Noisy Adversarial Representation Learning for Effective and Efficient Image Obfuscation

Jonghu Jeong[*1] **Minyong Cho**[*1] **Philipp Benz**[1] **Tae-hoon Kim**[1]

[1]Deeping Source Inc., Seoul, Republic of Korea

Abstract

Recent real-world applications of deep learning have led to the development of machine learning as a service (MLaaS). However, the scenario of *client-server inference* presents privacy concerns, where the server processes raw data sent from the user's client device. One solution to this issue is to provide an *obfuscator* function to the client device using Adversarial Representation Learning (ARL). Prior works have primarily focused on the privacy-utility trade-off while overlooking the computational cost and memory burden on the client side. In this paper, we propose an effective and efficient ARL method that incorporates feature noise into the ARL pipeline. We evaluated our approach on various datasets, comparing it with state-of-the-art ARL techniques. Our experimental results indicate that our method achieves better accuracy, lower computation and memory overheads, and improved resistance to information leakage and reconstruction attacks.

1 INTRODUCTION

In recent years, machine learning as a service (MLaaS) has gained popularity mainly due to cloud computing and deep learning advances. Often raw data generated on an edge device is sent to the cloud, where machine learning algorithms process it. However, transferring raw data has the drawback of directly leaking privacy-related information to the cloud server, which might violate user privacy. For example, we can consider an edge device transmitting images to the cloud to perform person identification. While a person's picture can be used for identification, the image can further reveal the person's gender, emotional state, race, or location. Ideally, before transmission to the cloud server, privacy-related information should be removed from the images while preserving task utility. Additionally, such a private data representation should be secure against attacks from adversarial actors who attempt to breach a user's privacy by retrieving private attributes from the data representation. It is important to note that the service provider might also be considered a possible adversarial actor. Hence, it is in the clients' interest to remove utility-unrelated information since the representation transmitted from the client is out of their control. We refer to this scenario as *client-server inference*: (1) on the client-side, the privacy-related information is removed from the data. After the data is transmitted, (2) the server-side performs the remaining inference computation without violating the user's privacy.

Many works have focused on the training framework of ARL (Roy and Boddeti, 2019; Bertran et al., 2019; Li et al., 2021; Edwards and Storkey, 2015; Raval et al., 2017; Huang et al., 2017; Wu et al., 2018; Pittaluga et al., 2019; Xiao et al., 2020; Ng et al., 2022; Osia et al., 2018; Mireshghallah et al., 2020, 2021) to mitigate the leakage of sensitive attributes in the context of client-server inference. Commonly, the ARL framework consists of three entities, (1) an obfuscator, which transforms input data into a representation that retains task utility while resolving the correlation of image features to private attributes, and (2) a task model, performing the utility task on the new data representation and (3) a proxy-adversary, attempting to extract sensitive attributes from the representation. In the above scenario of MLaaS, the service providers train an obfuscator, a task model, and a proxy adversary. Then, they deploy the obfuscator to the user's client device. For the sake of the users' privacy, the obfuscator should effectively remove all information unrelated to the utility task while using as least resources of the client as possible and retaining high utility with the obfuscated representation.

Another critical aspect of the obfuscator is client-side computational cost. Client-side resource burden should be as least as possible. To this end, we need a training scheme that

[*]Equal Contribution. Our code is available on the GitHub repository: https://github.com/DeepingSource/noisy-arl

outputs a lightweight obfuscator. Hence, in this work, we introduce a novel ARL approach to improve an obfuscator to be (1) robust against attacks of adversarial actors while retaining task utility (*privacy-utility trade-off*) and (2) limiting the computational resources on the edge device (*efficiency-performance trade-off*). To solve these problems simultaneously, we extend the standard ARL training scheme and propose *noisy adversarial training* and *noisy inference*. The proposed method utilizes off-the-shelf convolutional neural network (CNN) models for mobile-friendly and privacy-preserving machine learning inference. We demonstrate that our method outperforms the state-of-the-art ARL methods in terms of (1) privacy-utility trade-off, (2) efficiency-performance trade-off, (3) is readily applicable to commonly used CNN architectures, and (4) robust to privacy leakage and reconstruction attacks.

2 RELATED WORKS AND BACKGROUND

Data Privacy in Machine Learning Privacy attacks or preservation in machine learning is a vibrant research area with various approaches. The most well-known threats to data privacy are membership inference attacks (Shokri et al., 2017), inversion attacks (Fredrikson et al., 2014), and information leakage attacks (Roy and Boddeti, 2019). The membership inference attack is not applicable in the context of client-server inference since its purpose is to determine whether a single data point is used for model training. In the scenario of the client-server inference, attackers attempt to breach the transmitted representations. With the breached data, they can train their adversary models (a) to reconstruct the original data (*inversion attack*) or (b) to retrieve private information that users might not want to reveal (*information leakage attack*).

Recently, various methods have been proposed for privacy-preserving machine learning. Federated learning (Konečnỳ et al., 2016) and split learning (Gupta and Raskar, 2018; Vepakomma et al., 2018) are methods to train a new machine learning model without the direct input of raw user data. However, these methods focus on protecting privacy during the training phase, not at the inference stage.

In various contexts of deep learning, membership inference attacks have been defended with differential privacy (Dwork, 2008) by adding noise to the representations (Abadi et al., 2016; Arachchige et al., 2019; Fan, 2019a,b; Croft et al., 2021; Chen et al., 2021). However, differential privacy is designed to make two neighboring datasets or data points statistically indistinguishable, not to transform raw data into a new representation that is privacy-safe and usable for the intended tasks (Zhao et al., 2020). Training deep neural networks (DNN) with cryptographic methods has also been explored, such as secure multiparty computation (Cramer et al., 2015) and homomorphic encryption (Nandakumar et al., 2019). However, these are still difficult to deploy in practice due to the computational complexity of the involved operations.

Adversarial Representation Learning In the context of the client-server inference, the most suitable solution is to find a function that transforms data into a new representation that can be utilized for machine learning while being robust to privacy leakage attacks. For this purpose, adversarial representation learning (ARL) tries to find a representation function by optimizing an information-theoretic formulation of privacy and utility (Hsu et al., 2021). In the information-theoretic formulation, we represent the original data, the transformed representation, and the sensitive information from the original data as three random variables X, Z, and Y, respectively. To protect privacy, the mutual information between the sensitive information and the transformed representation, $I(Y;Z)$, should be as minimal as possible. Meanwhile, in terms of preserving utility information, $I(X;Z)$, the mutual information between the original and transformed data should be as maximal as possible. Thus, the objective of ARL is to find a probability distribution $P_{Z|X}$ that minimizes $I(Y;Z)$ while retaining the utility of the representation to a certain degree:

$$\arg\min_{P_{Z|X}} I(Y;Z) \quad s.t.\ I(X;Z) \geq u \tag{1}$$

where u is the desired utility level. ARL approaches commonly set $P_{Z|X}$ as a deterministic function $O : X \rightarrow Z$, where O stands for an *obfuscator*. As the objective suggests, the trade-off between privacy and utility is inevitable since ARL is an optimization problem between two conflicting objectives. Zhao et al. (2020) and Wu et al. (2018) showed that it is possible to define the lower bound of the trade-off formally and confirmed it with experiments on various tasks.

Various prior works have tried to find O in the image domain with deep neural networks (Singh et al., 2021; Roy and Boddeti, 2019; Edwards and Storkey, 2015; Raval et al., 2017; Pittaluga et al., 2019; Xiao et al., 2020; Mireshghallah et al., 2020; Osia et al., 2018). They solved Equation 1 by setting up two proxy models, $T : Z \rightarrow Y_t$ and $A : Z \rightarrow Y_p$, where T stands for the *utility task model* and A stands for the *proxy adversary model*. Y_t stands for the information that needs to be inferred for utility, and Y_p is the one that an adversary could leak. It has been demonstrated that the theoretical bounds for privacy and utility can be empirically reproduced by optimizing three models, O, T, and A, simultaneously (Bertran et al., 2019; Hsu et al., 2020; Xiao et al., 2020; Osia et al., 2018). The optimization is mainly done with stochastic data-driven training of deep neural networks, (a) by cooperatively training O and T to retain utilizable information in Z, (b) while training O and A to adversarially learn to remove private information from Z. After the models are fully trained, O and T are deployed to the client device and the service provider's server, respectively. Finally, O transforms raw data X to the privacy-safe representation Z, and T processes the received Z to infer Y_t for the utility task.

Previous ARL methods differ in various aspects, such as loss function design, model architecture, and training scheme. For example, MaxEnt (Roy and Boddeti, 2019) optimizes the entropy-based loss to make the ARL objective task-agnostic. DISCO (Singh et al., 2021) selectively removes features via pruning filters in the latent space. DeepObfuscator (Li et al., 2021) introduces a training scheme, which incorporates an additional adversary model that reconstructs X from Z. Similar to our approach, Shredder (Mireshghallah et al., 2020) and DPFE (Osia et al., 2018) utilize noisy representations. Their information-theoretic analysis has shown that privacy can be guaranteed by adding noise to encoded representations. Shredder learns a set of noise distributions with regard to a fixed neural net encoder, while our method trains an encoder to adapt to a fixed noise distribution. DPFE uses an auto-encoder during training and noise addition during testing to provide privacy, while our method only uses noise addition for both the training and testing phase. While some methods try to reduce the client-side computation burden with ARL (Mireshghallah et al., 2020; Osia et al., 2018), we propose that privacy can be achieved by choosing a split point of a model and simply training it with a noisy adversarial representation learning scheme.

3 PROBLEM FORMULATION

As mentioned in Section 2, we define three models for our ARL task; an obfuscator $O : X \to Z$, a task model $T : Z \to Y_t$, and an adversary model $A : Z \to Y_p$. The models are represented through CNNs. We consider X to be in RGB image domain, *i.e.* $x \sim X \subset \mathbb{R}^{H\times W\times 3}$ where H and W represent height and width. We define an obfuscator O, which aims to convert each data point x into the obfuscated representation $z \in Z$. We also set the utility task model T and the adversarial attacker A, which are to infer utility attributes y_t and private attributes y_p from the obfuscated representation z, respectively, such that $T(z) = \hat{y}_t \simeq y_t$ and $A(z) = \hat{y}_p \simeq y_p$. From the attacker's perspective, $\hat{y}_p$ should be similar to the private attributes y_p. In terms of the user and the service provider, however, $\hat{y}_p$ should be as dissimilar as possible from y_p while $\hat{y}_t$ should be similar to the utility attributes y_t. Previous works (Bertran et al., 2019; Singh et al., 2021) have shown that the ARL training scheme effectively achieve the mutual information objective (Eq. 1).

As an example of a practical attack scenario, we assume an attacker who is in control of an edge device such as a CCTV camera or an IoT device. The device holds an obfuscator model and transforms the raw data before data transmission. It allows attackers to generate their own datasets to train an adversary model, *e.g.* original input and obfuscated representation pairs. Further, we assume that the attackers are also aware of the original training dataset and the architecture of the service provider's models. Note that this constitutes a strong threat model, which makes it difficult to protect privacy for the service provider. We show that our method protects privacy even under severe conditions.

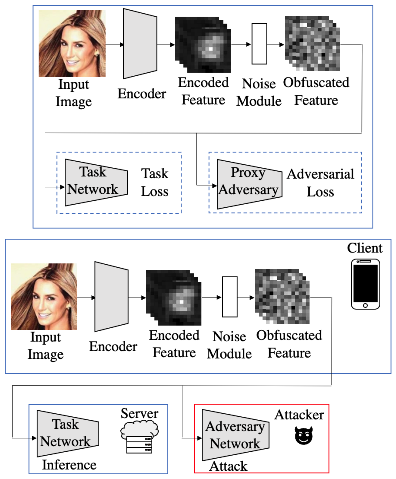

Figure 1: (Top) The training scheme of our method. (Bottom) Inference scenario with possible adversary attack.

We identify two possible attack scenarios: *information leakage attack* and the *reconstruction attack*. For the *information leakage attack*, an attacker can attempt to train a model $A_{leak} : Z \to Y_p$ that directly leaks the representations' private information, *i.e.* $A_{leak}(z) = \hat{y}_p$. In the *reconstruction attack*, the attacker attempts to obtain a model $A_{recon} : Z \to X$ which retrieves the original image from the intermediate representation, from which the private attributes can then be inferred $A_{recon}(z) = \hat{x}$.

4 METHODOLOGY

4.1 NOISY ADVERSARIAL REPRESENTATION LEARNING

As shown in Figure 1, we split an off-the-shelf CNN, such as ResNet (He et al., 2016) $M(x) = (M_2 \circ M_1)(x)$, and use the earlier layers M_1 as a client-side encoder while using the remaining layers M_2 as a server-side task model T. We discuss the influence of the splitting point on the efficiency-performance trade-off in Section 8.1. The encoded feature $M_1(x)$ is then added with noise $\eta \in \mathbb{R}^{H_z\times W_z\times C_z}$ sampled from a Gaussian distribution $\mathcal{N}(0, \sigma^2)$ where H_z, W_z, and C_z represent height, width, and number of channels of z, respectively. The obfuscated feature is $z = O(x) = M_1(x) + \eta$, which will be transmitted from the client-side to the server.

Algorithm 1 Noisy ARL Training Algorithm

Input: Dataset $\mathcal{D} : \mathcal{X} \times \mathcal{Y}_t \times \mathcal{Y}_p$, model initial parameters $\theta_{O=M_1}$, $\theta_{T=M_2}$, θ_A, loss functions $\mathcal{L}_t$, $\mathcal{L}_{leak}$, noise standard deviation σ, loss balance parameter λ, mini-batch size m, number of iterations I

Output: Parameters $\hat{\theta}_O, \hat{\theta}_T$

```
1:  for iteration $= 1, \ldots, I$ do
2:      $B \sim \mathcal{D}$: $|B| = m$          // Sample mini-batch
3:      $\eta \sim \mathcal{N}(0, \sigma^2)$          // Sample noise
4:      $z \leftarrow M_1(x) + \eta$
5:      $g_T \leftarrow \nabla_{\theta_T} \mathbb{E}_{(x,y_t) \sim B}[\mathcal{L}_t(T(z), y_t)]$
                                        // Calculate gradient
6:      $g_A \leftarrow \nabla_{\theta_A} \mathbb{E}_{(x,y_p) \sim B}[\mathcal{L}_{leak}(A(z), y_p)]$
                                        // Calculate gradient
7:      $g_O \leftarrow \nabla_{\theta_O} \mathbb{E}_{(x,y_t,y_p) \sim B}[\mathcal{L}_t(T(z), y_t)$
        $-\lambda * \mathcal{L}_{leak}(A(z), y_p)]$
                                        // Calculate gradient
8:      $\theta_O, \theta_T, \theta_A \leftarrow$
        Optim($\theta_O$,$g_O$), Optim($\theta_T$,$g_T$), Optim($\theta_A$,$g_A$)
                                        // Update parameters
9:  end for
10: return $\theta_O, \theta_T$
```

During training (Figure 1 Top), we train the obfuscator model jointly with the task model and the proxy adversary. We combine our noise module with the standard ARL training scheme. The obfuscator O, the task model T, and the proxy adversary network A are parameterized through their respective weights θ_O, θ_T, and θ_A. Since the objective of the task model is to perform well on the utility task, its loss is defined as $l_t = \mathbb{E}_{x \sim \mathcal{X}}[\mathcal{L}_t(T(z), y_t)]$, where $\mathcal{L}_t$ indicates the task loss function. In case the information leakage attack is chosen as the proxy adversary, the adversary loss can be calculated with $l_{leak} = \mathbb{E}_{x \sim \mathcal{X}}[\mathcal{L}_{leak}(A_{leak}(z), y_p)]$, where $\mathcal{L}_{leak}$ indicates the adversary task loss function. We use simple cross-entropy for $\mathcal{L}_t$ and $\mathcal{L}_{leak}$. Some ARL methods (Singh et al., 2021; Li et al., 2021) used the proxy reconstruction loss to defend against the reconstruction attack, *i.e.*, $l_{recon} = \mathbb{E}_{x \sim \mathcal{X}}[\mathcal{L}_{recon}(A_{recon}(z), x)]$, additional to l_{leak}. In Section 6.2, it is shown that our method is robust to the reconstruction attack. Finally, the obfuscator model should incorporate two objectives, retaining useful task information while discarding privacy-related information. Hence, the obfuscation loss $l_o = l_t - \lambda * l_{leak}$ is minimized by the obfuscator, where λ is a hyper-parameter that balances the two loss terms. To sum up, the final objective is: $\min_{\theta_O, \theta_T} \max_{\theta_A} l_o$.

Algorithm 1 describes the optimization process to solve the min-max objective. For each mini-batch, sampled noise is added to the intermediate feature from O. Then, noise added feature is fed into T and A to calculate the utility task loss and the proxy adversary loss. With the two losses, the gradients g_T and g_A are computed, while the gradient g_O is computed from the weighted sum of the losses. Finally, the models are updated with the gradients via an off-the-shelf optimizer. Note that A is discarded after training since its purpose is to simulate the possible attacks and give supervision to O and T.

During the inference phase (Figure 1 Bottom), the original images are first fed into the obfuscation module on the client side, which includes the encoder O and the noise module. The obfuscated feature is then sent to the server, where the final task predictions are performed with the task network.

4.2 EVALUATION PROTOCOL

To properly evaluate the performance of our approach, we outline our evaluation protocol in the following.

Performance Bounds Theoretically, the utility and privacy adversary accuracy is upper bounded by 100% for a globally optimal model. However, this is not true in real situations. Hence, we provide a "practical" upper bound for utility and privacy through the performance of models trained on the original images for each task, respectively.

Information Leakage Attack It is common to measure the effectiveness of information leakage attacks with the *privacy-utility trade-off* (Singh et al., 2021). The trade-off is measured with the difference (Δ) between the accuracy of the utility task and the leakage attack. In the perspective of privacy protection, the higher, the better for the utility task accuracy, and the lower, the better for the leakage attack accuracy. This naturally leads for Δ to be higher the better. Specifically, given a fully trained fixed obfuscator, we calculate the accuracy with a separately trained utility task model and an adversary model. The separate utility task model is trained to correctly predict target attributes, while the independent privacy leakage attack model is trained to infer private labels. Both models receive obfuscated features as input.

Reconstruction Attack Additionally, we consider the reconstruction (inversion) attack. We perform reconstruction attacks by training CNNs to recover original images from the corresponding obfuscated features. For example, for face images, reconstructed images should (1) not reveal a person's identity and (2) not show private attributes.

Model Efficiency One focus of this work is to consider the performance on the client-side since its computational capacity can often be restricted on the client-side. We evaluate the performance with two metrics on the client device, Giga FLoating point OPerations (GFLOPs) and memory consumption. We count all floating point operations such as additions, multiplications, and divisions on one-time inference to calculate GFLOPs. We also consider all parameters and buffers of models to measure memory usage.

5 EXPERIMENTAL SETUP

Table 1: Comparison of the privacy-utility trade-off (Δ). We compare our method with existing ARL approaches focusing on the privacy-utility trade-off. Regarding Δ, our method outperforms all other methods while showing comparable utility accuracy with the performance bound. Comparison with 'No Noise' shows the effectiveness of our noisy adversarial training and inference.

	Fairface(Race/Gender)			CelebA(Gender/Smiling)			CIFAR10(Multi/Binary)		
Method	Privacy $\downarrow$	Utility $\uparrow$	$\Delta\uparrow$	Privacy $\downarrow$	Utility $\uparrow$	$\Delta\uparrow$	Privacy $\downarrow$	Utility $\uparrow$	$\Delta\uparrow$
RN18	63.57	92.11	-	98.14	93.48	-	94.51	98.79	-
Image Noise	42.61	74.33	31.72	91.71	85.38	-6.33	54.37	87.77	33.40
No Noise (RN18$_3$)	45.22	89.55	44.33	94.54	93.38	-1.16	69.34	97.64	28.30
No Noise (RN18$_4$)	31.56	89.87	58.31	93.19	93.43	0.24	56.02	97.97	41.95
MaxEnt	24.56	90.52	65.96	59.28	93.43	34.15	24.61	97.74	73.13
DISCO	19.00	81.50	62.50	61.20	91.00	29.80	22.30	91.98	69.68
DeepObfs.	50.83	89.64	38.81	97.63	91.92	-5.71	73.79	92.86	19.07
Ours (RN18$_3$)	19.47	89.08	**69.61**	57.77	93.07	**35.30**	21.71	96.92	**75.21**
Ours (RN18$_4$)	15.60	88.34	**72.74**	53.77	90.86	**37.09**	19.81	94.25	**74.44**

5.1 IMPLEMENTATION DETAILS

Models We chose the commonly used CNN, ResNet18 (RN18) (He et al., 2016) as the base architecture to split. RN18 consists of one convolution layer, four residual blocks, and a fully connected layer. We choose the splitting point after each of the four residual blocks. We indicate the different configurations as RN18$_{\{1,2,3,4\}}$, respectively, where the subscript indicates the block after which the network was split.

Note that for the proxy adversary model, we only consider a proxy for information leakage attack, as discussed in Section 4, since we empirically show that our method is robust to reconstruction attacks without considering them during training. For the task and proxy adversary models we use the remaining part of the split architecture, *e.g.* for RN18$_4$, the remaining part would consist of the fully connected layer. This setting is consistent with previous works (Singh et al., 2021; Roy and Boddeti, 2019; Li et al., 2021). The noise parameter is chosen based on the dataset and model's privacy-utility trade-off. A separate Adam (Kingma and Ba, 2014) optimizer is used for all three models with a learning rate of 10^{-3}, and $\lambda = 10^{-2}$ is used to balance the losses.

Settings for Information Leakage Attack First, we compare the experiment setting from the previous ARL method DISCO (Singh et al., 2021). We set "Smiling" as the utility attribute and "Gender" as the privacy attribute for CelebA (Liu et al., 2015), "Gender" as the utility attribute, and "Race" as the privacy attribute for FairFace (Karkkainen and Joo, 2021). For CIFAR10 (Krizhevsky et al., 2009), following the setting from MaxEnt (Roy and Boddeti, 2019), the utility task is defined as classifying living objects (*e.g.* "bird", "cat", *etc.*) or non-living objects (*e.g.* "airplane", "automobile", *etc.*) and privacy task as classifying separate 10 classes. All datasets used the official train and validation split. Furthermore, results on more complex task settings, such as multi-class classification and facial landmark detection, are provided in the supplementary material.

Settings for the Reconstruction Attack The reconstruction attack is performed on the CelebA dataset with the decoder architecture from DeepObfs (Li et al., 2021), which is trained with the Adam optimizer with a learning rate of 10^{-3} and MSE between the original and the reconstructed image. We depict the qualitative results, and additionally provide quantitative visual dissimilarity comparison between the original and reconstructed images. Various visual metrics are reported, such as MSE, L_1, SSIM (Wang et al., 2004), MS-SSIM (Wang et al., 2003), PSNR (Horé and Ziou, 2010), and LPIPS (Zhang et al., 2018), which are commonly considered as proxies for human vision. Additionally, a qualitative user study is provided in Section 7.

5.2 BASELINES & COMPARED METHODS

ResNet18 We report the utility and privacy performance for ResNet18 (RN18) models trained on the respective task with original images to indicate the practical performance bounds.

Image Noise Directly adding sufficient noise to the input image is a simple way to obfuscate without any trainable parameters. We add Gaussian noise sampled from $\mathcal{N}(0, \sigma^2)$ to the input image directly while obeying the image range of pixels in the range (0,1), where $\sigma = 2$ is used for CelebA and FairFace and $\sigma = 0.8$ for CIFAR10. The σ is chosen based on the noise that fully obfuscates the image for a human observer. We used the entire ResNet18 model for both the utility and privacy models.

No Noise As an ablation experiment on our method, we conduct basic ARL training without a noise module. We report the performance of RN18$_3$ and RN18$_4$.

MaxEnt We compare the ARL method MaxEnt, which uses full ResNet18 as a client-side obfuscator. The obfuscator's final output is a vector with length d. $d = 128$ is used for CIFAR10, which is the original setting from MaxEnt, and $d = 256$ is used for FairFace and CelebA by considering the size of the input images.

Table 2: The efficiency of each client model. An image with a size of ($178 \times 178 \times 3$) is used to measure the performance. Our method (Bottom) shows the lowest computational costs compared to all the baselines (Top).

Benchmark	DeepObfs.	DISCO	MaxEnt	RN18
Comp. Cost (GFLOPs) $\downarrow$	6.00	2.52	2.52	2.52
Memory (MB) $\downarrow$	1.00	42.80	43.17	42.69
Benchmark	RN18$_1$	RN18$_2$	RN18$_3$	RN18$_4$
Comp. Cost (GFLOPs) $\downarrow$	0.75	1.31	1.92	2.52
Memory (MB) $\downarrow$	0.60	2.61	10.63	42.67

DISCO We report the privacy-utility trade-off numbers as in the original work. We reconfirm the reconstruction vulnerability of DISCO as reported in their work with their parameters.

DeepObfuscator Since the authors did not open-source their code, we re-implemented DeepObfuscator based on the provided information in the paper.

6 EXPERIMENTAL RESULTS

6.1 PRIVACY-UTILITY TRADE-OFF AND EFFICIENCY

Table 1 compares baselines and state-of-the-art methods with our proposed approach regarding the privacy-utility trade-off (Δ). First, we observe that 'Image Noise' decreases the performance for both privacy and utility. This is because that the method obfuscates the images without taking utility and privacy tasks into account. For RN18$_{\{3,4\}}$ without noise, which can be considered an ablation of our method, the utility task accuracy is nearly retained compared to the performance upper bound. However, the adversary achieved high accuracy for the leakage attack compared to other methods. The results show that it is hard for the obfuscator to learn to remove private information even with the adversary proxy loss. Another notable point is that training the obfuscator to remove the private information becomes more challenging with the lower layers, as it can be confirmed with the privacy accuracy gap between RN18$_3$ and RN18$_4$ (*e.g.* For FairFace 45.22% and 31.56%, respectively). This phenomenon is discussed in Section 8.1.

Among the state-of-the-art methods, MaxEnt and DISCO successfully achieve high Δ, with MaxEnt constituting the most robust technique. DeepObfs. shows limited privacy protection under our strong evaluation protocol.

Our methods (RN18$_{\{3,4\}}$) show the best Δ among all the information leakage attack settings. The privacy accuracy is comparably lower than other methods, such that the biggest Δ could be achieved even with the utility accuracies that are not always the highest. These are notable results since our models are efficient and lighter than compared methods, as indicated in Table 2. Our RN18$_4$ is comparable to MaxEnt and DISCO in terms of memory and computational cost. Hence, under similar efficiency, we observe that our noisy adversarial training and inference have a noticeable effect on the privacy-utility trade-off, outperforming all baselines and previous approaches with significant margins. For example, our approach achieves a privacy-utility trade-off of 72.74% for Fairface, a 7%p increase compared to MaxEnt.

Table 3: Quantitative results of the reconstruction attack on CelebA. Visual dissimilarity scores between original and reconstructed images. The result shows that our method outperforms the other methods with all the metrics.

Method	MSE $\uparrow$	L1 $\uparrow$	SSIM $\downarrow$	MS-SSIM $\downarrow$	PSNR $\downarrow$	LPIPS $\uparrow$
Image Noise	584.88	16.97	0.6017	0.7776	20.46	0.3710
No Noise (RN18$_3$)	1391.39	26.89	0.4666	0.6155	16.70	0.4882
No Noise (RN18$_4$)	1841.70	31.70	0.4558	0.5829	15.48	0.4857
MaxEnt	4955.44	58.83	0.3893	0.4057	11.19	0.6619
DeepObfs.	182.63	9.47	0.7834	0.9298	25.52	0.1864
DISCO	567.17	15.94	0.5765	0.7611	20.60	0.4351
Ours (RN18$_3$)	5437.02	63.22	**0.3086**	0.1682	10.78	0.8045
Ours (RN18$_4$)	**5454.12**	**63.48**	0.3301	**0.1571**	**10.77**	**0.8197**

Additionally, our proposed approach with an RN18 split after the third block (RN18$_3$) also achieves a higher privacy-utility trade-off than all methods. This is especially significant since RN18$_3$ achieves a noticeably smaller client-side burden, with an approximate memory reduction by a factor of 4 and a computational cost of only 76% (1.92 GFLOPS compared to 2.52 GFLOPs) of MaxEnt and DISCO.

While DeepObfs. exhibits a comparably small memory footprint, its low Δ and high computational cost indicate its inferiority to other approaches.

In summary, our noisy adversarial training and inference showed increased Δ with decreased client-side resources cost, outperforming various methods on other benchmarks.

6.2 RECONSTRUCTION ATTACK

Figure 2 shows the visual evaluation of the compared methods for the reconstruction attack. First, the reconstruction results of DeepObfs., DISCO, and 'Image Noise' show a slightly different identity from the original. However, they are still distinguishable regarding the private attribute, in this case, 'Gender'. The 'No Noise' method appears to have removed the identity and the background context, but it also shows the distinguishable 'Gender' attribute. Our method and MaxEnt are the only methods that successfully defended the attack concerning the identity and private attribute. It is noticeable that the reconstructed images of MaxEnt overall show the same identity and the 'Smile' task attribute, while nearly no facial features are distinguishable for our method.

Table 3 reaffirms our results quantitatively. DeepObfs., DISCO, and 'Image Noise' have the lowest visual dissimilarity, since they showed similar identity and private attribute

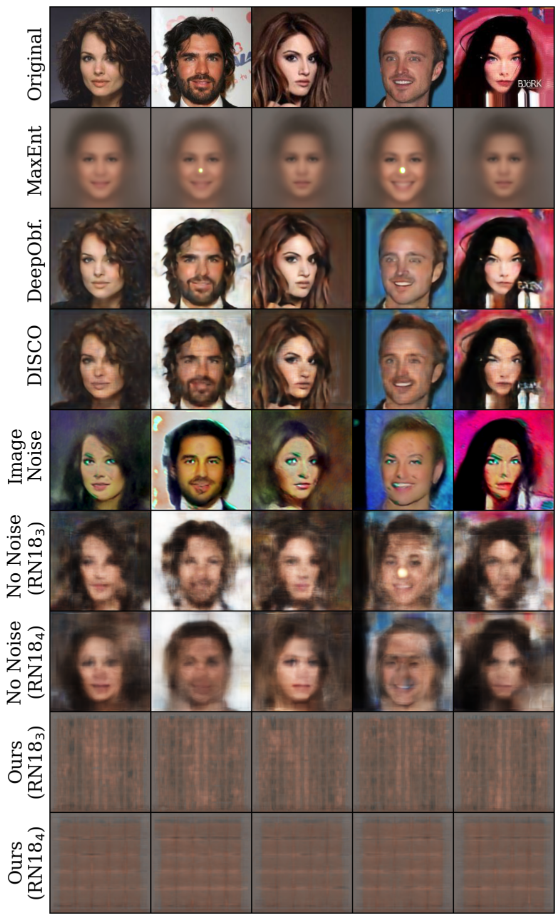

Figure 2: Reconstruction attack on CelebA. Except for our method and MaxEnt, all other methods failed to defend the reconstruction attack. While a few methods (*e.g.* 'Image Noise', No Noise (RN18$_4$)) have successfully defended revealing the exact identity of the person, they failed to remove the private attribute ('Gender').

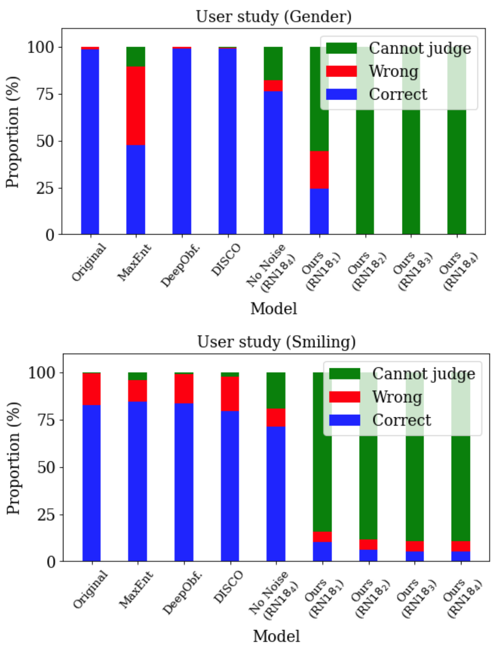

Figure 3: Results for user study on reconstructed images.

on Figure 2. 'No Noise' and MaxEnt have shown high dissimilarity, but still, our method shows the best score since no distinguishable objects are present.

In summary, our method showed the best robustness to the reconstruction attack in terms of both qualitative and quantitative results. Note that our method is robust against the reconstruction attack even without incorporating it into our optimization process. With our noisy adversarial training, the obfuscator successfully learns an obfuscated representation robust against the reconstruction attack, while retaining task utility and removing private information.

7 USER STUDY

In addition to the quantitative results of the reconstruction attack, we present a user study to show that our method's robustness against reconstruction is aligned with human vision. We obfuscate the images using each technique shown in Figure 3. Finally, we attack the obfuscated images using settings from Section 5.1. We randomly selected 30 people and set 270 images from the reconstruction attackers for the survey. We asked them to distinguish whether the person in the image is either smiling for the "Smiling" utility task, or whether the person is male or female for the "Gender" privacy task. We also provide the option to select "cannot judge" for instances in which the reconstructed image is too obfuscated for the person's gender or facial expression to be discernible.

As shown in Figure 3, we present that users correctly identified the "Gender" attribute on the reconstructed images when the obfuscated representations are from DeepObfs., DISCO, and No Noise (RN18$_4$). Only ours and MaxEnt show protections against the reconstruction attack for the "Gender" classification task. For the smiling attribute, we note that the wrong proportion of answers on original images is about 20%. This result is because judging whether a person is smiling or not is subjective. The results for the "Smiling" classification present that all methods other than ours failed to defend against the attacks. We highlight these results since concealing the "Smiling" attribute for utility tasks is out of our interest. We note that RN18$_1$ performs worse compared to (RN18$_{\{3,4\}}$) as expected. However, it outperforms the other compared methods by a considerable margin. We conclude that our approach successfully hides sensitive information from humans even under the reconstruction attack and outperforms previous methods. We report detailed settings for impartial results in the supplementary materials.

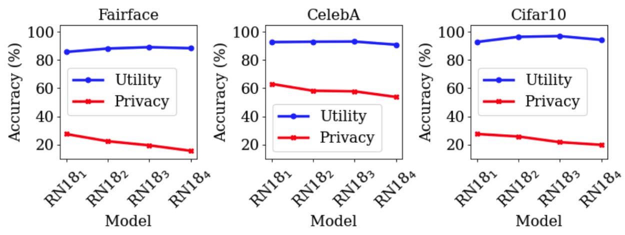

Figure 4: Performance-efficiency trade-off. The result shows better performance when choosing the split point from the higher layers. Nevertheless, our approach shows comparable performance even with the split at the lower layers.

Table 4: Model Ablation. Trained on FairFace dataset.

Method	GFLOPs	Memory (MB)	Privacy ↓	Utility ↑	Δ ↑
MobileNetV2 (MNV2) Orig.	0.4457	8.62	54.40	91.07	36.67
MNV2$_{16}$ + Adv loss	0.3585	3.97	38.23	90.48	52.25
MNV2$_{16}$ + Noise + Adv loss			22.22	90.07	**67.85**
AlexNet Orig.	0.8952	217.47	61.60	88.47	26.87
AlexNet$_4$ + Adv loss	0.6679	7.17	51.09	88.42	37.33
AlexNet$_4$ + Noise + Adv loss			31.59	86.79	**55.20**
VGG11 Orig.	9.5548	491.26	65.86	90.66	24.80
VGG11$_5$ + Adv loss	5.8864	8.19	63.40	89.62	26.22
VGG11$_5$ + Noise + Adv loss			42.11	87.80	**45.69**

8 ABLATION EXPERIMENTS

8.1 EFFICIENCY-PERFORMANCE TRADE-OFF FOR DIFFERENT SPLIT POINTS

Choosing the split point from the lower layers of the model results in compromised privacy, whereas choosing from the higher layers is advantageous for preserving privacy (Yosinski et al., 2014). However, choosing a split from the higher layer increases the client-side burden due to higher memory usage and computational cost. Thus, selecting an appropriate splitting point is essential by considering the trade-off between model efficiency and performance.

We report the privacy and utility accuracy for each variant of ResNet18 model in Figure 4. The result confirms that a better performance (Δ) can be achieved with a split point at higher layers. The privacy accuracy decreases as the split is at higher layers since the features are being processed to be more specific to the utility task. The utility accuracy remains similar, independent of the split location, which increases Δ for the higher layers. Note that our approach has comparable performance even with the lower layer split model. For example, RN18$_2$ achieves 88.10% for utility accuracy and 22.38% for privacy, which leads to the privacy-utility gap of 65.72%. This is 3.22%p better than DISCO and only 0.24%p worse than MaxEnt while having only half the computational cost and $1/16$ memory burden.

8.2 OTHER NETWORK ARCHITECTURES

We further investigate whether our method applies to other network architectures. We use three commonly used CNN models, MobileNetV2 (Sandler et al., 2018) split at the 16th convolution layer out of 19, AlexNet (Krizhevsky et al., 2012) split at the fifth convolution layer out of 8, VGG11 (Simonyan and Zisserman, 2014) split at the fourth convolution layer out of 5 with $\sigma = 30, 15, 60$, respectively. For each model, we followed the evaluation protocol for the FairFace dataset. The models are trained with proxy adversarial loss (Adv loss) and our proposed method (ARL with noise). Table 4 confirms that our method also works with various generally adopted architectures. For all three models, the utility is reasonably retained while effectively protecting privacy. This result shows that the training scheme with our noise module can be readily applied to off-the-shelf model architectures. The computational cost and memory usage is also compared between the original and split model. It is noticeable that the MobileNetV2 architecture could even further reduce the computational burden on the client side with our method.

9 DISCUSSION

In terms of adding noise to the representation, differential privacy based methods (Wang et al., 2018) for privacy protection in deep learning might seem similar to ours. However, differential privacy is not designed to be robust against the information leakage attack. Our method can consider possible information leakage attacks in advance by following the information-theoretic ARL formulation. Further, efficiency improvements can be achieved via pruning (Han et al., 2015), quantization (Jacob et al., 2018), and knowledge distillation (Hinton et al., 2015) which are orthogonal to our proposed method.

10 CONCLUSION

We proposed a novel ARL method that incorporates feature noise during training and inference. Compared to SOTA ARL methods, our approach achieves better accuracy, lower computation and memory overheads, and stronger resistance to information leakage and reconstruction attacks. In particular, we conducted a user study to validate the qualitative superiority of our method against reconstruction attacks. Moreover, with thorough ablation experiments, we demonstrated the insight for choosing model split points and the general applicability of our method to off-the-shelf CNNs. Overall, our findings highlighted the potential of feature noise in ARL as a promising direction for future research.

References

Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *ACM SIGSAC Conference on Computer and Communications security*, 2016.

Pathum Chamikara Mahawaga Arachchige, Peter Bertok, Ibrahim Khalil, Dongxi Liu, Seyit Camtepe, and Mohammed Atiquzzaman. Local differential privacy for deep learning. *IEEE Internet of Things Journal*, 2019.

Martin Bertran, Natalia Martinez, Afroditi Papadaki, Qiang Qiu, Miguel Rodrigues, Galen Reeves, and Guillermo Sapiro. Adversarially learned representations for information obfuscation and inference. In *International Conference on Machine Learning (ICML)*, 2019.

Jia-Wei Chen, Li-Ju Chen, Chia-Mu Yu, and Chun-Shien Lu. Perceptual indistinguishability-net (pi-net): Facial image obfuscation with manipulable semantics. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

Ronald Cramer, Ivan Bjerre Damgård, et al. *Secure multiparty computation*. Cambridge University Press, 2015.

William L Croft, Jörg-Rüdiger Sack, and Wei Shi. Obfuscation of images via differential privacy: From facial images to general images. *Peer-to-Peer Networking and Applications*, 2021.

Cynthia Dwork. Differential privacy: A survey of results. In *International conference on theory and applications of models of computation*, 2008.

Harrison Edwards and Amos Storkey. Censoring representations with an adversary. *arXiv preprint arXiv:1511.05897*, 2015.

Liyue Fan. Differential privacy for image publication. In *Theory and Practice of Differential Privacy (TPDP) Workshop*, 2019a.

Liyue Fan. Practical image obfuscation with provable privacy. In *2019 IEEE international conference on multimedia and expo (ICME)*. IEEE, 2019b.

Matthew Fredrikson, Eric Lantz, Somesh Jha, Simon Lin, David Page, and Thomas Ristenpart. Privacy in pharmacogenetics: An {End-to-End} case study of personalized warfarin dosing. In *USENIX Security Symposium*, 2014.

Otkrist Gupta and Ramesh Raskar. Distributed learning of deep neural network over multiple agents. *Journal of Network and Computer Applications*, 2018.

Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*, 2015.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Conference on computer vision and pattern recognition (CVPR)*, 2016.

Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Alain Horé and Djemel Ziou. Image quality metrics: Psnr vs. ssim. In *International Conference on Pattern Recognition*, 2010.

Hsiang Hsu, Shahab Asoodeh, and Flavio Calmon. Obfuscation via information density estimation. In *International Conference on Artificial Intelligence and Statistics*, 2020.

Hsiang Hsu, Natalia Martinez, Martin Bertran, Guillermo Sapiro, and Flavio P Calmon. A survey on statistical, information, and estimation—theoretic views on privacy. *IEEE BITS the Information Theory Magazine*, 2021.

Chong Huang, Peter Kairouz, Xiao Chen, Lalitha Sankar, and Ram Rajagopal. Context-aware generative adversarial privacy. *Entropy*, 2017.

Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

Kimmo Karkkainen and Jungseock Joo. Fairface: Face attribute dataset for balanced race, gender, and age for bias measurement and mitigation. In *Winter Conference on Applications of Computer Vision (WACV)*, 2021.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Jakub Konečnỳ, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems (NeurIPS)*, 2012.

Ang Li, Jiayi Guo, Huanrui Yang, Flora D Salim, and Yiran Chen. Deepobfuscator: Obfuscating intermediate representations with privacy-preserving adversarial learning on smartphones. In *International Conference on Internet-of-Things Design and Implementation*, 2021.

Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *International Conference on Computer Vision (ICCV)*, 2015.

Fatemehsadat Mireshghallah, Mohammadkazem Taram, Prakash Ramrakhyani, Ali Jalali, Dean Tullsen, and Hadi Esmaeilzadeh. Shredder: Learning noise distributions to protect inference privacy. In *International Conference on Architectural Support for Programming Languages and Operating Systems*, 2020.

Fatemehsadat Mireshghallah, Mohammadkazem Taram, Ali Jalali, Ahmed Taha Taha Elthakeb, Dean Tullsen, and Hadi Esmaeilzadeh. Not all features are equal: Discovering essential features for preserving prediction privacy. In *Proceedings of the Web Conference 2021*, 2021.

Karthik Nandakumar, Nalini Ratha, Sharath Pankanti, and Shai Halevi. Towards deep neural network training on encrypted data. In *Conference on Computer Vision and Pattern Recognition Workshops (CVPR-W)*, 2019.

Tony Ng, Hyo Jin Kim, Vincent T Lee, Daniel DeTone, Tsun-Yi Yang, Tianwei Shen, Eddy Ilg, Vassileios Balntas, Krystian Mikolajczyk, and Chris Sweeney. Ninjadesc: Content-concealing visual descriptors via adversarial learning. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

Seyed Ali Osia, Ali Taheri, Ali Shahin Shamsabadi, Kleomenis Katevas, Hamed Haddadi, and Hamid R Rabiee. Deep private-feature extraction. *Transactions on Knowledge and Data Engineering*, 2018.

Francesco Pittaluga, Sanjeev Koppal, and Ayan Chakrabarti. Learning privacy preserving encodings through adversarial training. In *Winter Conference on Applications of Computer Vision (WACV)*, 2019.

Nisarg Raval, Ashwin Machanavajjhala, and Landon P Cox. Protecting visual secrets using adversarial nets. In *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. IEEE, 2017.

Proteek Chandan Roy and Vishnu Naresh Boddeti. Mitigating information leakage in image representations: A maximum entropy approach. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *Symposium on security and privacy (SP)*, 2017.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Abhishek Singh, Ayush Chopra, Ethan Garza, Emily Zhang, Praneeth Vepakomma, Vivek Sharma, and Ramesh Raskar. Disco: Dynamic and invariant sensitive channel obfuscation for deep neural networks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

Praneeth Vepakomma, Otkrist Gupta, Tristan Swedish, and Ramesh Raskar. Split learning for health: Distributed deep learning without sharing raw patient data. *arXiv preprint arXiv:1812.00564*, 2018.

Ji Wang, Jianguo Zhang, Weidong Bao, Xiaomin Zhu, Bokai Cao, and Philip S Yu. Not just privacy: Improving performance of private deep learning in mobile cloud. In *ACM SIGKDD international conference on knowledge discovery & data mining*, 2018.

Zhou Wang, Eero P Simoncelli, and Alan C Bovik. Multi-scale structural similarity for image quality assessment. In *Asilomar Conference on Signals, Systems & Computers*, 2003.

Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *Transactions on Image Processing*, 2004.

Zhenyu Wu, Zhangyang Wang, Zhaowen Wang, and Hailin Jin. Towards privacy-preserving visual recognition via adversarial training: A pilot study. In *European Conference on Computer Vision (ECCV)*, 2018.

Taihong Xiao, Yi-Hsuan Tsai, Kihyuk Sohn, Manmohan Chandraker, and Ming-Hsuan Yang. Adversarial learning of privacy-preserving and task-oriented representations. In *AAAI Conference on Artificial Intelligence*, 2020.

Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? *Advances in neural information processing systems (NeurIPS)*, 2014.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

Han Zhao, Jianfeng Chi, Yuan Tian, and Geoffrey J Gordon. Trade-offs and guarantees of adversarial representation learning for information obfuscation. *Advances in neural information processing systems (NeurIPS)*, 2020.