A data sample is denoted as $(\mathbf{x},y) \sim \mathcal{D}$, where $\mathbf{x} \in \mathcal{X}$ is data features, $y \in \mathcal{Y}$ is the label, and $\mathcal{X}$ and $\mathcal{Y}$ are the data space and label space, respectively. We further denote a dataset as $D = \{\mathbf{X},\mathbf{y}\} = \{(\mathbf{x}_i, y_i)\}$, that consists of a set of data samples $(\mathbf{x}_i, y_i) \sim \mathcal{D}$, and will interchangeably use D and $\{\mathbf{X},\mathbf{y}\}$. We let $u \in \mathcal{U}$ 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 $u \in \mathcal{U} = \mathcal{X}$ 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] $F_\theta : \mathcal{X} \to \mathcal{Y}$ as a function, parameterized by θ, that maps a data sample $\mathbf{x} \in \mathcal{X}$ to a label $y \in \mathcal{Y}$. Given a training set $D \sim \mathcal{D}$, 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 $(\mathbf{x},y) \sim \mathcal{D}$ with a private membership u that is chosen uniformly at random from $\{0,1\}$, where $u = 1$ means $(\mathbf{x},y)$ is a member of D, and 0 otherwise. An MIA A_{MIA} has access to $\mathcal{D}$ and F, takes $(\mathbf{x},y)$ as input, and outputs a binary $A_{MIA}^{\mathcal{D},F}(\mathbf{x},y)$. We omit $\mathcal{D},F$ for notation simplicity. Then, the attack performance of an MIA A_{MIA} is defined as $\Pr_{(\mathbf{x},y,u)}(A_{MIA}(\mathbf{x},y) = u)$.

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 $\Pr_{(D_u,u)}(A_{PIA}(D_u) = u)$.

Formalizing DRAs: Given a random data $(\mathbf{x},y) \in D$, DRAs aim to reconstruct the private $\mathbf{x}$. A DRA A_{DRA} has access to $\mathcal{D}$ and F, and outputs a reconstructed $\hat{\mathbf{x}} = A_{DRA}(\mathbf{x},y)$. The DRA performance is measured by the similarity/difference between $\hat{\mathbf{x}}$ and $\mathbf{x}$. For instance, [7] introduces the (η,γ)-reconstruction metric defined as $\Pr_{(\mathbf{x},y)}(\|\hat{\mathbf{x}} - \mathbf{x}\|_2 \leq \eta) \geq \gamma$, where a smaller η and a larger γ 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 $f_\Theta : \mathcal{X} \to \mathcal{Z}$ as the *encoder*, parameterized by Θ, that maps a data sample $\mathbf{x} \in \mathcal{X}$ (or a dataset $\mathbf{X} \in \mathcal{X}$) to its representation vector $\mathbf{r} = f(\mathbf{x}) \in \mathcal{Z}$ (or representation matrix $\mathbf{R} = f(\mathbf{X}) \in \mathcal{Z}$), where $\mathcal{Z}$ is the representation space. Moreover, we let $C : \mathcal{Z} \to \mathcal{Y}$ be the *classification model* on top of the representation $\mathbf{r}$ or encoder f, which predicts the data label y (or dataset labels $\mathbf{y}$). We further let $A : \mathcal{Z} \to \mathcal{U}$ be the *inference model*, which infers the private attribute u using the learnt representations $\mathbf{r}$ or $\mathbf{R}$. Then, our defense goals are:

- **Defend against MIAs:** Given a random sample $(\mathbf{x}, y, u) \in \mathcal{D}$, we expect to learn f such that the MIA performance $\Pr(A_{MIA}(f(\mathbf{x}), y) = u)$ is low, and the utility loss/risk, i.e., $\text{Risk}_{MIA}(C \circ f) = \Pr(C \circ f(\mathbf{x}) \neq y)$, is also small.
- **Defend against PIAs:** Given a random dataset $(\mathbf{X}, \mathbf{y}, u) \in \mathcal{D}$, we expect to learn f with low PIA performance $\Pr(A_{PIA}(f(\mathbf{X}), \mathbf{y}) = u)$, and also a small utility loss/risk, i.e., $\text{Risk}_{PIA}(C \circ f) = \frac{1}{|\mathbf{y}|} \sum_{(\mathbf{x}, y) \in \{\mathbf{X}, \mathbf{y}\}} \Pr(C \circ f(\mathbf{x}) \neq y)$.
- **Defend against DRAs:** Given a random sample $(\mathbf{x}, y) \in \mathcal{D}$, we expect to learn f with low DRA performance, i.e., $\Pr_{(\mathbf{x}, y)}(\|\hat{\mathbf{x}} - \mathbf{x}\|_2 \geq \eta) \geq \gamma$ with a large η and γ (flipping the inequality direction on η for DRAs), and also a small utility risk $\text{Risk}_{DRA}(C \circ f) = \Pr(C \circ f(\mathbf{x}) \neq y)$.

3 Design of `Inf`^{2}`Guard`

3.1 `Inf`^{2}`Guard` against MIAs

3.1.1 MI objectives

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

- **Goal 1: Membership protection.** $\mathbf{r}$ contains as less information as possible about the private membership u. Ideally, when $\mathbf{r}$ does not include information about u (i.e., $\mathbf{r} \perp u$), it is impossible to infer u from $\mathbf{r}$. Formally, we quantify the membership protection using the MI objective as follows:
$$\min_f I(\mathbf{r}; u), \tag{1}$$
where we minimize such MI to maximally reduce the correlation between $\mathbf{r}$ and u.
- **Goal 2: Utility preservation.** $\mathbf{r}$ 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:
$$\max_f I(y; \mathbf{r} | u = 1), \tag{2}$$
where we maximize such MI to make $\mathbf{r}$ 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,

$$I(\mathbf{r}; u) \leq I_{vCLUB}(\mathbf{r}; u) = \mathop{\mathbb{E}}_{p(\mathbf{r}, u)}[\log q_\Psi(u|\mathbf{r})] - \mathop{\mathbb{E}}_{p(\mathbf{r})p(u)}[\log q_\Psi(u|\mathbf{r})],$$

where $q_\Psi(u|\mathbf{r})$ is an auxiliary posterior distribution of $p(u|\mathbf{r})$ needing to satisfy the below condition on KL divergence: $KL(p(\mathbf{r}, u)||q_\Psi(\mathbf{r}, u)) \leq KL(p(\mathbf{r})p(u)||q_\Psi(\mathbf{r}, u))$. To achieve this, we thus minimize:

$$\begin{aligned}
&\min_\Psi KL(p(\mathbf{r}, u)||q_\Psi(\mathbf{r}, u)) = \min_\Psi KL(p(u|\mathbf{r})||q_\Psi(u|\mathbf{r})) \\
&= \min_\Psi \mathop{\mathbb{E}}_{p(\mathbf{r}, u)}[\log p(u|\mathbf{r})] - \mathop{\mathbb{E}}_{p(\mathbf{r}, u)}[\log q_\Psi(u|\mathbf{r}))] \\
&\Longleftrightarrow \max_\Psi \mathop{\mathbb{E}}_{p(\mathbf{r}, u)}[\log q_\Psi(u|\mathbf{r})],
\end{aligned} \tag{3}$$

where we note that $\mathbb{E}_{p(\mathbf{r}, u)}[\log p(u|\mathbf{r})]$ is irrelevant to Ψ. [15] proved when $q_\Psi(u|r)$ 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:

$$\min_f \min_\Psi I_{vCLUB}(\mathbf{r}; u) \Longleftrightarrow \min_f \max_\Psi \mathop{\mathbb{E}}_{p(\mathbf{r}, u)}[\log q_\Psi(u|\mathbf{r})] \tag{4}$$

Remark. Equation (4) can be interpreted as an *adversarial game* between an adversary q_Ψ (i.e., a membership inference classifier) who aims to infer the membership u from $\mathbf{r}$; 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

$$\begin{aligned} I(y;\mathbf{r}|u=1) &= H(y|u=1) - H(y|\mathbf{r},u=1) \\ &= H(y|u=1) + \mathop{\mathbb{E}}_{p(y,\mathbf{r},u)}[\log p(y|\mathbf{r},u=1))] \\ &= H(y|u=1) + \mathop{\mathbb{E}}_{p(y,\mathbf{r},u)}[\log q_\Omega(y|\mathbf{r},u=1))] \\ &\quad + \mathop{\mathbb{E}}_{p(y,\mathbf{r},u)}[KL(p(\cdot|\mathbf{r},u=1)||q_\Omega(\cdot|\mathbf{r},u=1))] \\ &\geq H(y|u=1) + \mathop{\mathbb{E}}_{p(y,\mathbf{r},u)}[\log q_\Omega(y|\mathbf{r},u=1))], \end{aligned}$$

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

$$\max_f I(y;\mathbf{r}|u=1) \Longleftrightarrow \max_f \max_\Omega \mathop{\mathbb{E}}_{p(y,\mathbf{r},u)}[\log q_\Omega[(y|\mathbf{r},u=1)] \quad (5)$$

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

Objective function of Inf²Guard against MIAs. By combining Equations (4) and (5), our objective function of learning privacy-preserving representations against MIAs is:

$$\begin{aligned} \max_f \Big(\lambda \min_\Psi - \mathop{\mathbb{E}}_{p(\mathbf{x},u)}[\log q_\Psi(u|f(\mathbf{x}))] \\ + (1-\lambda)\max_\Omega \mathop{\mathbb{E}}_{p(\mathbf{x},y,u)}[\log q_\Omega(y|f(\mathbf{x}),u=1)]\Big), \end{aligned} \quad (6)$$

where $\lambda \in [0,1]$ tradeoffs privacy and utility. That is, a larger λ indicates a stronger membership privacy protection, while a smaller λ 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_Ψ associated with the posterior distribution q_Ψ, and utility preservation network h_Ω associated with the posterior distribution q_Ω) 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_Ω (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_Ψ (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.

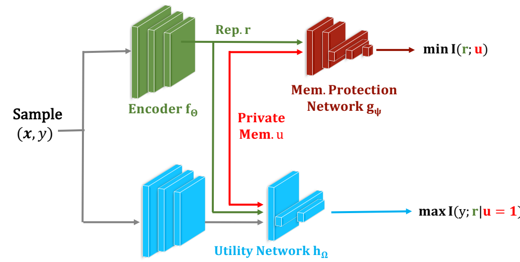


Figure 1: Inf²Guard against MIAs.

Training the membership protection network g_Ψ: We approximate the first expectation w.r.t. q_Ψ as[7]

$$\mathop{\mathbb{E}}_{p(\mathbf{x},u)} \log q_\Psi(u|f(\mathbf{x})) \approx -\sum_{(\mathbf{x}_j,u_j)\in D_1\cup D_0} H(u_j, g_\Psi(f(\mathbf{x}_j))),$$

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

Training the utility preservation network h_Ω: We approximate the second expectation w.r.t. q_Ω as:

$$\mathop{\mathbb{E}}_{p(\mathbf{x},y,u)} \log q_\Omega(y|f(\mathbf{x}),u=1) \approx -\sum_{(\mathbf{x}_j,y_j)\in D_1} H(y_j, h_\Omega(f(\mathbf{x}_j))).$$

We maximize this expectation to enhance the utility.

Training the encoder f: With the updated g_Ψ and h_Ω, 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 Inf²Guard against MIAs. Algorithm 1 in Appendix details the training.

Connection with AdvReg [45]. We observe that AdvReg is a special case of Inf²Guard. Specifically, the objective function of AdvReg can be rewritten as:

$$\max_f \Big(\lambda \min_\Psi \sum_{(\mathbf{x}_j,u_j)\in D_1\cup D_0} H(u_j, g_\Psi(f(\mathbf{x}_j))) - (1-\lambda)\sum_{(\mathbf{x}_j,y_j)\in D_1} H(y_j, f(\mathbf{x}_j))\Big),$$

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

3.2 Inf²Guard against PIAs

Different from MIAs, PIAs leak the training data properties at the *dataset-level*. To align this, instead of using a random sample $(\mathbf{x}, y)$, we consider a random dataset $(\mathbf{X}, \mathbf{y})$ in PIAs. Specifically, let $\mathbf{X} = \{\mathbf{x}_i\}$ consist of a set of independent data samples and $\mathbf{y} = \{y_i\}$ the corresponding data labels that are sampled from the underlying data distribution $\mathcal{D}$; and $\mathbf{X}$ 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 $\mathbf{X} \sim \mathcal{D}$ with a property u, the defender learns a dataset representation $\mathbf{R} = f(\mathbf{X})$ that satisfies two goals[8]:

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

$$\min_f I(\mathbf{R}; u). \tag{7}$$

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

$$\max_f I(\mathbf{y}; \mathbf{R}). \tag{8}$$

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:

$$\min_f I(\mathbf{R}; u) \Longleftrightarrow \min_f \max_\Psi \mathop{\mathbb{E}}_{p(\mathbf{R},u)} [\log q_\Psi(u|\mathbf{R})], \tag{9}$$

where $q_\Psi(u|\mathbf{R})$ is an arbitrary posterior distribution.

Remark. Similarly, Equation (9) can be interpreted as an *adversarial game* between a property inference adversary q_Ψ who aims to infer u from the dataset representations $\mathbf{R}$ 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-max objective function:

$$\max_f I(\mathbf{y}; \mathbf{R}) \Longleftrightarrow \max_f \max_\Omega \mathop{\mathbb{E}}_{p(\mathbf{y},\mathbf{R})} [\log q_\Omega(\mathbf{y}|\mathbf{R})], \tag{10}$$

where q_Ω is an *arbitrary* posterior distribution that aims to predict each label $y \in \mathbf{y}$ from the data representation $\mathbf{r} \in \mathbf{R}$.

Remark. Equation (10) can be interpreted as a *cooperative game* between f and q_Ω 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:

$$\max_f \Big(\lambda \min_\Psi - \mathop{\mathbb{E}}_{p(\mathbf{X},u)} [\log q_\Psi(u|f(\mathbf{X}))] + (1-\lambda) \max_\Omega \mathop{\mathbb{E}}_{p(\mathbf{X},\mathbf{y})} [\log q_\Omega(\mathbf{y}|f(\mathbf{X}))]\Big), \tag{11}$$

where $\lambda \in [0, 1]$ tradeoffs between privacy and utility. That is, a larger/smaller λ 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_Ψ, h_Ψ, q_Ω, h_Ω, etc.

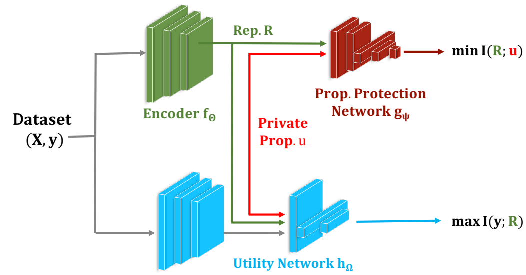


Figure 2: `Inf`^{2}`Guard` against PIAs.

3.2.3 Implementation in practice

Equation (11) is solved via three parameterized neural networks (i.e., the encoder f_Θ, the property protection network g_Ψ associated with q_Ψ, and the utility preservation network h_Ω associated with q_Ω) 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 $\{D_j = (\mathbf{X}_j, \mathbf{y}_j)\}_j$ 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_Ψ: We approximate the first expectation w.r.t. q_Ψ as

$$\mathop{\mathbb{E}}_{p(\mathbf{X},u)} \log q_\Psi(u|f(\mathbf{X})) \approx - \sum_{\{\mathbf{X}_j \in D_j\}} H(u_j, g_\Psi(f(\mathbf{X}_j))), \tag{12}$$

where $f(\mathbf{X}_j)$ is the aggregated representation of a dataset $\mathbf{X}_j$, i.e., $f(\mathbf{X}_j) = \text{Agg}(\{f(\mathbf{x})\}_{\mathbf{x} \in \mathbf{X}_j})$. 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 utility preservation network h_Ω: Similarly, we approximate the second expectation w.r.t. q_Ω as:

$$\mathop{\mathbb{E}}_{p(\mathbf{X},\mathbf{y})} \log q_\Omega(\mathbf{y}|f(\mathbf{X})) \approx - \sum_{\{D_j\}} \sum_{(\mathbf{x}_i, y_i) \in D_j} H(y_i, h_\Omega(f(\mathbf{x}_i))), \tag{13}$$

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.

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 $\mathbf{x} \sim \mathcal{D}$ with a label y, the defender learns a representation $\mathbf{r} = f(\mathbf{x})$ such that when $\mathbf{r}$ is perturbed by certain perturbation (denoted as $\boldsymbol{\delta}$), the shared perturbed representation $\mathbf{r}+\boldsymbol{\delta}$ cannot be used to well recover $\mathbf{x}$, 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.** $\mathbf{r}+\boldsymbol{\delta}$ contains as less information as possible about $\mathbf{x}$. Moreover, the perturbation $\boldsymbol{\delta}$ should be effective enough. Hence, we require $\boldsymbol{\delta}$ can cover all directions of $\mathbf{x}$, and force the entropy of $\boldsymbol{\delta}$ to be as large as possible. Formally, we quantify the data reconstruction protection using the below MI objective:

$$\min_{f, p(\boldsymbol{\delta}) \sim \mathcal{P}} I(\mathbf{r}+\boldsymbol{\delta}; \mathbf{x}) - H(\boldsymbol{\delta}), \tag{14}$$

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

$$\max_{f, p(\boldsymbol{\delta}) \sim \mathcal{P}} I(\mathbf{r}+\boldsymbol{\delta}; y) + I(\mathbf{r}; y), \tag{15}$$

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:

$$\min_{f, p(\boldsymbol{\delta}) \sim \mathcal{P}} I(\mathbf{r}+\boldsymbol{\delta}; \mathbf{x}) - \alpha H(\boldsymbol{\delta}) \tag{16}$$
$$\iff \min_{f, p(\boldsymbol{\delta}) \sim \mathcal{P}} \big(\max_{\Psi} \mathop{\mathbb{E}}_{p(\mathbf{r}, \boldsymbol{\delta}, \mathbf{x})} [\log q_{\Psi}(\mathbf{x} | \mathbf{r}+\boldsymbol{\delta})] - \alpha H(\boldsymbol{\delta}) \big).$$

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

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

$$I(\mathbf{r}; y) = \mathop{\mathbb{E}}_{p(y, \mathbf{r})} [\log q_{\Omega}(y|\mathbf{r}) / p(y)] + \mathop{\mathbb{E}}_{p(\mathbf{r})} [KL(p(y|\mathbf{r}) || q_{\Omega}(y||\mathbf{r}))]$$
$$\geq \mathop{\mathbb{E}}_{p(y, \mathbf{r})} [\log q_{\Omega}(y|\mathbf{r})] + H(y), \tag{17}$$

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

We have a similar form for the MI $I(\mathbf{r}+\boldsymbol{\delta}; y)$ as below

$$I(\mathbf{r}+\boldsymbol{\delta}; y) \geq \max_{p(\boldsymbol{\delta}) \sim \mathcal{P}} \mathop{\mathbb{E}}_{p(y, \mathbf{r}, \boldsymbol{\delta})} [\log q_{\Omega}(y | \mathbf{r}+\boldsymbol{\delta})] + H(y), \tag{18}$$

where we use the same q_{Ω} to predict the label y from the perturbed representation $\mathbf{r}+\boldsymbol{\delta}$.

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

$$\max_{f, p(\boldsymbol{\delta}) \sim \mathcal{P}} I(\mathbf{r}+\boldsymbol{\delta}; y) + I(\mathbf{r}; y) \tag{19}$$
$$\iff \max_{f, \Omega} \big(\max_{p(\boldsymbol{\delta}) \sim \mathcal{P}} \mathop{\mathbb{E}}_{p(y, \mathbf{r}, \boldsymbol{\delta})} [\log q_{\Omega}(y | \mathbf{r}+\boldsymbol{\delta})] + \mathop{\mathbb{E}}_{p(y, \mathbf{r})} [\log q_{\Omega}(y | \mathbf{r})] \big).$$

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

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

$$\max_{f, p(\boldsymbol{\delta}) \sim \mathcal{P}} \Big(\lambda \big(\min_{\Psi} - \mathop{\mathbb{E}}_{p(\mathbf{x}, \boldsymbol{\delta})} [\log q_{\Psi}(\mathbf{x} | f(\mathbf{x}) + \boldsymbol{\delta})] + \alpha H(\boldsymbol{\delta}) \big) \tag{20}$$
$$+ (1-\lambda) \big(\max_{\Omega} \mathop{\mathbb{E}}_{p(\mathbf{x}, \boldsymbol{\delta}, y)} [\log q_{\Omega}(y | f(\mathbf{x}) + \boldsymbol{\delta})] + \mathop{\mathbb{E}}_{p(\mathbf{x}, y)} [\log q_{\Omega}(y | f(\mathbf{x}))] \big) \Big),$$

where $\lambda \in [0,1]$ tradeoffs privacy and utility. A larger λ implies less data features can be inferred through the perturbed representation, while a smaller λ 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 $p(\boldsymbol{\delta})$ in Equation (20). Directly specifying the optimal perturbation distribution is challenging. Motivated by variational inference [36], we propose to parameterize $p(\boldsymbol{\delta})$ with trainable parameters, e.g., Φ. Then the optimization problem w.r.t. the perturbation $\boldsymbol{\delta}$ can be converted to be w.r.t. the parameters Φ, 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 $\boldsymbol{\delta}$ such that the reparameterization trick can be used in training. For instance, when considering $p_{\Phi}(\boldsymbol{\delta})$ as a Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}^2)$, we can reparameterize $\boldsymbol{\delta}$ (with a scale ε) as:

$$\boldsymbol{\delta} = \varepsilon \cdot \tanh(\mathbf{u}), \quad \mathbf{u} \sim \mathcal{N}(\boldsymbol{\mu}, \mathrm{diag}(\boldsymbol{\sigma}^2)), \tag{21}$$

That is, it first samples $\mathbf{u}$ from a diagonal Gaussian with a mean vector $\boldsymbol{\mu}$ and standard deviation vector $\boldsymbol{\sigma}$, and $\boldsymbol{\delta}$ is obtained by compressing $\mathbf{u}$ to be $[-1,1]$ via the $\tanh(\cdot)$ function and multiplying ε. $\Phi = (\boldsymbol{\mu}, \boldsymbol{\sigma})$ 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 g_{Ψ}, and utility preservation network h_{Ω}) using data samples from certain data distribution. Suppose we are given a set of data samples $D = \{\mathbf{x}_j, y_j\}$.

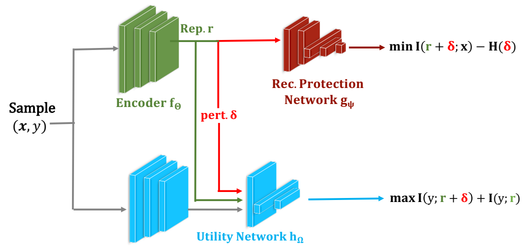


Figure 3: Inf^{2}Guard against DRAs.

Learning the data reconstruction network g_Ψ: As $\mathbf{x}$ and its representation $\mathbf{r}$ 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 Φ. We can approximate the expectation w.r.t. g_Ψ as

$$\begin{aligned}
&\mathop{\mathbb{E}}_{p(\mathbf{x}),p_\Phi(\boldsymbol{\delta})} \log q_\Psi(\mathbf{x}|f(\mathbf{x})+\boldsymbol{\delta}) = I^{(JSD)}_{\Theta,\Psi}(\mathbf{x}; f_\Theta(\mathbf{x})+\boldsymbol{\delta}) \\
&\approx \sum_{\mathbf{x}_j\in D,\boldsymbol{\delta}_j\sim p_\Phi(\boldsymbol{\delta})} [-\mathrm{sp}(-h_\Psi(\mathbf{x}_j, f_\Theta(\mathbf{x}_j)+\boldsymbol{\delta}_j))] \\
&\quad - \sum_{(\mathbf{x}_j,\mathbf{x}'_j)\in D,\boldsymbol{\delta}_j\sim p_\Phi(\boldsymbol{\delta})} [\mathrm{sp}(h_\Psi(\mathbf{x}'_j, f_\Theta(\mathbf{x}_j)+\boldsymbol{\delta}_j))], \qquad (22)
\end{aligned}$$

where $\mathbf{x}'_j$ is an independent and random sample from the same distribution as $\mathbf{x}_j$, and $\mathrm{sp}(z) = \log(1+\exp(z))$ is the softplus function. We maximize $I^{(JSD)}_{\Theta,\Psi}$ to update g_Ψ.

Learning the utility preservation network h_Ω: We first estimate the below expectation:

$$\mathop{\mathbb{E}}_{p(\mathbf{x},y)p_\Phi(\boldsymbol{\delta})} \log q_\Omega(y|f(\mathbf{x})+\boldsymbol{\delta}) \approx -\sum_{(\mathbf{x}_j,y_j)\in D,\boldsymbol{\delta}_j\sim p_\Phi(\boldsymbol{\delta})} H(y_j, h_\Omega(f(\mathbf{x}_j)+\boldsymbol{\delta}_j)). \qquad (23)$$

Similarly, we can approximate the third expectation as:

$$\mathop{\mathbb{E}}_{p(\mathbf{x},y)} \log q_\Omega(y|f(\mathbf{x})) \approx -\sum_{(\mathbf{x}_j,y_j)\in D} H(y_j, h_\Omega(f(\mathbf{x}_j))). \qquad (24)$$

We minimize the two cross entropy losses to update h_Ω.

Updating the distribution parameter Φ: Due to the reparameterization trick, the gradient can be back-propagated from each $\boldsymbol{\delta}_j$ to the parameters Φ. For simplicity, we do not consider the JSD term in Equation (22) due to its complexity. Then we have the terms relevant to Φ as below:

$$\mathop{\mathbb{E}}_{z\sim\mathcal{N}(0,1)} \sum_{(\mathbf{x}_j,y_j)} H(y_j, h_\Omega(f(\mathbf{x}_j)+\varepsilon\cdot\tanh(\boldsymbol{\mu}+\boldsymbol{\sigma}\mathbf{z}))) - \beta\cdot H(\varepsilon\cdot\tanh(\boldsymbol{\mu}+\boldsymbol{\sigma}\mathbf{z})), \qquad (25)$$

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. Φ in each term can be calculated. In practice, we approximate the expectation on $\mathbf{z}$ with (e.g., 5) Monte Carlo samples, and perform the *stochastic gradient descent* to update Φ. Details on updating Φ are in Algorithm 3. With Φ, we use it to generate $\boldsymbol{\delta}$ and add it to $\mathbf{r}$ to produce the perturbed representation.

Learning the encoder f. Finally, after updating g_Ψ, h_Ω, and Φ, 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 $\mathcal{A}_{MIA}$ be the set of all MIAs $\mathcal{A}_{MIA} = \{A_{MIA} : \mathcal{Z} \to u \in \{0,1\}\}$ that have access to the representations $\mathbf{r}$ by querying f with data $\mathbf{x}$ from the distribution $\mathcal{D}$. The MIA accuracy is bounded as below:

Theorem 1. *Let f be the learnt encoder by Equation (6) over a data distribution $\mathcal{D} \subset \mathcal{X}$. For a random data sample $\mathbf{x} \sim \mathcal{D}$ with the learnt representation $\mathbf{r} = f(\mathbf{x})$ and membership u, we have $Pr(A_{MIA}(\mathbf{r}) = u) \le 1 - \frac{H(u|\mathbf{r})}{2\log_2(6/H(u|\mathbf{r}))}, \forall A_{MIA} \in \mathcal{A}_{MIA}$.*

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

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

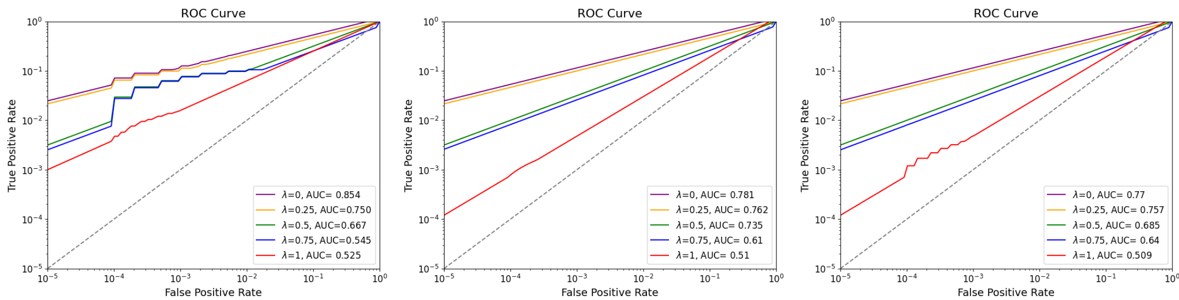
Theorem 2. *Let f be the learnt encoder by Equation (11) over a data distribution $\mathcal{D}$. For a random dataset $\mathbf{X} \sim \mathcal{D}$ with the learnt representation $\mathbf{R} = f(\mathbf{X})$ and dataset property u, we have $Pr(A_{PIA}(\mathbf{R}) = u) \le 1 - \frac{H(u|\mathbf{R})}{2\log_2(6/H(u|\mathbf{R}))}, \forall A_{PIA} \in \mathcal{A}_{PIA}$.*

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

Guaranteed privacy leakage of DRAs: Let $\mathcal{A}_{DRA}$ be the set of all DRAs that have access to the perturbed data representations, i.e., $\mathcal{A}_{DRA} = \{A_{DRA} : \mathbf{r}+\boldsymbol{\delta} \in \mathcal{Z}\times\mathcal{P} \to \mathbf{x} \in \mathcal{X}\}$. An ℓ_p-norm ball centered at a point $\mathbf{v}$ with a radius ρ is denoted as $\mathcal{B}_p(\mathbf{v},\rho)$, i.e., $\mathcal{B}_p(\mathbf{v},\rho) = \{\mathbf{v}' : \|\mathbf{v}'-\mathbf{v}\|_p \le \rho\}$. For a space $\mathcal{S}$, we denote its boundary as $\partial\mathcal{S}$, whose volume is denoted as $\mathrm{Vol}(\partial\mathcal{S})$. Then the reconstruction error (in terms of ℓ_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 $\mathcal{D} \subset \mathcal{X}$ and $\boldsymbol{\delta}$ be the perturbation for a random sample $\mathbf{x} \sim \mathcal{X}$. Then, $Pr(\|A_{DRA}(\mathbf{r}+\boldsymbol{\delta}) - \mathbf{x}\|_p \ge \eta) \ge 1 - \frac{I(\mathbf{x};\mathbf{r}+\boldsymbol{\delta})+\log 2}{\log Vol(\partial\mathcal{X}) - \log Vol(\partial\mathcal{X}(\eta))}, \forall A_{DRA} \in \mathcal{A}_{DRA}$, where $Vol(\partial\mathcal{X}(\eta)) = \max_{\mathbf{x}\in\mathcal{X}} Vol(\partial\mathcal{B}_p(\mathbf{x},\eta)\cap\mathcal{X})$.*

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



(a) CIFAR10 (b) Purchase100 (c) Texas100

Figure 4: TPR vs FPR of `Inf`^{2}`Guard` against LiRA on different λ's.

5 Evaluations

In this section, we will evaluate `Inf`^{2}`Guard` against the MIAs, PIAs, and DRAs on benchmark datasets. `Inf`^{2}`Guard` 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 `Inf`^{2}`Guard` 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].

Table 1: `Inf`^{2}`Guard` 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%.

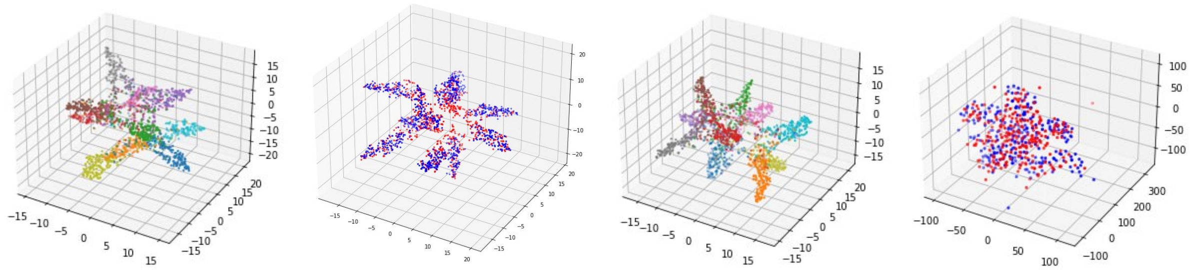
CIFAR10			Purchase100			Texas100		
λ	Utility	MIA Acc	λ	Utility	MIA Acc	λ	Utility	MIA Acc
0	78.9%	70.1%	0	81.7%	68.4%	0	49.9%	70.2%
0.25	78.2%	55.9%	0.25	80.9%	60%	0.25	49.1%	61%
0.5	78%	53.5%	0.5	80%	51%	0.5	47%	53%
0.75	77.2%	51.1%	0.75	78%	50%	0.75	46%	50%
1	20%	50%	1	20%	50%	1	2%	50%

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 `Inf`^{2}`Guard` 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 `Inf`^{2}`Guard`) 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

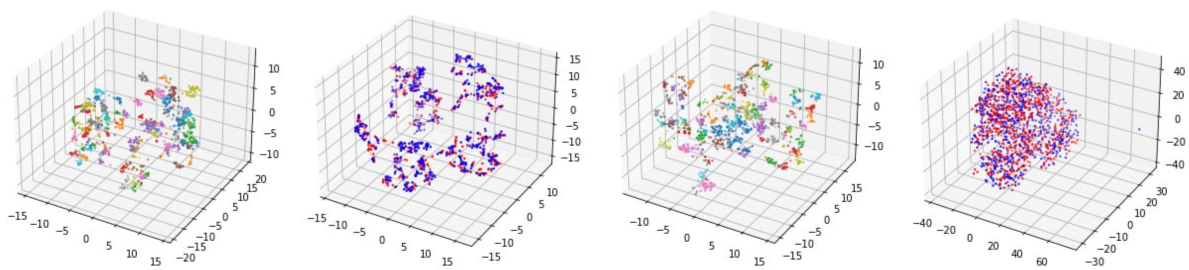
Utility-privacy results: According to Equation (6), $\lambda = 0$ indicates no privacy protection. Increasing λ's value enhances `Inf`^{2}`Guard`'s resilience against MIAs. $\lambda = 1$ means the maximum privacy protection without preserving utility. Table 1 shows the utility-MIA Accuracy results of `Inf`^{2}`Guard`. 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 $0 < \lambda < 1$, `Inf`^{2}`Guard` 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 `Inf`^{2}`Guard` 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 λ. This implies that `Inf`^{2}`Guard` indeed learns the representations that can defend against LiRA to some extent.



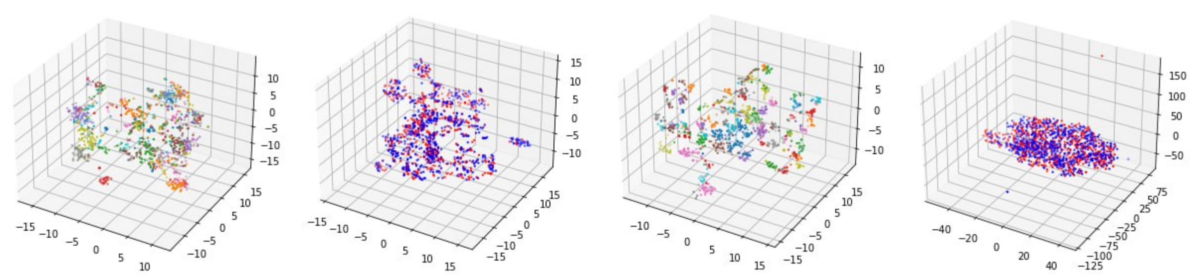
(a) Utility w/o. defense (78.9%) (b) MIA Acc w/o. defense (70.1%) (c) Utility w. defense (77.2%) (d) MIA Acc w. defense (51.1%)

Figure 5: Inf^{2}Guard against MIAs: 3D t-SNE embeddings results on the learnt representation of on CIFAR10.



(a) Utility w/o. defense (81.7%) (b) MIA Acc w/o. defense (68.4%) (c) Utility w. defense (80%) (d) MIA Acc w. defense (51%)

Figure 6: Inf^{2}Guard against MIAs: 3D t-SNE embeddings results on the learnt representation on Purchase100.



(a) Utility w/o. defense (49.8%) (b) MIA Acc w/o. defense (70.2%) (c) Utility w. defense (46%) (d) MIA Acc w. defense (50%)

Figure 7: Inf^{2}Guard against MIAs: 3D t-SNE embeddings results on the learnt representation on Texas100.

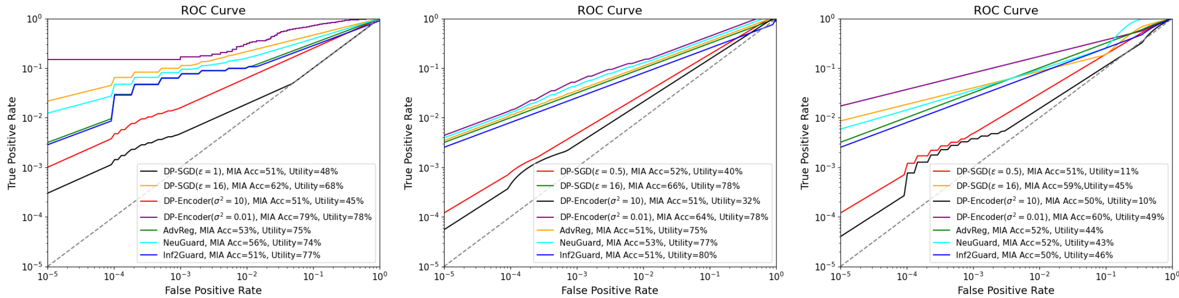
Visualizing the learnt representations: To better understand the learnt representations by Inf^{2}Guard, we adopt the t-SNE algorithm [64] to visualize the low-dimensional embeddings of them. λ is chosen in Table 1 that achieves the best utility-privacy tradeoff. We also compare with the case without 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 Inf^{2}Guard 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

Table 2: Comparing Inf^{2}Guard with existing defenses against MIAs on the three datasets. DP methods are under the same/close defense performance as Inf^{2}Guard.

Defense	CIFAR10		Purchase100		Texas100	
	Utility	MIA Acc	Utility	MIA Acc	Utility	MIA Acc
DP-SGD	48%	51%	40%	52%	11%	51%
DP-enc	45%	51%	32%	51%	10%	50%
AdvReg	75%	53%	75%	51%	44%	52%
NeuGuard	74%	56%	77%	53%	43%	52%
Inf^{2}Guard	77%	51%	80%	51%	46%	50%

of Inf^{2}Guard). 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 defense performance (w.r.t. MIA accuracy) as Inf^{2}Guard. AdvReg and NeuGuard also perform worse than Inf^{2}Guard.



(a) CIFAR10 (b) Purchase100 (c) Texas100

Figure 8: Comparing `Inf`^{2}`Guard` ($\lambda = 0.75$) with the existing defenses against LiRA.

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 `Inf`^{2}`Guard`. With an MIA accuracy close to random guessing (but low utility), we see 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 `Inf`^{2}`Guard`, their TPRs are much higher than `Inf`^{2}`Guard`'s at a low FPR. Besides, `Inf`^{2}`Guard` has smaller TPRs than AdvReg and NeuGuard.

Overhead comparison: All MIA defenses train a task classifier. AdvReg trains a task classifier and membership inference network. NeuGuard trains a task classifier with two regularizations. DP-SGD trains the task classifier on noisy models, while DP-encoder normally trains the encoder first and then trains the utility network on (Gaussian) noisy representations. In the experiments, we define the task classifier of the compared defenses as the concatenation of our encoder and utility network. In our platform (NVIDIA GeForce RTX 3070 Ti), it took `Inf`^{2}`Guard` (72,7,6), AdvReg (66,6,6), NeuGuard (62,5,5), DP-SGD (60,4,3) and DP-encoder (59,4,3) seconds to run each iteration on the three datasets, respectively[9].

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.

[9]We have similar conclusions on defending against the other two attacks.

Table 3: `Inf`^{2}`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.

Census			RSNA			CelebA		
λ	Utility	PIA Acc	λ	Utility	PIA Acc	λ	Utility	PIA Acc
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%

- **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.

5.2.2 Experimental results

Utility-privacy results: Table 3 shows the utility-privacy results of `Inf`^{2}`Guard`, where the encoder uses a mean-aggregator (i.e., average the representations of a subset of data. Note different subsets have different sizes). We have similar observations as in defending against MIAs: 1) The PIA accuracy can be as large as 68% without privacy protection ($\lambda = 0$ in Equation (11)), implying the PIA is effective; 2) When focusing on protecting privacy ($\lambda = 1$), the PIA performance can be largely reduced. However, the utility is also significantly decreased, e.g., from 85% to 45%. 3) Utility and privacy show a tradeoff w.r.t. $0 < \lambda < 1$. In most of the cases, the best tradeoff is obtained when $\lambda = 0.75$. Again, the results show the learnt privacy-preserving encoder/representations are effective against PIAs, and also maintain utility.

Table 4: Inf^{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.

Census			RSNA			CelebA		
λ	Utility	PIA Acc	λ	Utility	PIA Acc	λ	Utility	PIA Acc
0	85%	65%	0	83%	52%	0	91%	50%
0.25	83%	51%	0.25	82%	24%	0.25	91%	25%
0.5	79%	49%	0.5	82%	24%	0.5	91%	15%
0.75	77%	37%	0.75	81%	21%	0.75	88%	15%
1	47%	30%	1	50%	16%	1	53%	9%

Table 5: Comparing Inf^{2}Guard with DP against PIAs.

Defense	Census		RSNA		CelebA	
	Utility	PIA Acc	Utility	PIA Acc	Utility	PIA Acc
DP-encoder	52%	34%	57%	19%	66%	11%
Inf^{2}Guard	76%	34%	80%	19%	89%	11%

Visualizing the learnt representations: Figures 12-14 in Appendix B show the 3D t-SNE embeddings of the learnt representations with $\lambda = 0, 0.75$. 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 Inf^{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 Inf^{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, Inf^{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 MIAs. 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 Inf^{2}Guard, DP has much worse utility.

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 Inf^{2}Guard against DRAs.

Defense/attack training and testing: Table 11 in Appendix 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: Inf^{2}Guard results against DRAs. A smaller SSIM or PSNR indicates better defense performance (λ =0.4).

CIFAR10			CIFAR100			Activity		
Scale ε	Utility	SSIM/PSNR	ε	Utility	SSIM/PSNR	ε	Utility	MSE
0	89.5%	0.78 / 15.97	0	52.7%	0.92 / 22.79	0	95.1%	0.81
0.75	85.2%	0.42 / 12.09	0.75	49.1%	0.36 / 13.36	0.5	90.1%	1.06
1.25	**78.0%**	**0.21 / 11.87**	1.00	**46.5%**	**0.19 / 12.70**	1.0	**85.6%**	**1.32**
1.75	68.9%	0.17 / 11.21	1.25	43.3%	0.14 / 12.29	1.5	81.0%	1.64

Table 7: Impact of λ on Inf^{2}Guard against DRAs ($\varepsilon = 1.25$).

CIFAR10			CIFAR100			Activity		
λ	Utility	SSIM/PSNR	λ	Utility	SSIM/PSNR	λ	Utility	MSE
0.1	83.2%	0.52 / 12.79	0.1	46.8%	0.46 / 14.75	0.1	90.0%	1.25
0.4	78.0%	0.21 / 11.87	0.4	46.5%	0.21 / 12.70	0.4	85.6%	1.32
0.7	67.9%	0.15 / 11.43	0.7	46.5%	0.20 / 12.42	0.7	85.0%	1.62

- **Attack training:** We mimic the strongest DRA, where the attacker knows the reconstruction protection network, training set, and perturbation distribution. S/he 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

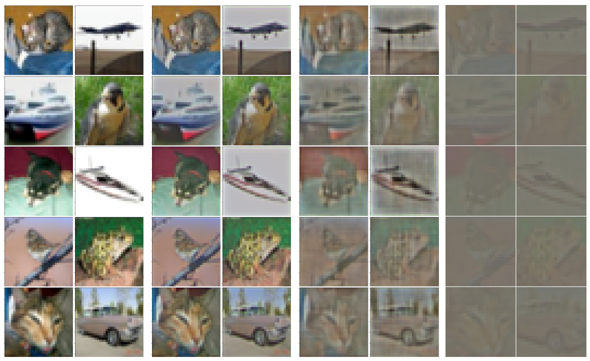
Utility-privacy results: Table 6 shows the defense results of Inf^{2}Guard with the Gaussian perturbation distribution, where $\lambda = 0.4$ in Equation (20). We can observe ε acts a utility-privacy tradeoff. A larger ε implies adding more perturbation to the representation during defense training.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

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

Comparing with the DP-based defense: All empirical defenses against DRAs are broken are 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 Inf^{2}Guard with DP and Table 8 shows the DP results. Viewing with results in Table 6, we see Inf^{2}Guard obtains better utility-privacy tradeoffs than DP-SGD.

Table 8: DP-SGD defense results against DRAs. A smaller SSIM or PSNR indicates better defense performance.

CIFAR10			CIFAR100			Activity		
Scale ε	Utility	SSIM/PSNR	ε	Utility	SSIM/PSNR	ε	Utility	MSE
0	89.5%	0.78 / 15.97	0	52.7%	0.92 / 22.79	0	95.1%	0.81
0.75	85.6%	0.50 / 13.21	0.75	49.5%	0.43 / 13.77	0.5	91.8%	0.88
1.25	**77.4%**	**0.37 / 12.45**	1.00	**46.7%**	**0.24 / 12.87**	1.0	90.1%	0.90
1.75	65.3%	0.36 / 12.35	1.25	43.3%	0.18 / 12.54	1.5	**84.1%**	**1.01**



(a) Raw images (b) No defense (c) DP (d) Inf^{2}Guard

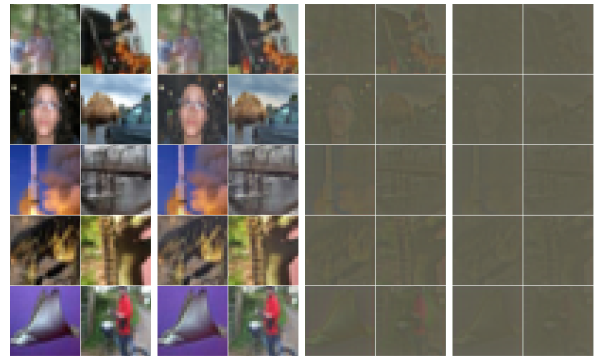
Figure 9: Raw images vs. reconstructions on CIFAR10. DP Utility: 77%, Inf^{2}Guard Utility: 78%.

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, Inf^{2}Guard 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 Inf^{2}Guard and that by DP to the true feature. A (larger) positive value implies Inf^{2}Guard is (more) dissimilar than DP to the true feature. We can see Inf^{2}Guard has better defense results than DP in most (413 out of 516) of the features.

6 Discussion and Future Work

Inf^{2}Guard and DP: Essentially, Inf^{2}Guard 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 Inf^{2}Guard 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 Inf^{2}Guard, which we will explore in the future. More importantly, our Inf^{2}Guard can complement DP. For instance, we can use the learnt (deterministic) data representations by Inf^{2}Guard 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 $\mathbf{r}$ includes as much information about the *training* sample $\mathbf{x}$ as possible (i.e., $u = 1$). Intuitively, when $\mathbf{r}$ retains all information about $\mathbf{x}$, the model trained on $\mathbf{r}$ will have the same performance as trained on the raw $\mathbf{x}$, despite the learning task. Formally, the MI objective becomes $\max_f I(\mathbf{x};\mathbf{r}|u=1)$.



(a) Raw images (b) No defense (c) DP (d) Inf^{2}Guard

Figure 10: Raw images vs. reconstructions on CIFAR100. DP Utility: 47%, Inf^{2}Guard Utility: 47%. Better zoom in.

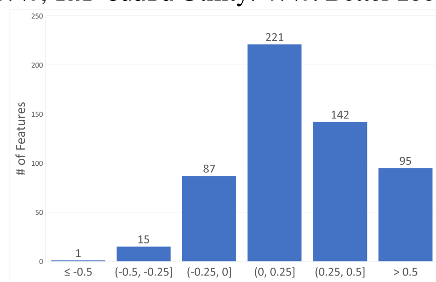


Figure 11: Inf^{2}Guard vs. DP on 50 samples in Activity (both utility=86%). We count #features located in each bin. The value range in each bin means the difference between the reconstructed feature by Inf^{2}Guard and that by DP to the true feature. A positive value implies Inf^{2}Guard produces more dissimilar reconstruction than DP.

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], L_2 norm regularization [58], model stacking [54], adversary 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 [22,74], batch normalization statistics [74]) into the training data, or introduce an auxiliary dataset to simulate the training data distribution [28,34,67] (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 [9]. 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 `Inf`^{2}`Guard`, 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 `Inf`^{2}`Guard` 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

[1] Chatgpt. https://chat.openai.com/. developed by OpenAI.

[2] Palm 2. https://ai.google/discover/palm2/. developed by Goolge.

[3] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *CCS*, 2016.

[4] Alexander A Alemi, Ian Fischer, Joshua V Dillon, and Kevin Murphy. Deep variational information bottleneck. In *ICLR*, 2017.

[5] Caridad Arroyo Arevalo, Sayedeh Leila Noorbakhsh, Yun Dong, Yuan Hong, and Binghui Wang. Task-agnostic privacy-preserving representation learning for federated learning against attribute inference attacks. In *AAAI*, 2024.

[6] Giuseppe Ateniese, Luigi V Mancini, Angelo Spognardi, Antonio Villani, Domenico Vitali, and Giovanni Felici. Hacking smart machines with smarter ones: How to extract meaningful data from machine learning classifiers. *IJSN*, 2015.

[7] Borja Balle, Giovanni Cherubin, and Jamie Hayes. Reconstructing training data with informed adversaries. In *IEEE SP*, 2022.

[8] Mislav Balunovic, Dimitar Dimitrov, Nikola Jovanović, and Martin Vechev. Lamp: Extracting text from gradients with language model priors. In *NeurIPS*, 2022.

[9] Mislav Balunovic, Dimitar Iliev Dimitrov, Robin Staab, and Martin Vechev. Bayesian framework for gradient leakage. In *ICLR*, 2022.

[10] Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeshwar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and Devon Hjelm. Mutual information neural estimation. In *ICML*, 2018.

[11] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE TPMAI*, 2013.

[12] Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis, and Florian Tramer. Membership inference attacks from first principles. In *IEEE SP*, 2022.

[13] Harsh Chaudhari, John Abascal, Alina Oprea, Matthew Jagielski, Florian Tramèr, and Jonathan Ullman. Snap: Efficient extraction of private properties with poisoning. In *IEEE SP*, 2023.

[14] Dingfan Chen, Ning Yu, Yang Zhang, and Mario Fritz. Gan-leaks: A taxonomy of membership inference attacks against generative models. In *CCS*, 2020.

[15] Pengyu Cheng, Weituo Hao, Shuyang Dai, Jiachang Liu, Zhe Gan, and Lawrence Carin. Club: A contrastive log-ratio upper bound of mutual information. In *ICML*, 2020.

[16] Christopher A Choquette-Choo, Florian Tramer, Nicholas Carlini, and Nicolas Papernot. Label-only membership inference attacks. In *ICML*, 2021.

[17] Clarifai. https://www.clarifai.com/demo. July 2019.

[18] Cynthia Dwork. Differential privacy. In *ICALP*, 2006.

[19] Liam H Fowl, Jonas Geiping, Wojciech Czaja, Micah Goldblum, and Tom Goldstein. Robbing the fed: Directly obtaining private data in federated learning with modified models. In *ICLR*, 2022.

[20] Karan Ganju, Qi Wang, Wei Yang, Carl A Gunter, and Nikita Borisov. Property inference attacks on fully connected neural networks using permutation invariant representations. In *CCS*, 2018.

[21] Wei Gao, Shangwei Guo, Tianwei Zhang, Han Qiu, Yonggang Wen, and Yang Liu. Privacy-preserving collaborative learning with automatic transformation search. In *CVPR*, 2021.

[22] Jonas Geiping, Hartmut Bauermeister, Hannah Dröge, and Michael Moeller. Inverting gradients–how easy is it to break privacy in federated learning? In *NeurIPS*, 2020.

[23] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NIPS*, 2014.

[24] Divya Gopinath, Hayes Converse, Corina Pasareanu, and Ankur Taly. Property inference for deep neural networks. In *ASE*, 2019.

[25] Jihun Hamm, Yingjun Cao, and Mikhail Belkin. Learning privately from multiparty data. In *ICML*, 2016.

[26] Valentin Hartmann, Léo Meynent, Maxime Peyrard, Dimitrios Dimitriadis, Shruti Tople, and Robert West. Distribution inference risks: Identifying and mitigating sources of leakage. In *IEEE SaTML*, 2023.

[27] Zecheng He, Tianwei Zhang, and Ruby B Lee. Model inversion attacks against collaborative inference. In *ACSAC*, 2019.

[28] Briland Hitaj, Giuseppe Ateniese, and Fernando Perez-Cruz. Deep models under the gan: information leakage from collaborative deep learning. In *CCS*, 2017.

[29] R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. In *ICLR*, 2019.

[30] Nils Homer, Szabolcs Szelinger, Margot Redman, David Duggan, Waibhav Tembe, Jill Muehling, John V Pearson, Dietrich A Stephan, Stanley F Nelson, and David W Craig. Resolving individuals contributing trace amounts of dna to highly complex mixtures using high-density snp genotyping microarrays. *PLoS genetics*.

[31] Bo Hui, Yuchen Yang, Haolin Yuan, Philippe Burlina, Neil Zhenqiang Gong, and Yinzhi Cao. Practical blind membership inference attack via differential comparisons. In *NDSS*, 2021.

[32] Roger Iyengar, Joseph P Near, Dawn Song, Om Thakkar, Abhradeep Thakurta, and Lun Wang. Towards practical differentially private convex optimization. In *IEEE SP*, 2019.

[33] Bargav Jayaraman and David Evans. Evaluating differentially private machine learning in practice. In *USENIX Security*, 2019.

[34] Jinwoo Jeon, Jaechang Kim, Kangwook Lee, Sewoong Oh, and Jungseul Ok. Gradient inversion with generative image prior. In *NeurIPS*, 2021.

[35] Jinyuan Jia, Ahmed Salem, Michael Backes, Yang Zhang, and Neil Zhenqiang Gong. Memguard: Defending against black-box membership inference attacks via adversarial examples. In *CCS*, 2019.

[36] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *ICLR*, 2014.

[37] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009.

[38] Hongkyu Lee, Jeehyeong Kim, Seyoung Ahn, Rasheed Hussain, Sunghyun Cho, and Junggab Son. Digestive neural networks: A novel defense strategy against inference attacks in federated learning. *computers & security*, 2021.

[39] Klas Leino and Matt Fredrikson. Stolen memories: Leveraging model memorization for calibrated white-box membership inference. In *Usenix Security)*, 2020.

[40] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *ICCV*, 2015.

[41] Saeed Mahloujifar, Esha Ghosh, and Melissa Chase. Property inference from poisoning. In *IEEE SP*, 2022.

[42] Pratyush Maini, Mohammad Yaghini, and Nicolas Papernot. Dataset inference: Ownership resolution in machine learning. In *ICLR*, 2021.

[43] Ilya Mironov. Rényi differential privacy. In *30th IEEE Computer Security Foundations Symposium, CSF 2017, Santa Barbara, CA, USA, August 21-25, 2017*, pages 263–275. IEEE Computer Society, 2017.

[44] Meisam Mohammady, Shangyu Xie, Yuan Hong, Mengyuan Zhang, Lingyu Wang, Makan Pourzandi, and Mourad Debbabi. R2DP: A universal and automated approach to optimizing the randomization mechanisms of differential privacy for utility metrics with no known optimal distributions. In *CCS*, 2020.

[45] Milad Nasr, Reza Shokri, and Amir Houmansadr. Machine learning with membership privacy using adversarial regularization. In *CCS*, 2018.

[46] Sebastian Nowozin, Botond Cseke, and Ryota Tomioka. f-gan: Training generative neural samplers using variational divergence minimization. In *NIPS*, 2016.

[47] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv*, 2018.

[48] Manas Pathak, Shantanu Rane, and Bhiksha Raj. Multiparty differential privacy via aggregation of locally trained classifiers. In *NeurIPS*, 2010.

[49] Xue Bin Peng, Angjoo Kanazawa, Sam Toyer, Pieter Abbeel, and Sergey Levine. Variational discriminator bottleneck: Improving imitation learning, inverse rl, and gans by constraining information flow. *arXiv preprint arXiv:1810.00821*, 2018.

[50] Ben Poole, Sherjil Ozair, Aaron van den Oord, Alexander A Alemi, and George Tucker. On variational bounds of mutual information. In *ICML*, 2019.

[51] Jorge Reyes-Ortiz, Davide Anguita, Alessandro Ghio, Luca Oneto, and Xavier Parra. Human Activity Recognition Using Smartphones. UCI Machine Learning Repository, 2012. DOI: https://doi.org/10.24432/C54S4K.

[52] Alexandre Sablayrolles, Matthijs Douze, Cordelia Schmid, Yann Ollivier, and Hervé Jégou. White-box vs black-box: Bayes optimal strategies for membership inference. In *ICML*, 2019.

[53] Ahmed Salem, Giovanni Cherubin, David Evans, Boris Köpf, Andrew Paverd, Anshuman Suri, Shruti Tople, and Santiago Zanella-Béguelin. Sok: Let the privacy games begin! a unified treatment of data inference privacy in machine learning. In *IEEE SP*, 2023.

[54] Ahmed Salem, Yang Zhang, Mathias Humbert, Pascal Berrang, Mario Fritz, and Michael Backes. Ml-leaks: Model and data independent membership inference attacks and defenses on machine learning models. In *NDSS*, 2018.

[55] Daniel Scheliga, Patrick Mäder, and Marco Seeland. Precode-a generic model extension to prevent deep gradient leakage. In *WACV*, 2022.

[56] Virat Shejwalkar and Amir Houmansadr. Membership privacy for machine learning models through knowledge transfer. In *AAAI*, 2021.

[57] Reza Shokri and Vitaly Shmatikov. Privacy-preserving deep learning. In *CCS*, pages 1310–1321, 2015.

[58] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *IEEE SP*, 2017.

[59] Liwei Song and Prateek Mittal. Systematic evaluation of privacy risks of machine learning models. In *USENIX Security*, 2021.

[60] Liwei Song, Reza Shokri, and Prateek Mittal. Privacy risks of securing machine learning models against adversarial examples. In *CCS*, 2019.

[61] Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *JMLR*, 2014.

[62] Jingwei Sun, Ang Li, Binghui Wang, Huanrui Yang, Hai Li, and Yiran Chen. Provable defense against privacy leakage in federated learning from representation perspective. In *CVPR*, 2021.

[63] Anshuman Suri and David Evans. Formalizing and estimating distribution inference risks. In *PETS*, 2022.

[64] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *JMLR*, 2008.

[65] Han Wang, Jayashree Sharma, Shuya Feng, Kai Shu, and Yuan Hong. A model-agnostic approach to differentially private topic mining. In *KDD*, pages 1835–1845, 2022.

[66] Xiuling Wang and Wendy Hui Wang. Group property inference attacks against graph neural networks. In *CCS*, 2022.

[67] Zhibo Wang, Mengkai Song, Zhifei Zhang, Yang Song, Qian Wang, and Hairong Qi. Beyond inferring class representatives: User-level privacy leakage from federated learning. In *INFOCOM*, 2019.

[68] Zihan Wang, Jason Lee, and Qi Lei. Reconstructing training data from model gradient, provably. In *AISTATS*, 2023.

[69] Kang Wei, Jun Li, Ming Ding, Chuan Ma, Howard H Yang, Farhad Farokhi, Shi Jin, Tony QS Quek, and H Vincent Poor. Federated learning with differential privacy: Algorithms and performance analysis. *IEEE TIFS*, 2020.

[70] Yuxin Wen, Arpit Bansal, Hamid Kazemi, Eitan Borgnia, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Canary in a coalmine: Better membership inference with ensembled adversarial queries. In *ICLR*, 2023.

[71] Nuo Xu, Binghui Wang, Ran Ran, Wujie Wen, and Parv Venkitasubramaniam. Neuguard: Lightweight neuron-guided defense against membership inference attacks. In *ACSAC*, 2022.

[72] Jiayuan Ye, Aadyaa Maddi, Sasi Kumar Murakonda, Vincent Bindschaedler, and Reza Shokri. Enhanced membership inference attacks against machine learning models. In *CCS*, 2022.

[73] Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. Privacy risk in machine learning: Analyzing the connection to overfitting. In *IEEE CSF*, 2018.

[74] Hongxu Yin, Arun Mallya, Arash Vahdat, Jose M Alvarez, Jan Kautz, and Pavlo Molchanov. See through gradients: Image batch recovery via gradinversion. In *CVPR*, 2021.

[75] Lei Yu, Ling Liu, Calton Pu, Mehmet Emre Gursoy, and Stacey Truex. Differentially private model publishing for deep learning. In *IEEE SP*, 2019.

[76] Xiaoyong Yuan and Lan Zhang. Membership inference attacks and defenses in neural network pruning. In *Usenix Security*, 2022.

[77] Wanrong Zhang, Shruti Tople, and Olga Ohrimenko. Leakage of dataset properties in multi-party machine learning. In *USENIX Security*, 2021.

[78] Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. idlg: Improved deep leakage from gradients. *arXiv*, 2020.

[79] Junhao Zhou, Yufei Chen, Chao Shen, and Yang Zhang. Property inference attacks against gans. *NDSS 2022*, 2022.

[80] Junyi Zhu and Matthew Blaschko. R-gap: Recursive gradient attack on privacy. *ICLR*, 2021.

[81] Ligeng Zhu, Zhijian Liu, and Song Han. Deep leakage from gradients. In *NeurIPS*, 2019.

Algorithm 1 `Inf`^{2}`Guard` against MIAs

Input: 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.
Output: Network parameters: Θ, Ψ, Ω.

Initialize Θ, Ψ, Ω for the encoder f, membership protection network g_Ψ, and utility preservation network h_ϕ;
for $t = 1$ to T **do**
  $L_1 = \sum_{(\mathbf{x}_j, u_j) \in D_1 \cup D_0} H(u_j, g_\Psi(f(\mathbf{x}_j)))$;
  $L_2 = \sum_{(\mathbf{x}_j, y_j) \in D_1} H(y_j, h_\Omega(f(\mathbf{x}_j)))$;
  for $i = 1$ to I **do**
    $\Psi \leftarrow \Psi - lr_1 \cdot \frac{\partial L_1}{\partial \Psi}$;
    $\Omega \leftarrow \Omega - lr_2 \cdot \frac{\partial L_2}{\partial \Omega}$;
    $\Theta \leftarrow \Theta + lr_3 \cdot \frac{\partial(\lambda L_1 - (1-\lambda) L_2)}{\partial \Theta}$;

Algorithm 2 `Inf`^{2}`Guard` against PIAs

Input: N datasets $\{D_j\}_{j=1}^N$ sampled from a reference dataset D_r with each D_j having a property value u_j, tradeoff hyperparameter $\lambda \in [0,1]$, learning rates lr_1, lr_2, lr_3; #local gradients I, #global rounds T.
Output: Network parameters: Θ, Ω, Ψ.

Initialize Θ, Ψ, Ω for the encoder f, property protection network g_Ψ, and utility preservation network h_ϕ;
for round $t = 1$ to T **do**
  $L_1 = \sum_{\{(\mathbf{X}_j, \mathbf{y}_j) = D_j\}_j} H(u_j, g_\Psi(f(\mathbf{X}_j)))$;
  $L_2 = \sum_{(\mathbf{x}_i, y_i) \in \bigcup_j D_j} H(y_i, h_\Omega(f(\mathbf{x}_i)))$;
  for $i = 1$ to I **do**
    $\Psi \leftarrow \Psi - lr_1 \cdot \frac{\partial L_1}{\partial \Psi}$;
    $\Omega \leftarrow \Omega - lr_2 \cdot \frac{\partial L_2}{\partial \Omega}$;
    $\Theta \leftarrow \Theta + lr_3 \cdot \frac{\partial(\lambda L_1 - (1-\lambda) L_2}{\partial \Theta}$.

Algorithm 3 Update perturbation distribution parameter Φ

Input: K Monte Carlo samples, the encoder f_Θ in the previous round, objective function Eqn (20). learning rate lr, #epochs I_l
Output: Perturbation distribution parameters Φ

Initialize $\Phi = (\boldsymbol{\mu}, \boldsymbol{\sigma})$.
for $i = 1$ to I_l **do**
  for $j = 1$ to K **do**
    Sample $\mathbf{z}_j$ from $\mathcal{N}(0,1)$ and compute $\boldsymbol{\delta}_j = \boldsymbol{\mu} + \boldsymbol{\sigma} \mathbf{z}_j$;
    Calculate the gradient $\mathbf{g}_\Phi$ of Eqn (25) w.r.t. Φ;
    Update Φ by: $\Phi \leftarrow \Phi - lr \cdot \mathbf{g}_\Phi$.

Algorithm 4 `Inf`^{2}`Guard` against DRAs

Input: A dataset $D = \{\mathbf{x}_n, y_n\}$, hyperparameters $\lambda \in [0,1]$, learning rates lr_1, lr_2, lr_3, #local gradients I, #global rounds T.
Output: Network parameters: Ω, Ψ, Θ.

Initialize $\Theta, \Psi, \Omega, \Phi$ for the encoder f, data reconstruction network g_Ψ, utility preservation network h_Ω, and perturbation distribution parameter.
for round $t = 1$ to T **do**
  for each batch $bs \subset D$ **do**
    Update Φ via Algorithm 3;
    Update g_Ψ (given Θ and $\{\boldsymbol{\delta}_i\}$): Calculate $I_{\Theta,\Psi}^{(JSD)}$ on bs with $\{\boldsymbol{\delta}_i\}$ via Eqn (22); $\Psi \leftarrow \Psi + lr_1 \cdot \partial I_{\Theta,\Psi}^{(JSD)} / \partial \Psi$;
    Update h_Ω (given Θ and $\{\boldsymbol{\delta}_i\}$): Calculate CE loss L_1 on bs with $\{\boldsymbol{\delta}_i\}$ via Eqn (23); Calculate CE loss L_2 on bs with clean data via Eqn (24); $\Omega \leftarrow \Omega - lr_2 \cdot \partial(L_1 + L_2) / \partial \Omega$;
    Update f_Θ (given Ψ, Ω, and $\{\boldsymbol{\delta}_i\}$): $\Theta \leftarrow \Theta - lr_3 \cdot \frac{\partial}{\partial \Theta} (\lambda I_{\Theta,\Psi}^{(JSD)} + (1-\lambda)(L_1 + L_2))$;

Table 9: Training and test sets for primary and MIA tasks.

	CIFAR10	Purchase100	Texas100
Utility training set	25,000	98,662	33,665
Utility test set	25,000	98,662	33,665
Attack training set	40,000	157,859	53,864
Attack test set	10,000	39,465	13,466

Table 10: Training and test sets for primary and PIA tasks.

	Census	RSNA	CelebA
Subset size	[2,32k]	[2,100]	[2,20]
female ratios	$\{0.2, 0.3, \cdots, 0.5\}$	$\{0.2, 0.3, \cdots, 0.8\}$	$\{0.0, 0.1, \cdots, 1.0\}$
Attack train set	8k subsets	14k subsets	22k subsets
Attack test set	2k subsets	3.5k subsets	5.5k subsets
Utility train set	data in 8k subsets	in 14k subsets	in 22k subsets
Utility test set	data in 2k subsets	in 3.5k subsets	in 5.5k subsets

Table 11: Training and test sets for primary and DRA tasks.

	CIFAR10	CIFAR100	Activity
Utility/Attack training set	50,000	50,000	7,352
Utility test set	10,000	10,000	2,947
Attack test set	50	50	50

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., to 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.

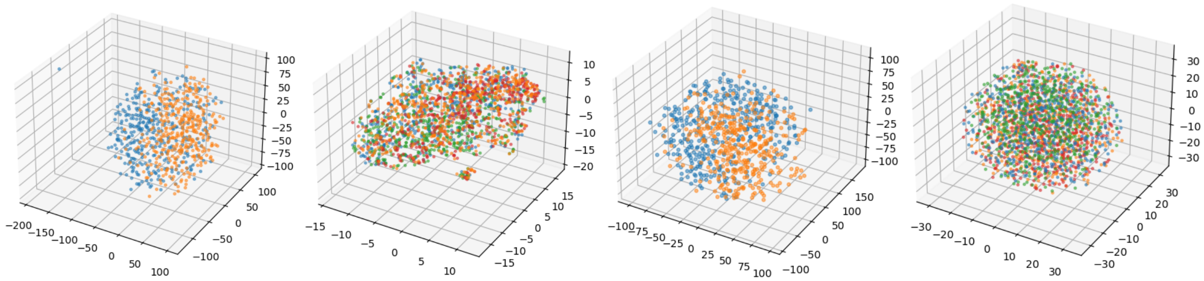
Table 12: More DP results against MIAs.

DP-SGD	CIFAR10		Purchase100		Texas100	
	Utility	MIA Acc	Utility	MIA Acc	Utility	MIA Acc
$\varepsilon = 0.5$	46%	50%	40%	52%	11%	51%
$\varepsilon = 1$	48%	51%	48%	54%	15%	52%
$\varepsilon = 2$	59%	55%	53%	57%	26%	54%
$\varepsilon = 4$	61%	57%	60%	59%	33%	55%
$\varepsilon = 8$	65%	59%	71%	62%	39%	57%
$\varepsilon = 16$	68%	62%	78%	66%	45%	59%
DP-encoder	**CIFAR10**		**Purchase100**		**Texas100**	
	Utility	MIA Acc	Utility	MIA Acc	Utility	MIA Acc
$\sigma^2 = 10$	48%	51%	32%	51%	10%	50%
$\sigma^2 = 1$	59%	56%	54%	56%	26%	54%
$\sigma^2 = 0.1$	71%	66%	65%	59%	46%	55%
$\sigma^2 = 0.01$	78%	79%	78%	64%	49%	60%
`Inf`^{2}`Guard`	**77%**	**51%**	**80%**	**51%**	**46%**	**50%**

Table 13: `Inf`^{2}`Guard` results against PIAs, where the attacker is unknown to the true (mean) aggregator and uses a substitute (max) one to aggregate the subset representations. Knowing the true aggregator is critical for designing better PIAs.

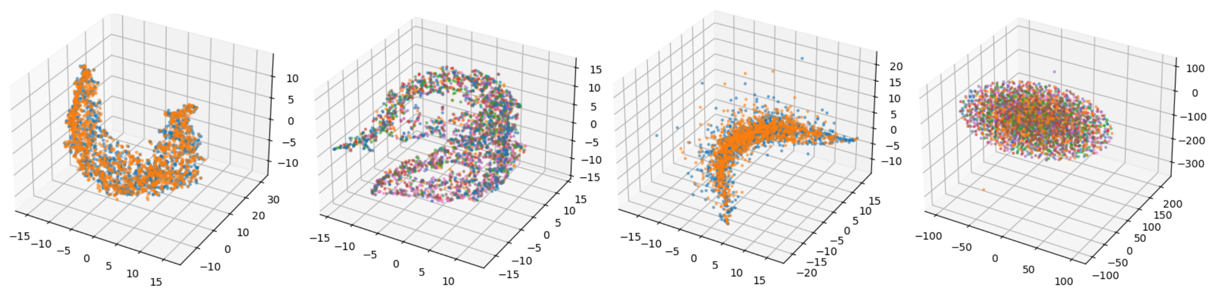
Census			RSNA			CelebA		
λ	Utility	PIA Acc	λ	Utility	PIA Acc	λ	Utility	PIA Acc
0	85%	48%	0	83%	42%	0	91%	40%
0.25	83%	45%	0.25	82%	24%	0.25	91%	22%
0.5	82%	44%	0.5	82%	21%	0.5	91%	15%
0.75	81%	28%	0.75	81%	15%	0.75	88%	9%
1	60%	25%	1	60%	14%	1	63%	9%

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.



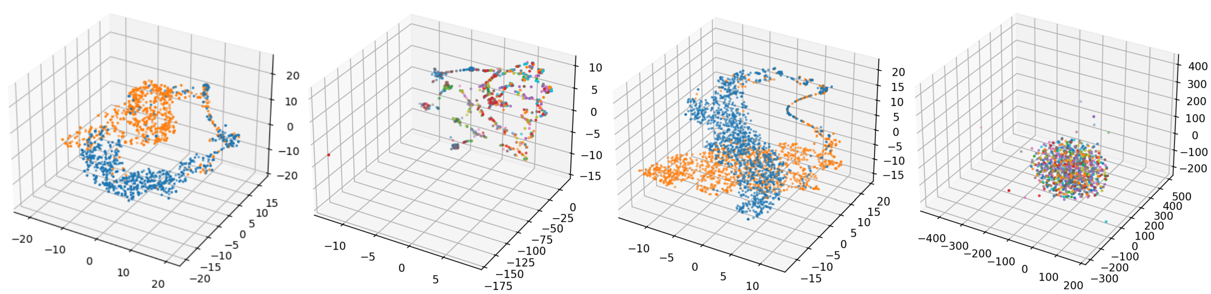
(a) Utility w/o. defense (85%) (b) PIA Acc w/o. defense (68%) (c) Utility w. defense (76%) (d) PIA Acc w. defense (34%)

Figure 12: Inf^{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.



(a) Utility w/o. defense (83%) (b) PIA Acc w/o. defense (52%) (c) Utility w. defense (80%) (d) PIA Acc w. defense (19%)

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



(a) Utility w/o. defense (91%) (b) PIA Acc w/o. defense (50%) (c) Utility w. defense (89%) (d) PIA Acc w. defense (11%)

Figure 14: Inf^{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 ϵ's) against MIAs. We can see Inf^{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 Inf^{2}Guard against PIAs. Table 13 shows Inf^{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 Inf^{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 Inf^{2}Guard.

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

CIFAR10		
Scale ϵ	Utility	SSIM/PSNR
0	89.5%	0.78 / 15.97
1.25	85.7%	0.28 / 13.23
2.25	77.7%	0.24 / 12.34
3.25	64.9%	0.20 / 12.93

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