Protecting Sensitive Attributes by Adversarial Training Through Class-Overlapping Techniques

Tsung-Hsien Lin, Ying-Shuo Lee, Fu-Chieh Chang, J. Morris Chang, *Senior Member, IEEE*, and Pei-Yuan Wu

***Abstract*— In recent years, machine learning as a service (MLaaS) has brought considerable convenience to our daily lives. However, these services raise the issue of leaking users' sensitive attributes, such as race, when provided through the cloud. The present work overcomes this issue by proposing an innovative privacy-preserving approach called privacy-preserving class overlap (PPCO), which incorporates both a Wasserstein generative adversarial network and the idea of class overlapping to obfuscate data for better resilience against the leakage of attribute-inference attacks(i.e., malicious inference on users' sensitive attributes). Experiments show that the proposed method can be employed to enhance current state-of-the-art works and achieve superior privacy–utility trade-off. Furthermore, the proposed method is shown to be less susceptible to the influence of imbalanced classes in training data. Finally, we provide a theoretical analysis of the performance of our proposed method to give a flavour of the gap between theoretical and empirical performances.**

***Index Terms*— Privacy-preserving machine learning, adversarial training, generative adversarial network, class overlap, machine learning as a service, Wasserstein distance, data obfuscation.**

I. Introduction

CLOUD computing provides us great convenience, but it also threatens our privacy and security. As an example, Google allows us to compose or receive emails on any device simply by logging into our account; however, the content of our emails may be viewed and used to recommend personalized ads [1]. Moreover, Facebook allows us to express our preferences in the community and share content with friends; nevertheless, the content may be used and analyzed by third parties and may even be used to manipulate elections, such as in the Cambridge Analytica scandal [2]. Another recent case from Facebook involved the violation of Illinois biometric privacy law by harvesting facial data for Tag Suggestions from the photos of millions of users in the state without their permission and without informing them about the duration for which the data would be kept [3].

With the emergence of privacy issues, emphasis on the protection of cloud data has become the trend in recent years. The European Union's (EU's) General Data Protection Regulation [4] has significantly impacted suppliers who need to obtain data from EU citizens and has formulated strict regulations and penalties to protect users. In the current era, privacy is always an issue that needs to be considered concurrently, as we enjoy the convenience of cloud computing.

Machine learning as a service (MLaaS) [5] is a type of cloud computing service. This analytical platform tends to bundle cloud-based machine learning and computing resources together (e.g., Microsoft Azure Machine Learning Studio, AWS Machine Learning, and Google Cloud Machine Learning Engine). Al-Rubaie [6] conducted a complete survey of the privacy issues in MLaaS. A MLaaS framework is shown in Fig 1. When users use machine learning as a service, uploading data to the cloud may reveal private information. The upper part of Fig 1 shows that without protection, the attacker can directly obtain the raw data to identify the privacy information. Consequently, a local privatizer is needed to obfuscate the data. In the past, researchers have designed several types of privatizers such as applying homomorphic encryption [7], removing the private area [8], or uploading only the information required by the model [9]. These methods not only obfuscated data (i.e., data that looks real while its confidential information is removed) as features to prevent privacy leakage but also retained the information required by the model. Nevertheless, [10] shows that even if the data are already obfuscated, an attacker may still steal privacy information from these obfuscated data, as illustrated in the lower part of Fig 1.

In recent years, the development of deep learning has enabled the analysis of obfuscated data to reveal the private information of the raw data. Common privacy leaks of obfuscated data include *reconstruction attacks* [11], [12], [13], [14] and *attribute-inference attacks* [10], [13], [14], [15], [16], [17]. In the reconstruction attack, the attacker will recover the original user data from the obfuscated data, and in the attribute-inference attack, the obfuscated data will leak a user's sensitive attributes, such as race. To mitigate this problem, Tripathy et al. [18] proposed a generative adversarial network (GAN) model to simulate an attacker through the inherent training procedure of GAN. The privatizer thus created could be trained to obfuscate the data containing as

Manuscript received 9 May 2022; revised 12 September 2022 and 18 November 2022; accepted 21 December 2022. Date of publication 11 January 2023; date of current version 24 January 2023. This work was supported in part by the Asian Office of Aerospace Research and Development under Grant FA2386-20-1-4039; in part by the Ministry of Science and Technology, Taiwan, under Grant 109-2634-F-002-038, Grant 110-2222-E-002-008, and Grant 110-2634-F-002-039; and in part by the National Taiwan University under Grant 106R891310 and Grant 111L880601. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Hossein Pishro-Nik. *(Corresponding author: Pei-Yuan Wu.)*

Tsung-Hsien Lin and Ying-Shuo Lee are with the Graduate Institute of Communication Engineering, National Taiwan University, Taipei 10617, Taiwan.

Fu-Chieh Chang is with the MediaTek Research, Taipei 10617, Taiwan, and also with the Graduate Institute of Communication Engineering, National Taiwan University, Taipei 10617, Taiwan.

J. Morris Chang is with the Department of Electrical Engineering, University of South Florida, Tampa, FL 33620 USA.

Pei-Yuan Wu is with the Department of Electrical Engineering, National Taiwan University, Taipei 10617, Taiwan (e-mail: peiyuanwu@ntu.edu.tw).

Digital Object Identifier 10.1109/TIFS.2023.3236180

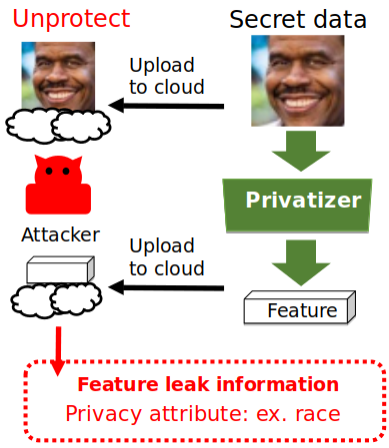

Fig. 1. Potential privacy leakage in MLaaS.

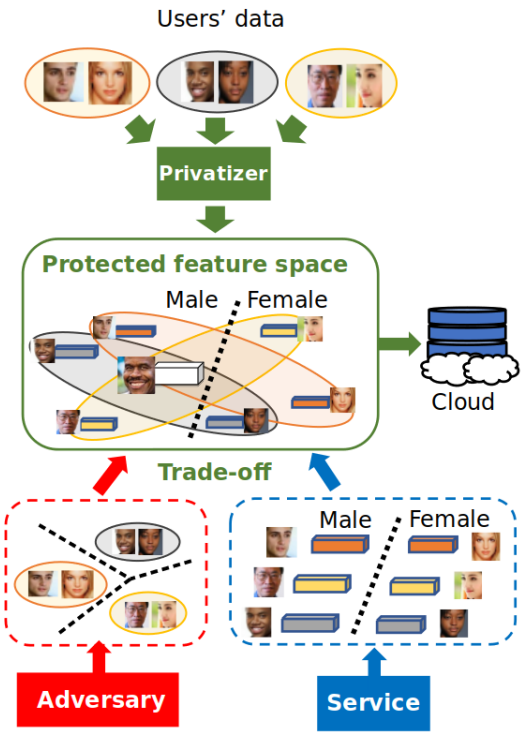

Fig. 2. Network architecture for generative adversarial privacy.

little privacy information as possible, yielding good results in their experiments.

In this paper, we further improve the GAN-based privatizer by introducing the idea of *class overlap* for privacy preservation from the field of domain adaptation. Class overlap is a phenomenon in feature space. When a class overlap occurs, successfully classifying the overlapping area becomes difficult. In the field of domain adaptation, researchers have devoted their efforts to making the class overlap bring different domain data into a common feature space. This matches our purpose of privacy preservation, where we also intend to make different privacy classes overlap with each other to defend against attribute-inference attacks.

We propose the method of **privacy-preserving class overlap (PPCO)**, which applies the distribution-matching technique to train the privatizer. We choose the Wasserstein distance [19] to measure the similarity between data distributions because it shows better convergence properties [19] than KL divergence minimized by cross-entropy loss under adversarial training. The model is trained to minimize the distance between different sensitive attributes, thus hindering the attacker from determining the correct sensitive attribute. Simultaneously, the model is trained to maintain the machine learning (ML) service accuracy to ensure that the obfuscated data contain the essential information for the ML service. Furthermore, we discuss the impact of class-imbalanced data on the training of the privatizer over image datasets.

The main contributions of the present study are as follows:

- We propose PPCO as a privacy-preserving ML method that incorporates both Wasserstein GAN (WGAN) and the idea of class overlapping.
- We demonstrate that the proposed PPCO can be applied to improve current state-of-the-art works [13], [14], achieving better utility–privacy trade-off.
- We demonstrate that PPCO can robustly protect privacy even when trained on extremely imbalanced datasets.
- We derive the theoretical performance of PPCO and show that our empirical performance is consistent with theoretical performance.

The outline of this paper is as follows. We first introduce the related works in Section II and define the threat model of concern in Section III. Further, we elaborate on the proposed method in Section IV and then describe experimental and theoretical analyses to validate the proposed method in Sections V–VII. Finally, Section VIII concludes this paper.

II. Related Work

In this section, we introduce the privacy leakage issue and existing solutions, namely, generative adversarial privacy and cryptography-based approaches.

A. Privacy Leakage During Feature Extraction

A naive approach toward privacy protection is to adopt feature-extraction methods that were not originally designed for privacy protection purposes. However, reportedly one can quite accurately reconstruct the original image from various well-known features extracted from, for example, the scale-invariant feature transform (SIFT) [20], histogram of oriented gradient (HOG) [21], or the middle layers of neural networks [22]. In addition to 2D images, Pittaluga et al. [11] also demonstrated how 3D images could be reconstructed from 3D point clouds.

Features may also contain the privacy attribute information from raw data. Song and Shmatikov [10] indicated that after training the neural network according to the task objective function, the intermediate layers of the neural network may convey information that is unrelated with the objective function, including the sensitive attributes of the raw data.

B. Generative Adversarial Privacy

To solve the problem of obfuscated data containing private information, the current state-of-the-art protection method trains privatizers that conceal private information through an adversarial training setting. The original idea of adversarial training [23] (i.e., GAN) is to simultaneously train two

neural networks, namely, the generator and discriminator. The generator generates images to try and fool the discriminator, while the discriminator aims to recognize the real images from the generated images. Thus, the goal of adversarial training is to train the generator so that it generates very realistic images.

Inspired by adversarial training, many works [12], [13], [14], [15], [24], [25], [26] on generative adversarial privacy have been proposed. They have mainly utilized three neural networks, namely, the privatizer, adversaries, and service, as shown in Fig. 2. The privatizer and adversaries correspond to the generator and discriminator under the adversarial training framework, respectively. The privatizer aims to extract obfuscated data from the raw data that contain the information needed for the service while fooling the adversaries. The adversaries' goal is not to be fooled by the privatizer and extract obfuscated data to achieve privacy leaks, such as attribute-inference and reconstruction attacks. The service is a model placed on the cloud that provides services based on the obfuscated data extracted by the privatizer. For example, in Fig. 2, the users' data includes two attributes: gender and race. Gender is the attribute to be classified, and race is the sensitive attribute. The adversary aims to attack the sensitive attribute classifying the race from the users' data.

In attribute-inference attack [15], [18] the adversary aims to classify the sensitive attribute. Tripathy et al. [18] proposed a privacy-preserving adversarial network, where the privatizer aims to extract the feature containing minimal mutual information of the sensitive attributes, under the constraint that the distortion between the extracted obfuscated data and the useful data attributes (for the service) is within some tolerable budget. Wu et al. [15] proposed two strategies—budget model restarting and ensemble—to enhance the generalization of the learned degradation in protecting privacy against unseen hacker models.

In reconstruction attack [12], [27] the adversary aims to recover the original users' data based on the obfuscated data. Tseng and Wu [27] proposed CPGAN, which applies compressive privacy to nonlinearly compress raw data before release. To alleviate the consequences of the adversary converging to a suboptimal solution, they proposed to integrate multiple adversaries including neural networks and non-neural network-based methods. Further, Chen et al. [12] considered perceptual indistinguishability (PI) as a formal privacy notion, particularly for images. Based on PI, they proposed PI-Net, a privacy-preserving mechanism that achieved image obfuscation with PI guarantee.

To defend against reconstruction and attribute-inference attacks simultaneously, two recent works—DISCO [13] and DeepObfuscator [14]—integrated the adversary for attribute-inference attack as well as that for reconstruction attack into a unifying framework and achieved state-of-the-art results for both types of attacks. In DeepObfuscator, a pre-training process is applied to facilitate the subsequent adversarial training. In DISCO, the privatizer contains three components, namely, pre-processing module, client network, and pruning network, which cooperate to further obfuscate the users' data. However, in both state-of-the-art works, their adversaries for attribute-inference attack optimize the cross-entropy loss, which has been shown to be unstable during training [28]; hence, they leave room for improvement. Our approach adopts a loss function similar to the loss in WGAN [19], whose loss landscape has been shown to be smoother than the cross-entropy loss. Our experiment demonstrated that the proposed method could improve these state-of-the-art performances.

C. Cryptography-Based Approaches

Another idea to prevent privacy leakage is to split a single user's data into multiple parts and provide them to different servers; consequently, no individual server can observe the intact user's data. Hence, it increases the difficulties of deciphering the privacy attribute from the data obtained from a single server. This is the fundamental idea of the secure multiparty computation (SMPC) protocol. Numerous studies [29], [30], [31] have proposed various cryptography-based approaches, which can provide rigorous security guarantees. However, these methods require a batch of distributed machines following the SMPC protocol. By contrast, the scenario considered in our study is more flexible without assuming any multiparty computation protocol required to be followed.

III. Threat Model and Attack

As mentioned in Section I, the issue of privacy leakage occurs even if the raw data are first obfuscated before release to the cloud for ML service. Accordingly, we consider the threat model discussed in Song et al.'s work [16] which gives a comprehensive survey of the threat models where the attacker aims to infer private information from the obfuscated data, as well as various attack [10], [11], and defense strategies [15], [27], [32]. Among various attack methods, our work focuses on proposing a better strategy for defending against the attribute-inference attack.

A. Threat Model

We define $\mathcal{X}$ as the space of raw data, and $\mathcal{Z}$ as the space of obfuscated data. The threat model comprises the following entities:

- $D_{train} \subset \mathcal{X}$ is a training dataset containing privacy information.
- $f_p : \mathcal{X} \to \mathcal{Z}$, the privatizer, which may be available in a white-box or black-box fashion. White-box access to the model reveals the model architecture and all parameters, while black-box access allows one to compute $f_p(x)$ where x is a sample from $\mathcal{X}$.
- $D_{target} = \{f_p(x_i^*)|x_i^* \in \mathcal{X}\}$, a set of obfuscated data obtained by applying privatizer to secret data x_i^*.
- $D_{adv} \subset \mathcal{X}$ is a dataset available to the adversary comprising of data drawn from the same distribution as D_{train}.

B. Attribute-Inference Attack

In this threat model, the adversary's goal is to infer the sensitive attribute s^* of a secret data x^* from a target-obfuscated data $f_p(x^*)$. We assume the adversary has access to f_p and a set of labeled data of the form (x, s), where $x \in D_{adv}$ and s is its corresponding label on the sensitive attribute (referred as privacy label in this manuscript). We focus on discrete attributes $s \in \mathcal{S}$, where $\mathcal{S}$ is the set of all classes of privacy attribute. The attributes s^* may be inferred by an

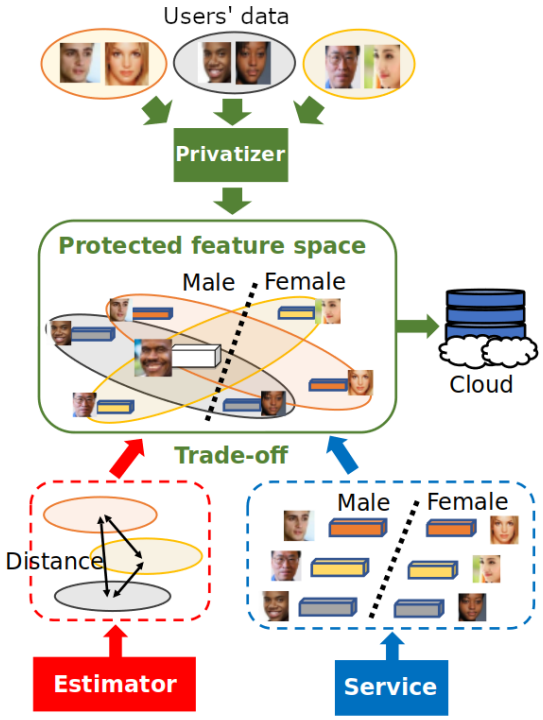

Fig. 3. Resulting architecture of the proposed method.

adversary who trains a classifier f_a on D_{adv}. This training algorithm is given by **Algorithm 1**. To demonstrate this threat model is realistic, we give a real-world example as follows. Assuming a cell phone app that has a privatizer f_p installed in the storage of the cell phone, the sensitive attribute is the users' race. The adversary can obtain f_p by installing this app on his/her cell phone. Then, this adversary uses publicly available datasets such as celebA with the label attribute of race as D_{adv}. A labeled dataset can be subsequently created by executing **Algorithm 1**.

Algorithm 1 Attribute Inference Attack (Algorithm 3 in [16])

1 **Input:** Target-obfuscated data $f_p(x^*)$, privatizer f_p, labeled adversary data D_{adv}.
2 Query f_p with D_{adv} and collect $\{(f_p(x_i), s_i)|x_i \in D_{adv}\}$.
3 Based on $\{(f_p(x_i), s_i)|x_i \in D_{adv}\}$, train a classifier $f_a : \mathcal{Z} \to \mathcal{S}$ that predicts the privacy label from the obfuscated data.
4 **return** $\hat{s} = f_a(f_p(x^*))$.

IV. Methodology

Our goal is to find a privatizer that obfuscates data before uploading to the cloud, so as to utilize the ML service while avoiding leakage of private information. Intuitively, the distributions of the obfuscated data with various sensitive labels should be "closely overlapping" in $\mathcal{Z}$, and therefore difficult to distinguish. Accordingly, we adopted the concept of WGAN [19] to remove the private information in obfuscated data. Furthermore, as the sensitive attributes usually contain more than two classes, we adopted the mathematical approach proposed by Yitong et al. [33] and proposed a privacy preserving class overlap (PPCO) method to obtain the privatizer through a data-driven approach.

Our proposed method is to replace the adversary in the architecture of generative adversarial privacy shown in Fig 2 with an estimator for class-overlapping. In the original architecture, the adversary aims to classify a sensitive attribute, which we replace with an estimator that measures how "closely overlapping" it is between obfuscated data points of different sensitive labels. The resulting architecture is shown in Fig 3. The reason for this replacement is because the adversary's loss function (i.e., cross-entropy loss) has been proved to be unstable [28] during training, while our loss function, which adopts the concept of WGAN [19] is shown to be more stable.

The entire architecture is trained through an iterative process, where each iteration contains two phases: In the first phase, the service network adopts its parameters to minimize the cross-entropy loss that measures the quality of the service based on the obfuscated data, whereas the estimator network computes the Wasserstein distance between data distributions of various sensitive labels in $\mathcal{Z}$ as an indication of the extent to which the data distributions of various sensitive labels are "closely overlapping." In the second phase, the parameters in both service and estimator networks are fixed, and the privatizer adopts its parameters to minimize the cross-entropy loss pertaining to the service network, as well as the Wasserstein distances evaluated by the estimator network. The objective functions of the estimator network, service network, and the privatizer are elaborated in the following sections.

A. Estimator

The estimator measures the extent to which it is "closely overlapping" it is between obfuscated data points of different sensitive labels. Suppose there are S classes of sensitive attributes, where the data points are divided into S distributions $P_1, P_2, \ldots, P_S$ according to their sensitive labels. Then, an intuitive way to quantify how obfuscated data points of different sensitive labels overlap is

$$\frac{2}{S(S-1)} \sum_{1 \le i < j \le S} W(f_p(P_i), f_p(P_j)), \tag{1}$$

where $f_p(P)$ denotes the distribution of $f_p(x)$ for which x is randomly drawn from distribution P, and $W(P_A, P_B)$ denotes the Wasserstein distance between distributions P_A and P_B.

However, when many types of sensitive attributes exist, the computation cost for (1) grows quadratically with S. To accelerate the computation, we instead only compute the Wasserstein distances between distribution pairs $(f_p(P_i), f_p(P_{/i}))$ for each $1 \le i \le S$, where $P_{/i}$ denotes the normalized sum of the remaining distributions:

$$P_{/i} = \frac{1}{S-1} \sum_{1 \le j \le S, \text{ and } j \ne i} P_j. \tag{2}$$

This leads to our design of the estimator which computes

$$\frac{1}{S}\sum_{i=1}^{S} W(f_p(P_i), f_p(P_{/i})) = \frac{1}{S}\sum_{i=1}^{S} \max_{\|f_i\|_L \le 1} (E_{x\sim P_i}[f_i(f_p(x))] - E_{x\sim P_{/i}}[f_i(f_p(x))]). \tag{3}$$

Here the second equality in (3) follows from the Kantorovich–Rubinstein duality [33]

$$W(P_A, P_B) = \max_{\|f\|_L \le 1} E_{x\sim P_A}[f(x)] - E_{x\sim P_B}[f(x)], \tag{4}$$

where $\|f\|_L \le 1$ indicates that the Lipschitz constant of the function $f(\cdot)$ is at most 1, i.e. $|f(x') - f(x)| \le ||x' - x||_2$, and the Lipschitz-smooth nonlinear function f can be approximated by a neural network [19].

It can be shown that Eq (4), and its gradient are well-defined in the domain of network parameters (see Theorem 3 in [19]). By contrast, in the conventional architecture of generative adversarial privacy shown in Fig 2, the adversary aims to optimize the cross-entropy loss. Reference [23] shows that the adversarial training with such loss function is to optimize the Jensen-Shannon divergence between P_A and P_B, as follows:

$$\max_{f_{adv}} \frac{1}{2}KL\left(P_A, \frac{P_A + P_B}{2}\right) + \frac{1}{2}KL\left(P_B, \frac{P_A + P_B}{2}\right) \tag{5}$$

where f_{adv} denotes the adversary and $KL(P_A, P_B)$ denotes KL-divergence between P_A and P_B. However, it has been shown that the gradient of Eq (5) is not well-defined in some cases (see Example 1 in [19]), serving as an evidence that Eq (4) is more stable than Eq (5) during adversarial training.

For the sake of computational efficiency, in our implementation the Lipschitz-smooth nonlinear functions $f_1, \ldots, f_S$ in (3) are approximated with a single neural network with S outputs $[f_1, \ldots, f_s]$. To enable the minimization of (3), we adopt the mathematical approach proposed by Yitong et al. [33] as follows: Let N be the number of training data, and n_s be the number of data whose sensitive attribute is of class s. Denoting $\pi_s = \frac{n_s}{N}$, the objective function that the estimator aims to maximize can be written as follows (cf.(3)):

$$\begin{aligned}
&\frac{1}{S}\sum_{i=1}^{S}\left(E_{x\sim P_i}[f_i(f_p(x))] - E_{x\sim P_{/i}}[f_i(f_p(x))]\right) \\
&= \frac{1}{S}\sum_{j=1}^{S} E_{x\sim P_j}[f_j(f_p(x))] \\
&\quad - \frac{1}{S(S-1)}\sum_{i\neq j} E_{x\sim P_j}[f_i(f_p(x))] \\
&= \frac{1}{S}\sum_{j=1}^{S} E_{x\sim P_j}\left[f_j(f_p(x)) - \frac{1}{S-1}\sum_{i\neq j} f_i(f_p(x))\right] \\
&= E_{s\sim\pi}\left[E_{x\sim P_s}\left[\frac{1}{S\pi_s}\gamma_s^T f_{EST}(f_p(x))\right]\right],
\end{aligned} \tag{6}$$

where $s \sim \pi$ indicates that s is a random variable that takes a value j with probability π_j, and $f_{EST}(y) = [f_1(y), \ldots, f_S(y)]^T$, and $\gamma_s = [\gamma_s^{(1)}, \ldots, \gamma_s^{(S)}]^T \in \mathbb{R}^S$ is defined as

$$\gamma_s^{(i)} = \begin{cases} -\dfrac{1}{S-1}, & \text{if } i \neq s \\ 1, & \text{if } i = s. \end{cases} \tag{7}$$

We may empirically approximate (6) with a mini-batch of data,

$$E_{s\sim\pi}\left[E_{x\sim P_s}\left[\frac{1}{S\pi_s}\gamma_s^T f_{EST}(f_p(x))\right]\right] \simeq \frac{1}{SN}\sum_{i=1}^{N}\pi_{s_i}^{-1}\gamma_{s_i}^T f_{EST}(f_p(x_i)), \tag{8}$$

where s_i represents the privacy label of x_i.

B. Service and Privatizer

Here we consider the classification service with C categories. The service adopts its parameters so as to minimize the cross-entropy loss that measures the quality of the service based on the obfuscated data. We denote $f_{SER} : \mathcal{Z} \to \mathcal{U}$, and $\mathcal{U} = \{u = [u^{(1)}, \ldots, u^{(C)}]^T \in \mathbb{R}^C | \ \|u\|_1 = 1, u^{(i)} \ge 0, \forall i = 1, \ldots, C\}$. The objective function of the service is:

$$\min_{f_{SER}} -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} u_i^{(c)} \log(f_{SER}^{(c)}(f_p(x_i))), \tag{9}$$

where $f_{SER}^{(c)}$ is the c^{th} entry of the output, which is the predicted score of the c^{th} category, and $u_i^{(c)}$ denotes the c^{th} entry of the label of x_i in one-hot representation.

The privatizer adopts its parameters to minimize the cross-entropy loss in (9) pertaining to the service, as well as the Wasserstein distances in (8) evaluated by the estimator. The complete objective function of PPCO is thus given by

$$\min_{f_p, f_{SER}} \max_{\|f_{EST}\|_L \le 1} -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} u_i^{(c)} \log(f_{SER}^{(c)}(f_p(x_i))) + \lambda\frac{1}{SN}\sum_{i=1}^{N}\pi_{s_i}^{-1}\gamma_{s_i}^T f_{EST}(f_p(x_i)), \tag{10}$$

where λ controls the trade-off between the estimator and the service.

C. Algorithm

We denote θ_p as the parameters in the privatizer, θ_{EST} as the parameters in the estimator, and θ_{SER} as the parameters in the service. The training process of PPCO is illustrated in Algorithm 2. In lines 3–7, the objective function of the estimator is evaluated, and the parameters in the estimator are updated through gradient ascent. Here we choose $k_E = 7$ as the number of inner loop iterations (line 6), and we apply the spectrum normalization [34] to implement the Lipschitz constraint. In lines 8–10, the objective function of service is evaluated, and the parameters in the service are updated through gradient descent. In lines 11–12, the objective function of the privatizer is evaluated. The objective function of the privatizer is the weighted sum of the objective functions of the service and the estimator, and the parameters in the privatizer are updated through gradient descent.

Algorithm 2 Training Process

for $iter = 1$ *to* $maxiter$ **do**
  Sample a mini-batch of m data points $\{(x_i, u_i, s_i)\}_{i=1}^{m}$ from D_{train}.
  Compute estimator loss $L_E = \frac{1}{Sm} \sum_{i=1}^{m} \pi_{s_i}^{-1} \gamma_{s_i}^{T} f_{EST}(f_p(x_i))$.
  for $iter_E = 1$ *to* k_E **do**
    Compute $\nabla\theta_{EST} = \frac{\partial L_E}{\partial \theta_{EST}}$.
    Update $\theta_{EST} \leftarrow \theta_{EST} + \eta_{EST} \nabla\theta_{EST}$.
  end
  Compute service loss $L_S = -\frac{1}{m} \sum_{i=1}^{m} \sum_{c=1}^{C} u_i^{(c)} \log(f_{SER}^{(c)}(f_p(x_i)))$.
  Compute $\nabla\theta_{SER} = \frac{\partial L_S}{\partial \theta_{SER}}$.
  Update $\theta_{SER} \leftarrow \theta_{SER} - \eta_{SER} \nabla\theta_{SER}$.
  Compute $\nabla\theta_P = \frac{\partial (L_S + \lambda L_E)}{\partial \theta_p}$.
  Update $\theta_p \leftarrow \theta_p - \eta_p \nabla\theta_P$.
end

D. Approaches to Enhance the State-of-the-Art Architecture

Our class-overlapping estimator can be integrated into the state-of-the-art of generative adversarial privacy whose architecture is similar to the one shown in Fig. 2. For example, DeepObfuscator [14] and DISCO [13] are two of the state-of-the-art implementations that follow this architecture. Furthermore, they contain adversaries for both reconstruction and attribute-inference attacks. To enhance these state-of-the-art architectures against an attribute-inference attack, we can replace their adversary for sensitive attribute with the proposed estimator; hence, they can achieve the effect of class overlapping of PPCO. The followings show the detail of the implementation of this enhancement.

1) DeepObfuscator: The DeepObfuscator adopts the cross-entropy loss in their adversary for the sensitive attribute. We replace it with our estimator with loss function Eq (4). The remaining architecture and training procedures are the same as their original implementation.

2) DISCO: The architecture of DISCO is more sophisticated than DeepObfuscator. Its privatizer contains three components: pre-processing module, client network, and pruning network. In addition to the client network, the other two additional components further obfuscate the input images: the pre-processing module splits an image into several patches, and the pruning network prunes excess channels. To enhance its architecture with our proposed method, we replace its adversary with our estimator as well. Nevertheless, in the adversarial training of the original DISCO, most of the components in the privatizer are not affected by the gradient of adversarial loss except for the Dynamic Channel Pruning Network, which protects the data by masking out some channels of the feature map produced by the client network. By contrast, in our enhanced PPCO implementation, the whole privatizer will be updated by our estimator's loss to be encouraged to protect the privacy information. We make this change because our loss aims at making two distributions similar. The original implementation which filters out some channels is not strong enough to achieve this purpose.

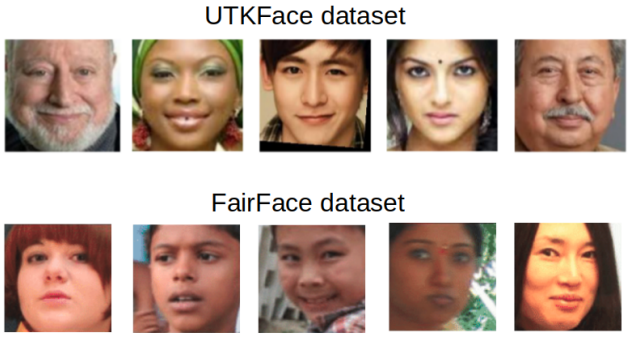

Fig. 4. Example of experiment data.

TABLE I
NUMBER OF IMAGES IN UTKFACE

Race	White	Black	Asian	Indian	Others	Total
Total	10078	4526	3434	3975	1692	23705
(%)	42.51(%)	19.09(%)	14.49(%)	16.77(%)	7.14(%)	100(%)
Training	6365	2897	2151	2528	1059	15000
Validation	1696	729	577	698	300	4000
Testing	2017	900	706	749	333	4075

E. Applicability and Limitation

To apply our approach, first we assume that we have an environment that allows jointly training the privatizer, estimator, and service, and adequate amount of training data is available. After training, the trained model of the privatizer could be distributed into users' machines. Second, a user should have sufficient computational resources to execute the privatizer because it is a neural network that requires substantial computational resources. The limitation of the proposed method is that it can only enhance protection against the attribute-inference attack. Other privacy attack methods, such as reconstruction attack, are beyond the scope of the proposed method. Another limitation is that the obfuscated data could only be used for the utilities (e.g., gender) that the privatizer has been trained for. It could not be used for other utilities (e.g., age) after training.

V. EXPERIMENT DATASET

A. UTKFace

We first measured the performance of the proposed PPCO method on the UTKFace dataset [35] from which some sampled images are illustrated in Fig. 4. UTKFace is a set of 23,705 face images labeled with age, gender, and race, which we rescaled into 32 × 32 RGB pixels. The goal of this task was to predict gender, and the sensitive attribute to be protected was race.

The UTKFace dataset was selected for specific reasons. First, it is widely adopted in studies related to attribute-inference attack [10], [17], [36]. Furthermore, the sensitive attribute—race—in UTKFace is highly unbalanced, as presented in TABLE III, so that we can analyze how the training of the privatizer will be influenced by the imbalanced distribution of sensitive attributes.

We divided the 23,705 images into 15000/4000/4075 training/validation/testing sets, respectively. We further recognized that the distribution of races in the original dataset was not balanced. To observe this, as presented in TABLE I, the White race contains 10,078 images, while other races such as Black,

TABLE II
SETTING OF UTKFACE

	White	Black	Asian	Indian	Others
D_{train} (Balance)	500	500	500	500	500
D_{train} (Imbalance)	5000	100	100	10	10
D_{adv}	500	500	500	500	500
Validation	300	300	300	300	300
Testing	300	300	300	300	300

TABLE III
NUMBER OF IMAGES IN FAIRFACE

Race	East Asian	Indian	Black	White	Middle Eastern	Latino Hispanic	Southeast Asian
Total	13837	13835	13789	18612	10425	14990	12210
(%)	14.2(%)	14.2(%)	14.1(%)	19.0(%)	10.7(%)	15.3(%)	12.5(%)
Training	12287	12319	12233	16527	9216	13367	10795
Validation	1550	2085	1623	1415	1556	1516	1209

Asian, Indian, and Others contain less than 5000 images per race.

To analyze how the training of the privatizer would be influenced by balanced and/or imbalanced race distributions, we resampled the training data with the settings described in TABLE II. Under the balanced setting, D_{train} was constructed by sampling 500 images from each race, while under the imbalanced setting, D_{train} was composed of 5000/100/100/10/10 images among various races. Further, D_{adv} was constructed by sampling 500 images from each race, where the sampled images in D_{adv} were distinct from those in D_{train}. In the validation and testing sets, we randomly selected 300 images from each sensitive attribute. In short, 1500 images were selected from the 4000/4075 images in the original validation/testing sets, respectively. Finally, the gender-and race-classification accuracies were both evaluated on the testing set.

B. FairFace

The FairFace [37] dataset comprises 108,501 images, with race, gender, and age groups. This dataset is designed with emphasis on balanced race composition. The number of images for each race is given in TABLE III. The goal of using this dataset was to compare with the state-the-art work, DISCO [13], because the authors adopted this dataset in their experiment, in which the task attribute was gender and the sensitive attribute was race.

We adopt the setting used in [13] for splitting the training, validation set as well. The number of samples in each set is given in TABLE III. Notice that in their setting, they follow the same training and validation split as in [37]. Furthermore, the authors assumed that their adversary can directly access D_{train}, and therefore they did not split another dataset D_{adv} to train their adversary.

VI. EXPERIMENT SETTING AND RESULT

A. UTKFace

In the experiment with UTKFace, we compared the following methods:

- **ADV-CE** applies adversarial training to train the privatizer. The privatizer is trained to minimize the negative cross-entropy loss evaluated by the adversary [18] as well as cross-entropy loss pertaining to the service.
- **ADV-PPCO** is the enhancement of **ADV-CE**, by replacing the adversary with the proposed estimator.
- **DeepObfuscator** [14] is a state-of-the-art work in generative adversarial privacy, while its adversary is still trained to maximize the negative cross-entropy loss against the privatizer.
- **DeepObfuscator-PPCO** is the enhancement of DeepObfuscator by applying our PPCO method.
- **A deep neural network (DNN)** directly trains a stacked neural network model by the objective function of the service. After training, the neural network model is separated into two parts: the preceding multiple layers work as a privatizer and the remaining layers make predictions for the task of the service.

Training detail: For ADV-CE, and ADV-PPCO, we choose LeNet [39] as the model architecture. The privatizer is the convolution part of LeNet, and the service is the fully connected part of LeNet. For a fair comparison with these methods, they adopt the same neural network architecture and best-selected hyperparameters as presented in TABLE IV. We employ the training process presented in Section IV-C to train ADV-CE and ADV-PPCO.

For a fair comparison between DeepObfuscator, DeepObfuscator-PPCO, and DNN, we followed the same architecture, hyperparameters, and learning algorithm proposed in [14], while the only difference between DeepObfuscator and DeepObfuscator-PPCO was that we replaced its adversary for the sensitive attribute by the proposed estimator, and DNN is implemented by removing the adversarial loss of DeepObfuscator. To implement DeepObfuscator, DeepObfuscator-PPCO, and DNN, we adopt a publicly available implementation of DeepObfuscator in https://github.com/splitlearning/InferenceBenchmark.

Note that ADV-CE was not customized for imbalanced settings, where the adversary's objective function in its original design was the summation of losses from each individual data point with equal weights. For comparison under imbalanced settings, in the experiment we adjusted the weights of data points from race categories in White/Black/Asian/Indian/Others as 5220/5000, 5220/100, 5220/100, 5220/10, and 5220/10, respectively, to account for the various amounts between different sensitive attributes.

For the attribute-inference attack, we simulated the adversary using five classification methods: neural network, logistic regression, support vector machine, decision tree, and gradient boosting. The maximum accuracy among the five classification methods is reported herein. For those non-neural-network classifiers, we use the implementations of these classifiers provided by *scikit-learn* package [38]. These implementation are *LinearSVC*, *LogisticRegression*, *RandomForestClassifier* and *GradientBoostingClassifier*, and their hyper-parameters are given in TABLE IV. The incentive behind considering different classification methods that could be adopted by the adversary was to prevent overestimating the protection ability of the privatizer. As such, one evaluates the gender recognition performance (by the service) and the race recognition performance (by the adversary) achievable with the obfuscated data extracted from the privatizer. We repeated the training and evaluation procedures five times; herein, we report the average accuracy.

TABLE IV
IMPLEMENTATION DETAILS OF UTKFACE EXPERIMENT

NETWORK ARCHITECTURE FOR ADV-PPCO AND ADV-CE

	Layers	Optimizer	Learning rate	Input size	Output size
Privatizer	Convolution with ReLu, stride = 5, unit = 6 Batch normalization Max pooling = 2 Convolution with ReLu, stride = 5, unit = 16 Batch normalization Max pooling = 2 flatten	Adam	0.001	[batch size, 32, 32, 3]	[batch size, 400]
Service	Fully connect with ReLu, unit = 120 Batch normalization Fully connect with ReLu, unit = 84 Batch normalization Fully connect with ReLu, unit = 2	Adam	0.001	[batch size, 400]	[batch size, 2]
Estimator	Fully connect with LeakyReLu, unit = 64 Fully connect with LeakyReLu, unit = 32 Fully connect , unit = 5 All parameter will use spectral normalization	Adam	0.001	[batch size, 400]	[batch size, 5]
epochs	300				

HYPERPARAMETERS OF ATTRIBUTE INFERENCE ATTACK

support vector machine*	logistic regression*	decision tree*	gradient boosting*	neural network
penalty=l2 loss=squared_hinge dual=True tol=0.0001 C=1.0 multi_class=ovr fit_intercept=True intercept_scaling=1 class_weight=None verbose=0 random_state=None max_iter=1000	penalty=l2 dual=False tol=0.0001 C=1.0 fit_intercept=True intercept_scaling=1 class_weight=None random_state=None solver=lbfgs max_iter=100 multi_class=auto verbose=0 warm_start=False n_jobs=None l1_ratio=None	n_estimators=100 criterion=gini max_depth=None min_samples_split=2 min_samples_leaf=1 min_weight_fraction_leaf=0.0 max_features=sqrt max_leaf_nodes=None min_impurity_decrease=0.0 bootstrap=True oob_score=False n_jobs=None random_state=None verbose=0 warm_start=False class_weight=None ccp_alpha=0.0 max_samples=None	loss=log_loss learning_rate=0.1 n_estimators=100 subsample=1.0 criterion=friedman_mse min_samples_split=2 min_samples_leaf=1 min_weight_fraction_leaf=0.0 max_depth=3 min_impurity_decrease=0.0 init=None random_state=None max_features=None verbose=0 max_leaf_nodes=None warm_start=False validation_fraction=0.1 n_iter_no_change=None tol=0.0001 ccp_alpha=0.0	Fully connect with ReLu, unit = 64 Batch normalization Fully connect with ReLu, unit = 64 Batch normalization Fully connect with ReLu, unit = 5 Adam with learning rate = 0.001

*For the non-neural-network attacker, we used sklearn default parameters, which can be found in [38].

Results and Analysis: The experimental results of the proposed methods and other baseline methods are summarized in Figs. 5 and 6, showing the trade-off between the accuracies of gender and race classification. A desirable trade-off should achieve maximal gender accuracy and minimal race accuracy; thus, a point closer to the top-left corner corresponds to a privatizer with better performance.

The multiple points in these figures indicate the results of varying trade-off factor λ (cf. (10)), respectively. The value of λ was decided by the experiment. First, we selected the value of λ within a relatively large range, and then gradually refined the value of it. The plots of utility and privacy protection trade-off are shown by this way, and the range of λ is between 0.1 and 50.

Fig. 5 shows the results obtained under balanced settings. DNN, which is directly optimized for gender-classification performance but not for privacy protection, achieves high gender-classification accuracy at the cost of the highest race-classification accuracy achievable by the adversary. Compared to DNN, other methods demonstrated better utility–privacy trade-off. Among them, DeepObfuscator-PPCO dominated the other methods, particularly DeepObfuscator, which demonstrated that the proposed method could be applied to improve the state-of-the-art work. Notice that DeepObfuscator did not perform better than ADV-CE, because its hyperparameters were from [14], and hence they were not tailored for UTKFace dataset. Nevertheless, despite this poor choice of hyperparameters, DeepObfuscator-PPCO could still dominate the other methods. By contrast, ADV-PPCO did not show better performance compared to ADV-CE under balanced settings; therefore, we further compared ADV-PPCO and ADV-CE under imbalanced settings.

Fig. 6 shows the results obtained under the imbalanced settings. A comparison of ADV-PPCO and ADV-CE with weighting revealed that ADV-PPCO obviously achieved better trade-off. Furthermore, under imbalanced settings, even though adding class-dependent weights improved the utility–privacy trade-off for ADV-CE, ADV-PPCO achieved significantly better utility–privacy trade-off, indicating better privacy-protection capabilities.

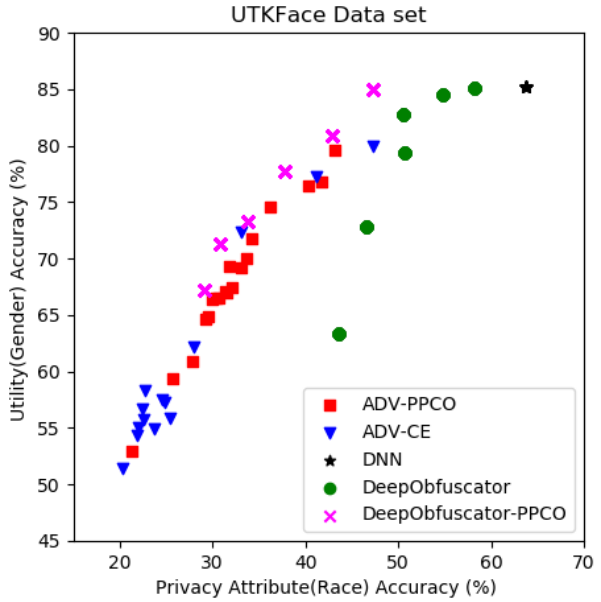

Fig. 5. Utility and privacy protection trade-off under balanced settings.

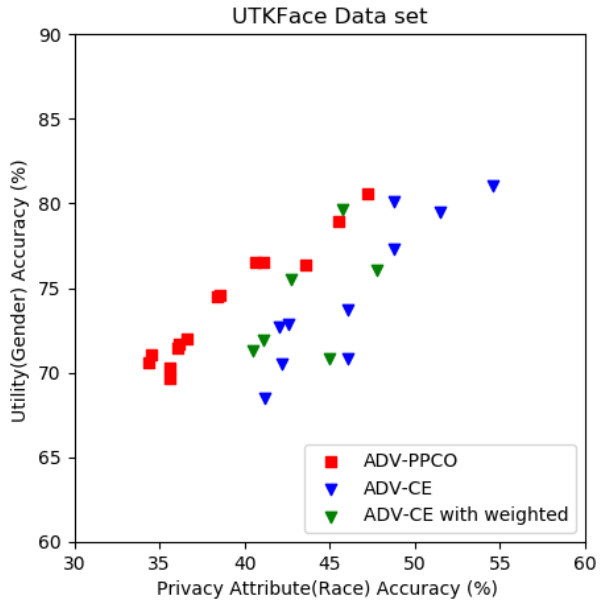

Fig. 6. Utility and privacy protection trade-off under imbalanced settings.

B. FairFace

In the experiment with FairFace, we compared the following methods:

- **DISCO** [13] is a state-of-the-art work in generative adversarial privacy. Despite its sophisticated architecture, its adversary is still trained to maximize the negative cross-entropy loss against the privatizer.
- **DISCO-PPCO** is the enhancement of DISCO by applying the proposed PPCO method.

Training detail: For a fair comparison between these two methods, we followed the architecture and learning algorithm proposed in [13], with the only difference between DISCO and DISCO-PPCO being that we replaced its adversary for sensitive attribute with the proposed estimator. To implement DISCO and DISCO-PPCO, we adopt a publicly available implementation of DISCO in https://github.com/splitlearning/InferenceBenchmark.

TABLE V

UTILITY (GENDER) ACCURACY AND PRIVACY ATTRIBUTE (RACE) ACCURACY OF DISCO AND DISCO-PPCO

	Utility (Gender)	Privacy Attribute (Race)
DISCO	0.791 ± 0.053	0.204 ± 0.087
DISCO-PPCO	$\mathbf{0.792 \pm 0.056}$	$\mathbf{0.192 \pm 0.014}$

For the attribute-inference attack of DISCO, we followed the same settings as in [13], in which the authors reported the accuracy of attribute-inference attack during the adversarial training process. However, in the DISCO-PPCO architecture, we only included the estimator and no adversary during the adversarial training process, as shown in Algorithm 2. Hence, we created an additional adversary for DISCO-PPCO, which was only for evaluating the accuracy of the attribute-inference attack. To ensure that this adversary would not affect the other components in DISCO-PPCO, we blocked its gradient from backpropagating to the other components in DISCO-PPCO. We repeated the training and evaluation procedures five times; herein, we report the average accuracy and standard deviation in TABLE V.

Results and Analysis: The experimental results of DISCO and DISCO-PPCO are summarized in TABLE V. DISCO-PPCO achieved slightly better gender accuracy than DISCO and simultaneously achieves lower race accuracy. Note that a desirable trade-off should achieve maximal gender accuracy and minimal race accuracy; thus, we validate that our proposed method can improve the performance of DISCO.

VII. THEORETICAL ANALYSIS

In this section, we analyze the theoretical performance of PPCO under Gaussian mixture models, in which the theoretical analysis is tractable, so that we can compare with empirical results to give a flavour of the gap between theoretical and empirical performances.

The intuition of this analysis is as follows. First, we assume that the input data distribution is Gaussian mixture models. Each Gaussian model has a unique label for both utility and privacy. For example, in race-preserving gender classification, the white males' images are from a Gaussian model, and the black females' images are from another Gaussian model. Then, a trained PPCO privatizer, represented by a linear matrix, projects these gaussian models into a low-dimensional feature space for privacy protection. Because this PPCO privatizer is a linear matrix, the data distributions in this low-dimensional space are still Gaussian mixture models. By maximum, a posterior (MAP) estimation on utility or privacy, the theoretical optimal utility accuracy loss, and privacy loss can be obtained. Finally, we compare these theoretical losses with empirical losses.

A. Gaussian Mixture Model Settings

We followed a theoretical analysis method similar to that given in [27]. Consider the setting where utility label $Y_1 \in \{0, 1\}$, and privacy label $Y_2 \in \{0, 1\}$ such that input data $X \in \mathbb{R}^m$ is a Gaussian random variable whose mean and

covariance matrix are dependent on Y. In particular,

$$\begin{aligned} X|_{Y_1=0,Y_2=0} &\sim \mathcal{N}(\boldsymbol{\mu}_{00}, \Sigma_{00}), \quad \mathbb{P}[Y_1=0, Y_2=0] = p_{00} \\ X|_{Y_1=0,Y_2=1} &\sim \mathcal{N}(\boldsymbol{\mu}_{01}, \Sigma_{01}), \quad \mathbb{P}[Y_1=0, Y_2=1] = p_{01} \\ X|_{Y_1=1,Y_2=0} &\sim \mathcal{N}(\boldsymbol{\mu}_{10}, \Sigma_{10}), \quad \mathbb{P}[Y_1=1, Y_2=0] = p_{10} \\ X|_{Y_1=1,Y_2=1} &\sim \mathcal{N}(\boldsymbol{\mu}_{11}, \Sigma_{11}), \quad \mathbb{P}[Y_1=1, Y_2=1] = p_{11}. \end{aligned} \tag{11}$$

We also note that

$$\begin{aligned} \mathbb{P}[Y_1=0] &= p_0^{Y_1} = p_{00} + p_{01} \\ \mathbb{P}[Y_1=1] &= p_1^{Y_1} = p_{10} + p_{11} \\ \mathbb{P}[Y_2=0] &= p_0^{Y_2} = p_{00} + p_{10} \\ \mathbb{P}[Y_2=1] &= p_1^{Y_2} = p_{01} + p_{11}. \end{aligned} \tag{12}$$

To make the problem more tractable, we make the following simplifications:

- The four Gaussian distributions have identical co-variance matrices $\Sigma_{00} = \Sigma_{01} = \Sigma_{10} = \Sigma_{11} = \Sigma$.
- The privatizer is a linear mapping $Z = \mathbf{A}X$, where $\mathbf{A} \in \mathbb{R}^{d\times m}$, and $Z \in \mathbb{R}^d$.
- X has zero mean, namely $p_{00}\boldsymbol{\mu}_{00} + p_{01}\boldsymbol{\mu}_{01} + p_{10}\boldsymbol{\mu}_{10} + p_{11}\boldsymbol{\mu}_{11} = \mathbf{0}$.
- Y_1 and Y_2 are independent random variables: $\mathbb{P}[Y_1 = y, Y_2 = y] = \mathbb{P}[Y_1 = y]\mathbb{P}[Y_2 = y]$

For the sake of analysis, we may write $X = \Upsilon + \Xi$, where $\Upsilon = \boldsymbol{\mu}_{Y_1Y_2}$ and $\Xi \sim \mathcal{N}(\mathbf{0}, \Sigma)$ are independent r.v.s. Moreover, we note that $\mathbf{R}_\Xi = \mathbb{E}\left[\Xi\Xi^T\right] = \Sigma$.

B. Utility Perspective

The utility loss is measured as the error in predicting Y_1 from Z.

$$\begin{aligned} Z|_{Y_1=0,Y_2=0} &= \mathbf{A}(\boldsymbol{\mu}_{00} + \Xi) \sim \mathcal{N}\left(\mathbf{A}\boldsymbol{\mu}_{00}, \mathbf{A}\mathbf{R}_\Xi\mathbf{A}^T\right) \\ Z|_{Y_1=0,Y_2=1} &= \mathbf{A}(\boldsymbol{\mu}_{01} + \Xi) \sim \mathcal{N}\left(\mathbf{A}\boldsymbol{\mu}_{01}, \mathbf{A}\mathbf{R}_\Xi\mathbf{A}^T\right) \\ Z|_{Y_1=1,Y_2=0} &= \mathbf{A}(\boldsymbol{\mu}_{10} + \Xi) \sim \mathcal{N}\left(\mathbf{A}\boldsymbol{\mu}_{10}, \mathbf{A}\mathbf{R}_\Xi\mathbf{A}^T\right) \\ Z|_{Y_1=1,Y_2=1} &= \mathbf{A}(\boldsymbol{\mu}_{11} + \Xi) \sim \mathcal{N}\left(\mathbf{A}\boldsymbol{\mu}_{11}, \mathbf{A}\mathbf{R}_\Xi\mathbf{A}^T\right). \end{aligned} \tag{13}$$

Therefore

$$\mathbb{P}[Y_1 = 1 \mid Z = z] = \frac{\sum_j p_{1j}\mathcal{N}(z; \mathbf{A}\boldsymbol{\mu}_{1j}, \mathbf{A}\mathbf{R}_\Xi\mathbf{A}^T)}{\sum_{ij} p_{ij}\mathcal{N}(z; \mathbf{A}\boldsymbol{\mu}_{ij}, \mathbf{A}\mathbf{R}_\Xi\mathbf{A}^T)}. \tag{14}$$

Denoting $\mathcal{Z}_1$ as the set of z in which $\mathbb{P}[Y_1 = 1 \mid Z = z] \geq \mathbb{P}[Y_1 = 0 \mid Z = z]$, namely

$$\begin{aligned} \mathcal{Z}_1 = \Bigg\{ z : &\sum_j p_{1j}\mathcal{N}\left(z; \mathbf{A}\boldsymbol{\mu}_{1j}, \mathbf{A}\mathbf{R}_\Xi\mathbf{A}^T\right) \\ &\geq \sum_j p_{0j}\mathcal{N}\left(z; \mathbf{A}\boldsymbol{\mu}_{0j}, \mathbf{A}\mathbf{R}_\Xi\mathbf{A}^T\right) \Bigg\}. \end{aligned} \tag{15}$$

The maximum a posteriori (MAP) estimation of Y_1 given z is

$$\hat{Y}_1 = \begin{cases} 1, & \text{if } z \in \mathcal{Z}_1 \\ 0, & \text{otherwise.} \end{cases} \tag{16}$$

Then, the average error of predicting Y_1 over $z \in \mathbb{R}^d$ is

$$\begin{aligned} &\mathbb{P}\left[\hat{Y}_1 \neq Y\right] \\ &= \mathbb{P}\left[\left(Y_1, \hat{Y}_1\right) = (0,1)\right] + \mathbb{P}\left[\left(Y_1, \hat{Y}_1\right) = (1,0)\right] \\ &= \sum_j p_{0j} \int_{z\in\mathcal{Z}_1} \frac{e^{\frac{-1}{2}(z-\mathbf{A}\boldsymbol{\mu}_{0j})^T\left(\mathbf{A}\mathbf{R}_\Xi\mathbf{A}^T\right)^{-1}(z-\mathbf{A}\boldsymbol{\mu}_{0j})}}{(2\pi)^{0.5d}\det\left(\mathbf{A}\mathbf{R}_\Xi\mathbf{A}^T\right)^{0.5}} dz \\ &\quad + \sum_j p_{1j} \int_{z\notin\mathcal{Z}_1} \frac{e^{\frac{-1}{2}(z-\mathbf{A}\boldsymbol{\mu}_{1j})^T\left(\mathbf{A}\mathbf{R}_\Xi\mathbf{A}^T\right)^{-1}(z-\mathbf{A}\boldsymbol{\mu}_{1j})}}{(2\pi)^{0.5d}\det\left(\mathbf{A}\mathbf{R}_\Xi\mathbf{A}^T\right)^{0.5}} dz. \end{aligned} \tag{17}$$

C. Privacy Perspective

Similarly, for $Y_2 = 1$, we have

$$\mathbb{P}[Y_2 = 1 \mid Z = z] = \frac{\sum_i p_{i1}\mathcal{N}(z; \mathbf{A}\boldsymbol{\mu}_{i1}, \mathbf{A}\mathbf{R}_\Xi\mathbf{A}^T)}{\sum_{ij} p_{ij}\mathcal{N}(z; \mathbf{A}\boldsymbol{\mu}_{ij}, \mathbf{A}\mathbf{R}_\Xi\mathbf{A}^T)}. \tag{18}$$

Denoting $\mathcal{Z}_2$ as the set of z in which $\mathbb{P}[Y_2 = 1 \mid Z = z] \geq \mathbb{P}[Y_2 = 0 \mid Z = z]$, namely

$$\begin{aligned} \mathcal{Z}_2 = \Bigg\{ Z : &\sum_i p_{i1}\mathcal{N}\left(z; \mathbf{A}\boldsymbol{\mu}_{i1}, \mathbf{A}\mathbf{R}_\Xi\mathbf{A}^T\right) \\ &\geq \sum_i p_{i0}\mathcal{N}\left(z; \mathbf{A}\boldsymbol{\mu}_{i0}, \mathbf{A}\mathbf{R}_\Xi\mathbf{A}^T\right) \Bigg\}. \end{aligned} \tag{19}$$

The MAP estimation of Y_2 given z is

$$\hat{Y}_2 = \begin{cases} 1, & \text{if } z \in \mathcal{Z}_2 \\ 0, & \text{otherwise.} \end{cases} \tag{20}$$

Then, the average error of predicting Y_2 over $z \in \mathbb{R}^d$ is

$$\begin{aligned} &\mathbb{P}\left[\hat{Y}_2 \neq Y\right] \\ &= \mathbb{P}\left[\left(Y_2, \hat{Y}_2\right) = (0,1)\right] + \mathbb{P}\left[\left(Y_2, \hat{Y}_2\right) = (1,0)\right] \\ &= \sum_i p_{i0} \int_{z\in\mathcal{Z}_2} \frac{e^{\frac{-1}{2}(z-\mathbf{A}\boldsymbol{\mu}_{i0})^T\left(\mathbf{A}\mathbf{R}_\Xi\mathbf{A}^T\right)^{-1}(z-\mathbf{A}\boldsymbol{\mu}_{i0})}}{(2\pi)^{0.5d}\det\left(\mathbf{A}\mathbf{R}_\Xi\mathbf{A}^T\right)^{0.5}} dz \\ &\quad + \sum_i p_{i1} \int_{z\notin\mathcal{Z}_2} \frac{e^{\frac{-1}{2}(z-\mathbf{A}\boldsymbol{\mu}_{i1})^T\left(\mathbf{A}\mathbf{R}_\Xi\mathbf{A}^T\right)^{-1}(z-\mathbf{A}\boldsymbol{\mu}_{i1})}}{(2\pi)^{0.5d}\det\left(\mathbf{A}\mathbf{R}_\Xi\mathbf{A}^T\right)^{0.5}} dz. \end{aligned} \tag{21}$$

D. Optimal Utility/Privacy Losses

In summary, the optimal utility/privacy losses given a specific privatization scheme (as represented by $\mathbf{A}$ under linear privatization setting), are given as follows:

- The optimal utility loss by the service provider:

$$\begin{aligned} L_{util}^{(opt)}(\mathbf{A}) &= \mathbb{P}\left[\hat{Y}_1 \neq Y\right] \\ &= \sum_j p_{0j} \int_{z\in\mathcal{Z}_1} \frac{e^{\frac{-1}{2}(z-\mathbf{A}\boldsymbol{\mu}_{0j})^T\left(\mathbf{A}\mathbf{R}_\Xi\mathbf{A}^T\right)^{-1}(z-\mathbf{A}\boldsymbol{\mu}_{0j})}}{(2\pi)^{0.5d}\det\left(\mathbf{A}\mathbf{R}_\Xi\mathbf{A}^T\right)^{0.5}} dz \\ &\quad + \sum_j p_{1j} \int_{z\notin\mathcal{Z}_1} \frac{e^{\frac{-1}{2}(z-\mathbf{A}\boldsymbol{\mu}_{1j})^T\left(\mathbf{A}\mathbf{R}_\Xi\mathbf{A}^T\right)^{-1}(z-\mathbf{A}\boldsymbol{\mu}_{1j})}}{(2\pi)^{0.5d}\det\left(\mathbf{A}\mathbf{R}_\Xi\mathbf{A}^T\right)^{0.5}} dz. \end{aligned} \tag{22}$$

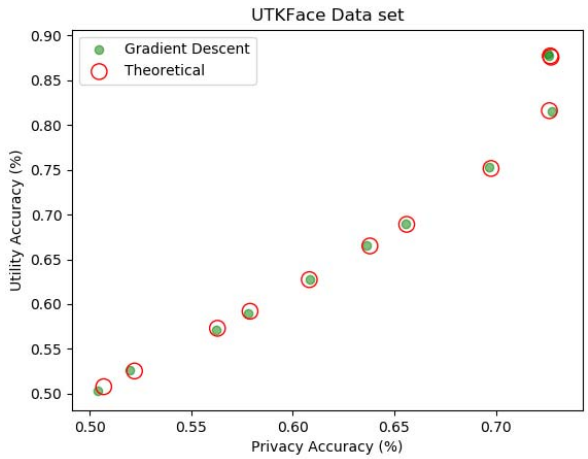

Fig. 7. Under the linear case, PPCO achieves utility/privacy trade-off nearly identical to the theoretical solution under various trade-off factors λ (cf. (10)).

- The optimal privacy loss by the adversary:

$$L_{adv}^{(opt)}(\mathbf{A}) = \mathbb{P}\left[\hat{Y}_2 \neq Y\right] = \sum_i p_{i0} \int_{z\in\mathcal{Z}_2} \frac{e^{\frac{-1}{2}(z-\boldsymbol{A}\boldsymbol{\mu}_{i0})^T\left(\mathbf{A}\mathbf{R}_\Xi\mathbf{A}^T\right)^{-1}(z-\boldsymbol{A}\boldsymbol{\mu}_{i0})}}{(2\pi)^{0.5d}\det\left(\mathbf{A}\mathbf{R}_\Xi\mathbf{A}^T\right)^{0.5}} dz + \sum_i p_{i1} \int_{z\notin\mathcal{Z}_2} \frac{e^{\frac{-1}{2}(z-\boldsymbol{A}\boldsymbol{\mu}_{i1})^T\left(\mathbf{A}\mathbf{R}_\Xi\mathbf{A}^T\right)^{-1}(z-\boldsymbol{A}\boldsymbol{\mu}_{i1})}}{(2\pi)^{0.5d}\det\left(\mathbf{A}\mathbf{R}_\Xi\mathbf{A}^T\right)^{0.5}} dz. \tag{23}$$

E. Comparison Between Empirical and Theoretical Results

To compare the utility and adversary losses achieved by PPCO through gradient descent with the theoretical results as given by (22) and (23), we conducted an experiment using the settings in Sec.VII-A, and we generated a toy-example synthetic dataset. To investigate the trade-off between utility and privacy loss, we consider the case where the feature to classify the privacy attribute is correlated to the feature for utility classification. If they are not correlated, the feature for privacy attribute can be completely removed without affecting the utility classification. For example, in the case of $m = 2$, $d = 1$, and $\mu_{00} = [2, 2]$, $\mu_{01} = [2, -2]$, $\mu_{10} = [-2, 2]$, $\mu_{11} = [-2, -2]$, a projection onto the first dimension can perfectly remove the feature for classifying Y_2, without affecting the classification accuracy of Y_1. Thus, in this analysis we focus on the more interesting case such as $\mu_{00} = [2, 2]$, $\mu_{01} = [-2, -2]$, $\mu_{10} = [-2, 2]$, $\mu_{11} = [2, -2]$, with probability $p_{00} = p_{10} = p_{01} = p_{11} = \frac{1}{4}$, and $\Sigma = \mathbf{I}_2$ is a 2×2 identity matrix. We synthesized $40k$ samples for training PPCO, $40k$ for training the attacker, $10k$ and $10k$ for validation and testing, respectively. The privatizer was a linear classifier with parameter $\mathbf{A}$, while the service and estimator in PPCO were two-layer perceptrons with the $6d$-width hidden layer. We trained this PPCO using the Adam optimizer with a learning rate 0.001.

The empirical and theoretical results of the utility/privacy loss over 10k synthetic validation samples are compared in Figure 7. Each dot corresponds to a trade-off factor $\lambda \in \{0.01, 0.1, 0.2, 0.3, \ldots, 0.7, 0.8, 0.9, 0.99\}$. Note that for each λ, the privatizer (represented by $\mathbf{A}$) is determined by minimizing $L_{ppcon}(\mathbf{A}) = (1-\lambda)L_{util}^{(opt)}(\mathbf{A}) - \lambda L_{adv}^{(opt)}(\mathbf{A})$ through gradient descent. Fig. 7 reveals that the utility-privacy trade-off of the empirical result is nearly consistent with the theoretical results (namely $L_{util}^{(opt)}(\mathbf{A})$ and $L_{adv}^{(opt)}(\mathbf{A})$). In addition, it shows that varying λ has a significant influence on the privacy-utility trade-off.

VIII. Conclusion

In this work, we propose PPCO as a privacy-preserving ML method that incorporates both WGAN and the idea of class overlapping. PPCO can be applied to improve state-of-the-art methods such as DeepObfuscator and DISCO, and the resulting network, DeepObfuscator-PPCO, and DISCO-PPCO are shown to achieve better utility–privacy trade-off compared with the original implementation. In addition, for heavily imbalanced training data, we demonstrate that ADV-PPCO could protect the privacy more robustly than its counterpart ADV-CE, even when it is trained with dataset in which the ratio between different classes can be as large as five hundred times. Furthermore, we derive the theoretical performance of PPCO and show the consistency between empirical and theoretical performance. In future work, we would further investigate a method that can simultaneously improve privacy protection against both reconstruction and attribute-inference attacks.

References

[1] *No, Gmail Isn't Private at All—But You Can Fix That*. Accessed: May 8, 2022. [Online]. Available: https://cybernews.com/privacy/no-gmail-isnt-private-at-all-but-you-can-fix-that/

[2] *Cambridge Analytica: How Did it Turn Clicks Into Votes?* Accessed: May 8, 2022. [Online]. Available: https://www.theguardian.com/news/2018/may/06/cambridge-analytica-howturn-clicks-into-votes-christopher-wylie

[3] *Facebook to Pay $550 Million in One of Largest Privacy Settlements in U.S. History*. Accessed: May 8, 2022. [Online]. Available: https://fortune.com/2020/01/29/facebook-privacy-settlement/

[4] P. Voigt and A. Von dem Bussche, *The EU General Data Protection Regulation (GDPR) A Practical Guide*, 1st ed. Cham, Switzerland: Springer, 2017.

[5] M. Ribeiro, K. Grolinger, and M. A. M. Capretz, "MLaaS: Machine learning as a service," in *Proc. IEEE 14th Int. Conf. Mach. Learn. Appl. (ICMLA)*, Dec. 2015, pp. 896–902.

[6] M. Al-Rubaie and J. M. Chang, "Privacy-preserving machine learning: Threats and solutions," *IEEE Security Privacy*, vol. 17, no. 2, pp. 49–58, Mar./Apr. 2019.

[7] C. Gentry and D. Boneh, *A Fully Homomorphic Encryption Scheme*, vol. 20, no. 9. Stanford, CA, USA: Stanford Univ., 2009.

[8] T. Orekondy, B. Schiele, and M. Fritz, "Towards a visual privacy advisor: Understanding and predicting privacy risks in images," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 3686–3695.

[9] M. Ryoo, B. Rothrock, C. Fleming, and H. J. Yang, "Privacy-preserving human activity recognition from extreme low resolution," in *Proc. AAAI Conf. Artif. Intell.*, Feb. 2017, vol. 31, no. 1, pp. 1–8.

[10] C. Song and V. Shmatikov, "Overlearning reveals sensitive attributes," in *Proc. 8th Int. Conf. Learn. Represent.*, 2020, pp. 1–12.

[11] F. Pittaluga, S. J. Koppal, S. B. Kang, and S. N. Sinha, "Revealing scenes by inverting structure from motion reconstructions," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 145–154.

[12] J.-W. Chen, L.-J. Chen, C.-M. Yu, and C.-S. Lu, "Perceptual indistinguishability-net (PI-Net): Facial image obfuscation with manipulable semantics," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 6478–6487.

[13] A. Singh et al., "DISCO: Dynamic and invariant sensitive channel obfuscation for deep neural networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 12125–12135.

[14] A. Li, J. Guo, H. Yang, F. D. Salim, and Y. Chen, "DeepObfuscator: Obfuscating intermediate representations with privacy-preserving adversarial learning on smartphones," in *Proc. Int. Conf. Internet-Things Design Implement.*, New York, NY, USA, May 2021, pp. 28–39, doi: 10.1145/3450268.3453519.

[15] Z. Wu, Z. Wang, Z. Wang, and H. Jin, "Towards privacy-preserving visual recognition via adversarial training: A pilot study," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 606–624.

[16] C. Song and A. Raghunathan, "Information leakage in embedding models," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Oct. 2020, pp. 377–390.

[17] H. Zhao, J. Chi, Y. Tian, and G. J. Gordon, "Trade-offs and guarantees of adversarial representation learning for information obfuscation," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 9485–9496.

[18] A. Tripathy, Y. Wang, and P. Ishwar, "Privacy-preserving adversarial networks," in *Proc. 57th Annu. Allerton Conf. Commun., Control, Comput. (Allerton)*, Sep. 2019, pp. 495–505.

[19] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 214–223.

[20] A. Mahendran and A. Vedaldi, "Understanding deep image representations by inverting them," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 5188–5196.

[21] H. Kato and T. Harada, "Image reconstruction from bag-of-visual-words," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 955–962.

[22] A. Dosovitskiy and T. Brox, "Inverting visual representations with convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 4829–4837.

[23] I. Goodfellow et al., "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 2014, pp. 2672–2680.

[24] F. Pittaluga, S. Koppal, and A. Chakrabarti, "Learning privacy preserving encodings through adversarial training," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Jan. 2019, pp. 791–799.

[25] B. M. L. Srivastava, A. Bellet, M. Tommasi, and E. Vincent, "Privacy-preserving adversarial representation learning in ASR: Reality or illusion?" in *Proc. Interspeech*, Sep. 2019, pp. 3700–3704.

[26] M. Coavoux, S. Narayan, and S. B. Cohen, "Privacy-preserving neural representations of text," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2018, pp. 1–10.

[27] B.-W. Tseng and P.-Y. Wu, "Compressive privacy generative adversarial network," *IEEE Trans. Inf. Forens. Security*, vol. 15, pp. 2499–2513, 2020.

[28] M. Arjovsky and L. Bottou, "Towards principled methods for training generative adversarial networks," in *Proc. Int. Conf. Learn. Represent.*, 2017, pp. 1–15.

[29] P. Mohassel and Y. Zhang, "SecureML: A system for scalable privacy-preserving machine learning," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2017, pp. 19–38.

[30] N. Agrawal, A. S. Shamsabadi, M. J. Kusner, and A. Gascón, "QUOTIENT: Two-party secure neural network training and prediction," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Nov. 2019, pp. 1231–1247.

[31] P. Mishra, R. Lehmkuhl, A. Srinivasan, W. Zheng, and R. A. Popa, "Delphi: A cryptographic inference system for neural networks," in *Proc. Workshop Privacy-Preserving Mach. Learn. Pract.*, Nov. 2020, pp. 2505–2522.

[32] D. B. F. Agakov, "The IM algorithm: A variational approach to information maximization," in *Proc. Adv. Neural Inf. Process. Syst.*, 2004, vol. 16, no. 320, p. 201.

[33] L. Yitong, M. Michael, G. Dawson, and D. E. Carlson, "Extracting relationships by multi-domain matching," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 6798–6809.

[34] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral normalization for generative adversarial networks," in *Proc. Int. Conf. Learn. Represent.*, 2018, pp. 1–13.

[35] Z. Zhang, Y. Song, and H. Qi, "Age progression/regression by conditional adversarial autoencoder," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 5810–5818.

[36] I. Ilanchezian, P. Vepakomma, A. Singh, O. Gupta, G. N. S. Prasanna, and R. Raskar, "Maximal adversarial perturbations for obfuscation: Hiding certain attributes while preserving rest," 2019, *arXiv:1909.12734*.

[37] K. Karkkainen and J. Joo, "FairFace: Face attribute dataset for balanced race, gender, and age for bias measurement and mitigation," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Jan. 2021, pp. 1548–1558.

[38] *Scikit-Learn: Machine Learning in Python*. Accessed: May 8, 2022. [Online]. Available: https://scikit-learn.org/stable/

[39] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

Tsung-Hsien Lin was born in Taichung, Taiwan, in 1994. He received the M.S. degree from the Graduate Institute of Communication Engineering, National Taiwan University, Taipei, Taiwan, in 2021. His research interests include machine learning, image processing, and computer vision.

Ying-Shuo Lee is currently pursuing the master's degree with the Graduate Institute of Electrical Engineering, National Taiwan University, under the supervision of Prof. Pei-Yuan Wu. His research interests include artificial intelligence, privacy-preserving machine learning, and face obfuscation.

Fu-Chieh Chang received the M.S. degree from the Graduate Institute of Electronics Engineering, National Taiwan University, Taipei, Taiwan, in 2014, where he is currently pursuing the Ph.D. degree with the Graduate Institute of Communication Engineering. In August 2021, he joined MediaTek Research Laboratory, Taipei, as a Research Scientist. His research interests include reinforcement learning and electronic design automation.

J. Morris Chang (Senior Member, IEEE) received the Ph.D. degree from North Carolina State University, Raleigh, NC, USA. He is currently a Professor with the Department of Electrical Engineering, University of South Florida, Tampa, FL, USA. His past industrial experiences include positions with the Texas Instruments, Microelectronic Center of North Carolina, and AT&T Bell Labs. His research interests include cyber security, data privacy, machine learning, and mobile computing. He was a recipient of the University Excellence in Teaching Award at the Illinois Institute of Technology in 1999. He was inducted into the NC State University ECE Alumni Hall of Fame in 2019. He is a Handling Editor of the journal of *Microprocessors and Microsystems* and an Editor of the IEEE IT PROFESSIONAL.

Pei-Yuan Wu was born in Taipei, Taiwan, in 1987. He received the B.S.E. degree from NTUEE in 2009 and the M.A. and Ph.D. degrees in electrical engineering from Princeton University, in 2012 and 2015, respectively. He has been an Associate Professor at NTUEE, since 2017. He has joined TSMC from 2015 to 2017. His research interests include artificial intelligence, signal processing, estimation and prediction, and cyber-physical system modeling. He was a recipient of the Gordon Y. S. Wu Fellowship in 2010, the Outstanding Teaching Assistant Award at Princeton University in 2012, and the 2020 FutureTech Breakthrough Award by MOST.