Privacy Adversarial Network: Representation Learning for Mobile Data Privacy

SICONG LIU, Xidian University, China
JUNZHAO DU*, Xidian University, China
ANSHUMALI SHRIVASTAVA, Rice University, USA
LIN ZHONG, Rice University, USA

The remarkable success of machine learning has fostered a growing number of cloud-based intelligent services for mobile users. Such a service requires a user to send data, *e.g.* image, voice and video, to the provider, which presents a serious challenge to user privacy. To address this, prior works either obfuscate the data, *e.g.* add noise and remove identity information, or send representations extracted from the data, *e.g.* anonymized features. They struggle to balance between the service utility and data privacy because obfuscated data reduces utility and extracted representation may still reveal sensitive information.

This work departs from prior works in methodology: we leverage adversarial learning to better balance between privacy and utility. We design a *representation encoder* that generates the feature representations to optimize against the privacy disclosure risk of sensitive information (a measure of privacy) by the *privacy adversaries*, and concurrently optimize with the task inference accuracy (a measure of utility) by the *utility discriminator*. The result is the privacy adversarial network (PAN), a novel deep model with the new training algorithm, that can automatically learn representations from the raw data. And the trained encoder can be deployed on the user side to generate representations that satisfy the task-defined utility requirements and the user-specified/agnostic privacy budgets.

Intuitively, PAN adversarially forces the extracted representations to only convey information required by the target task. Surprisingly, this constitutes an implicit regularization that actually improves task accuracy. As a result, PAN achieves better utility and better privacy at the same time! We report extensive experiments on six popular datasets, and demonstrate the superiority of PAN compared with alternative methods reported in prior work.

CCS Concepts: • **Human-centered computing → Ubiquitous and mobile computing systems and tools**; • **Security and privacy** → *Usability in security and privacy*.

ACM Reference Format:
Sicong Liu, Junzhao Du, Anshumali Shrivastava, and Lin Zhong. 2019. Privacy Adversarial Network: Representation Learning for Mobile Data Privacy. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 3, 4, Article 144 (December 2019), 18 pages. https://doi.org/10.1145/3369816

1 INTRODUCTION

Machine learning has benefited numerous mobile services, such as speech-based assistant (*e.g.* Siri), reading log enabled book recommendation (*e.g.* Youboox). Many such services submit user data, *e.g.* sound, image, and human

*Corresponding Author: Junzhao Du

Authors' addresses: Sicong Liu, Xidian University, School of Computer Science and Technology, Xi'an, China; Junzhao Du, Xidian University, School of Computer Science and Technology, Xi'an, China; Anshumali Shrivastava, Rice University, Department of Computer Science, Houston, TX, USA; Lin Zhong, Rice University, Department of Electrical & Computer Engineering, Houston, TX, USA.

2474-9567/2019/12-ART144 $15.00
https://doi.org/10.1145/3369816

activity records, to the *service provider*, posing well-known privacy risks [1, 2, 9]. Our goal is to avoid disclosing raw data to service providers by creating a device-local intermediate component that encodes the raw data and only sends the encoded data to the service provider. And the encoded data must be both *useful* and *private*. For inference-based services, *utility* can be quantified by the inference accuracy, achieved by the service provider using a discriminative model. And *Privacy* can be quantified by the disclosure risk of private information.

Existing solutions addressing the privacy concern struggle to balance between above two seemingly conflicting objectives: privacy vs. utility. An obvious and widely practiced solution is to transform the raw data into task-specific features and upload features only, like Google Now [17] and Google Cloud [16]; This not only reduces the data utility but also is vulnerable to reverse models that reconstruct the raw data from extracted features [28]. The authors of [33] additionally apply dimensionality reduction, Siamese fine-tuning, and noise injection to the features before sending them to the service provider. This unfortunately result in further loss in utility.

Unlike previous work, we employ deep models and adversarial training to automatically learn features for a sweet tradeoff between *privacy* and *utility*. Our key idea is to judiciously combine the discriminative learning, for minimizing the task-specific discriminative error as well as maximizing the user-specified privacy discriminative error, and the generative learning, for maximizing the agnostic privacy reconstruction error. Specifically, we present the Privacy Adversarial Network (PAN), an end-to-end deep model, and its training algorithm. PAN controls three types of descent gradients, *i.e.*, utility discriminative error, privacy discriminative error, and privacy reconstruction error, in back propagation to guide the training of a feature extractor.

As shown in Fig. 2, a PAN consists of four parts: a feature extractor (Encoder $E(\cdot)$), a utility discriminator (UD), an adversarial privacy reconstructor (PR), and an adversarial privacy discriminator (PD). The output of the Encoder (E) feeds to the input of the utility discriminator (UD), privacy reconstructor (PR), and privacy discriminator (PD). We envision the Encoder (E) runs in mobile devices to extract features from raw data. The utility discriminator (UD) represents the inference service to ensure the utility of extracted features. PAN emulates two types of adversarials to ensure the privacy: the privacy discriminator (PD) emulates a malicious party that seeks to extract private information, *e.g.* user location; the privacy reconstructor (PR) emulates one that seeks to reconstruct raw data from the features. We present a novel algorithm to explicitly train PAN via an adversarial process that alternates between *i.e.*, training the Encoder with the utility discriminator (UD) to improve the utility and confronting the Encoder with the adversaries of privacy discriminator (PD) and privacy reconstructor (PR) to enhance the privacy. All four parts iteratively evolve with others during the training phase. Understood from the perspective of manifold, the separate flows of gradients from utility discriminator (UD), privacy discriminator (PD), and privacy reconstructor (PR) through the Encoder in back-propagation can iteratively produces the feature manifold that is both useful and private.

Using digit recognition (MNIST [25]), image classification (CIFAR-10[23] and ImageNet [6]), sound sensing (Ubisound [37]), human activity recognition (Har [40]), and driver behavior prediction (StateFarm [21]), we show PAN is effective in training the Encoder to generate deep features that provide better privacy-utility tradeoff than other privacy preserving methods. Surprisingly, we observe that the adversarially learned features to remove redundant information, for privacy, even surpass the recognition accuracy of discriminatively learned features. That is, removing task-irrelevant information for privacy actually improves generalization and as a result, utility.

In the rest of the paper, we formulate the problem of utility-privacy tradefoff in § 2 and present the PAN's design and its training algorithm in § 3. We report an evaluation of PAN in § 4. We attempt a theoretic interpretation of PAN in § 5, review the related work in § 6, and conclude in § 7.

2 PROBLEM DEFINITION OF MOBILE DATA PRIVACY PRESERVING

This section mathematically formulates the problem of utility-privacy tradeoff for mobile data. Many appealing cloud-based services exist today that require data from mobile users. For example, as shown in Fig 1, a user takes

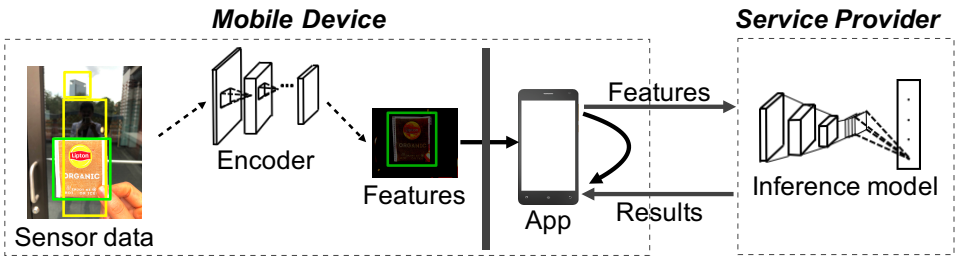


Fig. 1. Privacy preservation in cloud-based mobile services. Mobile users leverage a learned Encoder to locally generate deep features from the raw data (*i.e.*, "tea bag" picture) and give them to the App. The App may send the features to its cloud-based backend.

a picture of a product and sends it to a cloud-based service to find out how to purchase it, a service Amazon actually provides. The picture, on the other hand, can accidentally contain sensitive information, e.g., face and other identifying objects in the background. Therefore, the user faces a touch challenge: how to obtain the service without trusting the service provider with the sensitive information?

Toward addressing this challenge, our key insight is that most services actually do not need the raw data. The user can encode raw data I into representation $E(I)$ through an Encoder $E(\cdot)$ on the mobile device and only sends $E(I)$ to the service provider. The representation $E(I)$ ideally should have the following two properties:

- **Utility:** it must contain enough task-relevant information to be useful for the intended service, e.g., high accuracy for object recognition;
- **Privacy:** it must have little task-irrelevant information, especially that is considered sensitive by the user.

In this work, we focus on classification-based services. Therefore, the *utility* of $E(I)$ is measured by the task inference error C_u (*e.g.* cross entropy) in the service provider. And we quantify the *privacy* of $E(I)$ by the privacy leak risk C_p of raw data in all possible attacking models X. Since the Encoder is distributed to mobile users, we assume it is available to both service providers and potential attackers. That is, both the service provider and the malicious party can train their models using raw data I and their corresponding Encoder output $E(I)$. As such we can restate the desirable properties for the Encoder output $E(I)$ within dataset $\mathbf{T}$ as below:

$$\begin{aligned} \textbf{Utility}: &\quad \underset{E}{Min}\ C_u(E(I_i)), i \in \mathbf{T} \\ \textbf{Privacy}: &\quad \underset{E}{Min}\ \underset{X}{Max}\ C_p(E(I_i)), i \in \mathbf{T} \end{aligned} \tag{1}$$

The first objective (**Utility**) is well-understood for discriminative learning, and achievable via a standard optimization process on the Encoder (E) and the corresponding specialist discriminative model, *i.e.*, minimizing the cross entropy between the predicted task label and the ground truth in a supervised manner [24].

The second objective (**Privacy**) has two parts. The inner part, $\underset{X}{Max}\ C_p(E(I))$, is opposite to the the outer part $\underset{E}{Min}\ C_p(E(I))$. Therefore, the Encoder (E) employed by the mobile user and the specialist attacker (X) used by the malicious party is adversarial to each other in their optimization objectives. Given the information loss in $E(I)$ for privacy, utility loss appears to be certain in theory. One would only hope to find a good, ideally Pareto-optimal, tradeoff between privacy and utility in devising $E(\cdot)$. However, as we will show later, $E(\cdot)$ discovered via PAN actually improves privacy and utility at the same time, a result that can be explained by the practical limits of deep learning in §5.

Important to the quantification of privacy, one must enumerate the privacy leak risks C_p by all possible attackers X in theory. Moreover, the measurement of the privacy leak risk C_p is an open problem in itself [30]. Therefore, we approximate privacy with two specific attackers, each with its own measurement of privacy, elaborated below.

(1) *Specified privacy quantification* in which the user specifies what inference tasks should be forbidden and privacy can be quantified by the accuracy of these tasks. For example, users may want to prevent a malicious party from inferring their identity. In this case, the privacy can be measured by the inaccuracy of identify inference. In this case, the privacy leak risk C_{p1} can be defined as the inference accuracy by a discrimination model employed by the attacker.
(2) *Intuitive privacy quantification* in which the privacy leakage risk C_{p2} is agnostic of the inference tasks under taken by the attacker. In this work, we quantify this agnostic privacy by the difference between the raw data, I, and I', data reconstructed by a malicious party from the Encoder output $E(I)$. We choose this reconstruction error as the agnostic measure for two reasons. First, the raw data in theory contains all information and difference between I and I' is computationally straightforward and intuitive. Second, prior works have already shown that it is possible to reconstruct the raw data from feature representations optimized for accuracy [28, 35, 43].

3 DESIGN OF PAN

To find a good, hopefully Pareto-optimal tradeoff between utility and privacy, we design PAN to learn an Encoder $E(\cdot)$ via a careful combination of discriminative, generative, and adversarial training. As we will show in §4, to our surprise, the resulting Encoder actually improves utility and privacy at the same time.

3.1 Architecture of PAN

As shown in Fig 2, PAN employs two additional neural network modules, utility discriminator (UD) and privacy attacker, to quantify utility and privacy, respectively, in training the Encoder $E(\cdot)$. The utility discriminator simulates the intended classification service; when PAN is trained by the service provider, the utility discriminator can be the same discriminative model used by the service. The privacy attacker, *i.e.*, the intuitive privacy reconstructor (PR) and the specified privacy discriminator (PD), simulates a malicious attacker that attempts to obtain sensitive information from the encoded features $E(I)$. These modules are end-to-end trained to learn the Encoder $E(\cdot)$ for users to extract deep features $E(I)$ from raw data I. The training is an iterative process that we will elaborate in §3.2. Below we first introduce PAN's neural network architecture, along with some empirically gained design insights.

- The **Encoder** $E(\cdot)$ consists of an input layer, multiple convolutional layers, pooling layers, and batch-normalization layers. The convolution layer applies a convolution operation to output activation map with a set of trainable filters. We note that the clever usage of pooling layers and batch-normalization layers contribute to the deep feature's utility and privacy. The batch-normalization layer normalizes the outputted activation map of a previous layer by subtracting the batch mean and dividing by the batch standard deviation [20]. It helps the features' utility because it normalizes the activation to avoid being too high or too low thus has a regularization effect [20]. It contributes to features' privacy as well since it makes it harder for an attacker to recover sensitive information from normalized features. And then, the pooling layer adopts a maximum or average value from a sub-region of the previous layer to form more compact features, which reduces the computational error and avoids over-fitting [12]. It helps privacy because none of un-pooling techniques can recover fine details from the resulting features through shifting small parts to precisely arranging them into a larger meaningful structure [31].

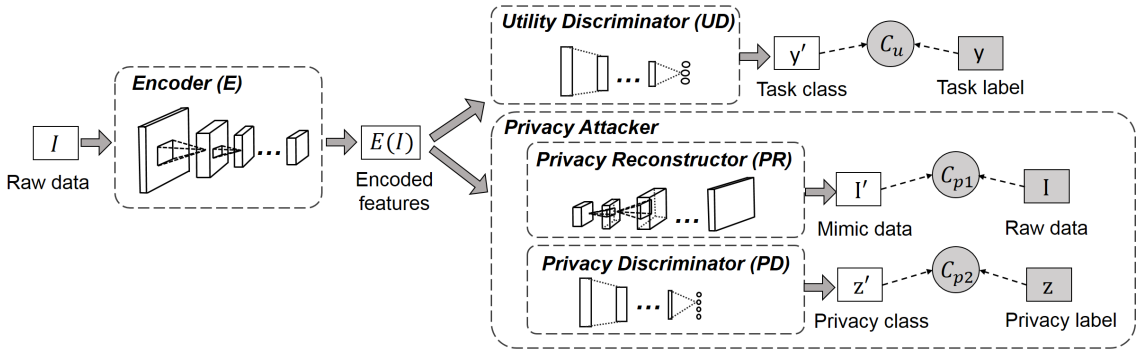


Fig. 2. Architecture of the privacy adversarial network (PAN). The error C_u, C_{p1}, and C_{p2}, which respectively refer to the utility quantification, the specified privacy quantification and the intuitive privacy quantification, build the weight updating criterion during back-propagation.

- The **Utility Discriminator (UD)** builds a multi-layer perceptron (MLP) to process deep features $E(I)$ and output the task classification results y' with several full-connected layers [24]. We also note that a service provider can explore any classification architectures for its utility discriminator model, given the Encoder or its binary version. We choose the MLP architecture because some of the most successful CNN architectures, *e.g.* VGG and AlexNet, can be viewed as the Encoder plus an MLP. The standard cross entropy between the utility discriminator's prediction output y' and the task ground truth y measures the utility error C_u.
- The **Privacy Attacker** employs the two privacy attacking models presented at the end of §2. Specifically, the privacy discriminator (PD) evaluates the recognition accuracy of private class from encoded features $E(I)$. And the privacy reconstructor (PR) quantifies the intuitive reconstruction error between mimic data I' and raw data I.
  - **Specified Privacy Discriminator (PD)** employs a similar MLP classifier as the utility discriminator (UD) to predict the user-specified privacy class z', *e.g.* personal identity, from features $E(I)$. The difference is that the multi-layer PD maps to the corresponding private classes. As noted before, the architecture and training algorithm of PAN can easily incorporate other architectures as the privacy discriminator (PD). The error between the predicted private class z' and the privacy label z measures the specified privacy leak risk C_{p2}.
  - **Intuitive Privacy Reconstructor (PR)** is a usual Encoder turned upside down, composed of multiple un-pooling layers and deconvolutional layers. The un-pooling operation is realized by feature resizing or nearest-value padding [28]. And then the Deconvolution layer densifies the sparse activation obtained by un-pooling through reverse convolution operations [42]. The PR simulates a malicious party and quantifies the intuitive privacy error C_{p1}. After obtaining a (binary) version of the Encoder, a malicious party is free to explore any neural architectures to reconstruct the raw data. In this work, we examine multiple reconstructor architectures and select the one with the lowest reconstruction error as the specialist privacy reconstructor. And we also include an exactly layer-to-layer reversed architecture to mirror the Encoder, to emulate a powerful adversarial reconstructor that knows the internals of the Encoder throughout training. The reconstruction error, *e.g.* Euclidean distance, between I and I' measures the disclosure risk C_{p1} of agnostic privacy information.

Algorithm 1: Mini-batch stochastic training of privacy adversarial network (PAN)

Input: Dataset **T**
Output: PAN's Weights $\{\theta_E, \theta_{UD}, \theta_{PD}, \theta_{PR}\}$
Initialize $\theta_E, \theta_{UD}, \theta_{PD}, \theta_{PR}$;
for *n epochs* **do**
  Sample mini-batch I of m samples from **T**;
  for *k steps* **do**
    Update θ_e and θ_{UD} by gradient ascent with learning rate l_1: minimize C_u ;
    Update θ_{PD} by gradient ascent with learning rate l_2: minimize C_{p1} ;
    Update θ_{PR} by gradient ascent with learning rate l_3: minimize C_{p2} ;
  end
  Update θ_E and θ_{UD} by gradient ascent with learning rate l_4: minimize C_{sum} ;
end
*Note: n and k are two hyper-parameters to synchronize the training of E, UD, PD, and PR parts.

3.2 Training Algorithm of PAN

Our goal with PAN is to train an Encoder that can produce output that is both useful, *i.e.*, leading to high inference accuracy when used for classification tasks, and private, *i.e.*, leading to low privacy inference accuracy and high reconstructive error when maliciously processed and reversely engineered by the attacker, respectively. As we noted in §2, the utility and privacy objectives can be competing when taken naively. The key idea of the PAN's training algorithm is to train the Encoder along with the utility discriminator and the two types of privacy attackers, which specialize in discrimination and reconstruction, respectively. Given a training dataset **T** of m pairs of I, the raw data, y, the true task label, and z, the privacy label, we train a PAN through an iterative process with the following four stages:

(1) Discriminative training mainly maximizes the accuracy to train a specialist utility discriminator (UD); mathematically, it minimizes the cross entropy τ between predicted class $UD(E(I_i))$ and true label y_i:

$$Min\ C_u = \sum_{i=1}^{m} \tau(y_i, UD(E(I_i))). \tag{2}$$

(2) Discriminative training minimizes the cross entropy τ between predicted private class $PD(E(I_i))$ and private ground truth z_i, to primarily train a specialist privacy discriminator (PD):

$$Min\ C_{p1} = \sum_{i=1}^{m} \tau(z_i, PD(E(I_i))) \tag{3}$$

(3) Generative training minimizes the reconstructive error to train a specialist privacy reconstructor (PR):

$$Min\ C_{p2} = \sum_{i=1}^{m} |I_i - PR(E(I_i))|^2 \tag{4}$$

(4) Adversarial training minimizes the sum error to find a *privacy-utility tradeoff*. Specifically, it trains the Encoder to suppress utility error C_u and increase privacy error (C_{p1}, C_{p2}):

$$Min\ C_{sum} = \sum_{i=1}^{m} \lambda_1 \tau(y_i, UD(E(I_i))) - \lambda_2 |I_i - PR(E(I_i))|^2 - \lambda_3 \tau(z_i, PD(E(I_i))) \tag{5}$$

Table 1. Summary of the mobile applications and corresponding datasets for evaluating PAN.

No.	Target task (utility label)	Private attribute (privacy label)	Dataset	Description
T_1	Digit (10 classes)	None	MNIST [25]	70, 000 images
T_2	Image (10 classes)	None	CIFAR-10 [23]	60, 000 images
T_3	Image (5 classes)	None	ImageNet [6]	65, 000 images
T_4	Acoustic event (9 classes)	None	UbiSound [37]	7, 500 audio clips
T_5	Human activity (7 classes)	Human identity (33 classes)	Har [40]	10, 000 records of accelerometer and gyroscope
T_6	Driver behavior (10 classes)	Driver idenity (26 classes)	StateFarm [21]	22, 424 images

C_{sum} is a Lagrangian function of C_u, C_{p_1} and C_{p2}. λ_1, λ_2, and λ_3 are Lagrange multipliers that can be used to control the relative importance of privacy and utility. When we set $\lambda_2 = 0$ or $\lambda_3 = 0$, PAN only trains the Encoder to resist against the specified privacy discriminator or the intuitive privacy reconstructor, respectively.

Algorithm 1 summarizes the training algorithm of PAN. We leverage mini-batch techniques to split the training data into small batches, over which we calculate the average of the gradient to reduce the variance of gradients, which balance the training robustness and efficiency (line 3) [26]. Within each epoch, we first perform the standard discriminative and generative stages (line 5, 6, 7) to initialize the Encoder's weights θ_E and train the specialist utility discriminator (UD), privacy discriminator (PD) and privacy reconstructor (PR). And then, we perform the adversarial stage (line 9) to shift the utility-privacy tradeoff on the Encoder weight θ_E tuning. We note that k in line 4 is a hyper-parameter of the first three stages. These k steps followed by a single iteration of the forth stage is trying to synchronize the convergence speed of these four training stages well, borrowing existing techniques in generative adversarial network [13]. Our implementation uses an empirically optimized value of $k = 3$. And we leverage the AdamOptimizer [22] with an adaptive learning rate for all four stages (line 5, 6, 7 and 9).

4 EVALUATION

In this section, we evaluate PAN's performance using six classification services for mobile apps, with a focus on the utility-privacy tradeoff. We compare PAN against alternative methods reported in the literature and visualize the results for insight into why PAN excels.

4.1 Experiment Setup

Evaluation applications & datasets. We evaluate PAN, especially the resulting Encoder, with six commonly used mobile applications/services, for which the corresponding benchmark datasets are summarized in Table 1. Specifically, the target task in T_1 (MNIST [25]) is handwritten digit recognition. The agnostic private information in the real-world raw image may include individual handwritten style and the background paper. We use 50, 000 images for PAN training and 20, 000 images for validation and testing. The target tasks in T_2 (CIFAR-10 [23]) and T_3 (ImageNet [6]) are image classification. The agnostic private information in the real-world raw image may involve background location, color, and brand. We choose 40, 000 images for training and remaining images for testing in both cases. The target task in T_4 (UbiSound [37]) is to recognize acoustic event. The agnostic private information covers background voice and environment information. We use 6, 000 audio clips for training and 1, 500 audio clips for testing. The target task in T_5 (Har [40]) is human activity identification based on the records of accelerometer and gyroscope. The specified private attribute we intend to hide is useridentity. And the agnostic private information we expect to protect may contain individual habit. We randomly select 8, 000 records for training and 2, 000 records for testing. The target task in T_6 (StateFarm [21]) is to predict driver behavior. The specified private attribute we choose is driver identity. And the agnostic private information within the real-world raw image can be face and gender. We use 18, 000 images for training and 4, 424 images for testing.

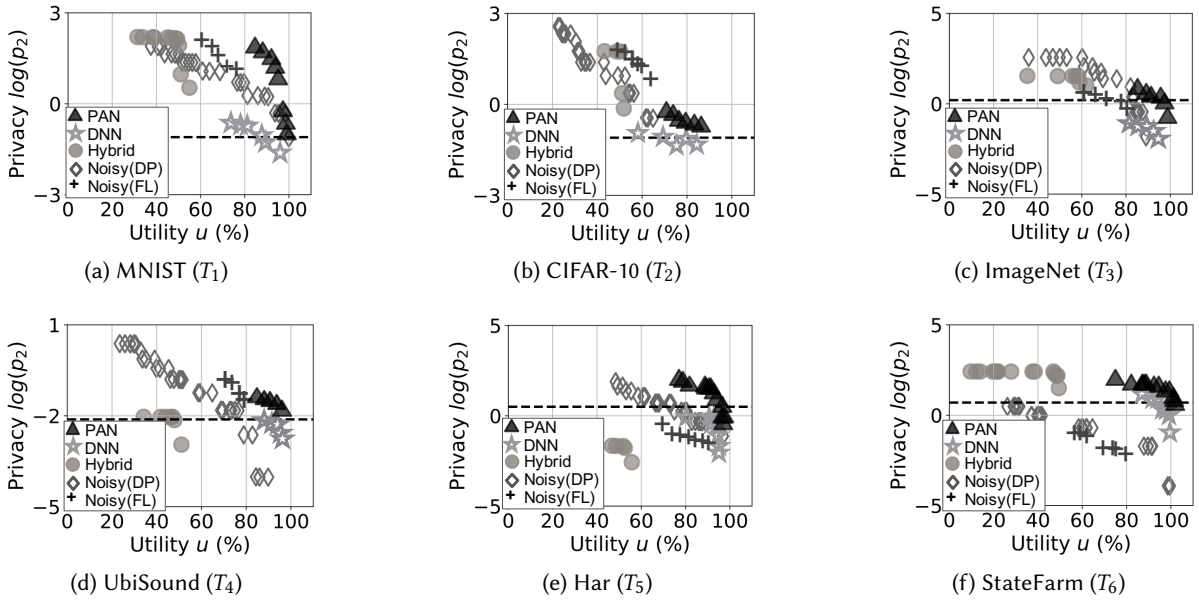
Evaluation models. In PAN, we leverage a utility discriminator (UD), a privacy discriminator (PD), and a privacy reconstructor (PR) model to train and validate the Encoder (E). In the training phase, we refer to the successful neural network architectures to build PAN's Encoder (E), Utility Discriminator (UD) and Privacy Discriminator (PD) for different types of datasets. For example, according to the sample shape in the datasets, the LeNet is chosen as the reference for T_1, T_4 and T_5, AlexNet is for T_2 and T_6, and VGG-16 model is for T_3. To evaluate the learned Encoder in the testing phase, we leverage another set of separately trained Utility Discriminator (UD) and Privacy Attackers (PD and PR), given PAN's Encoder output, to simulate the service provider and malicious parties. In particular, we ensemble multiple optional MLP architectures to simulate the service provider's Utility Discriminator (UD) for task recognition, as well as the malicious attacker's Privacy Discriminator (PD) for private attribute prediction. These MLP models have different fully-connected architectures by using varying scales of singular value decomposition, sparse-coding factorization, and global-average computation to replace the initial fully-connected layers. We also employ multiple generative architectures to select the most powerful one as the privacy reconstruction attacker (PR). To emulate a powerful adversary that knows the Encoder for the attackers' training, we include a PR model that exactly mirrors the Encoder for each task.

Prototype implementation. PAN has two phases: an offline phase to train the Encoder, and an online phase where we deploy the learned Encoder as a middleware on mobile platforms to encode the raw sensor data into features. In the offline phase, we use the Python library of TensorFlow [38] to train the Encoder, utility discriminator, privacy discriminator as well as privacy reconstructor using the datasets summarized in Table 1. And we leverage h5py [5] to separately save the trained models. To speedup the training, we leverage a server with four Geforce GTX 1080 Ti GPUs with CUDA 9.0. In the online phase, we prototype the mobile-side on the Android platform, *i.e.*, Xiaomi Mi6 smartphone, using TensorFlow Mobile Framework [15]. And we store the learned Encoder in the smartphone's L2-cache using Android's LuCache API [14], which speeds up the on-device data encoding. The Encoder intercepts the incoming testing data and encodes it into features, which are then fed into the corresponding Android Apps for real-word performance evaluation.

4.2 Comparison Baselines

We employ four types of state-of-the-art data privacy preserving baselines to evaluate PAN. The DNN method provides a high utility standard, and the DP, FL, and Hybrid DNN methods set a strict utility-privacy tradeoff benchmark for PAN. The detail settings of the baseline approaches and PAN are as below.

- **Noisy (DP)** method perturbs the raw data I by adding Laplace noise with diverse factors $\{0.1, 0.2, ...0.9\}$, and then submit the noisy data $\overline{I}$ to the service provider. This is a typical local differential privacy (DP) method [7, 18]. The utility u of noisy data is tested by the task (*e.g.* the driver behavior in T_6) recognition accuracy in a MLP classifier UC with multiple fully-connected layers. The specified privacy p_1 is measured by the inference accuracy over private attribute (*e.g.* driver identity in T_6) in another MLP classifier PC. And the intuitive privacy p_2 is evaluated by the average information loss, *i.e.*, $p_2 = avg(|I_i - \overline{I_i}|^2), I_i \in T_{test}$. Here T_{test} is the corresponding testing set within datasets $T_1 \sim T_6$.
- **Noisy (FL)** method perturbs the data I by adding Gaussian noise $N(0, \sigma^2)$ with mean 0 and variance σ^2, where we set $\sigma = 40$ according to [34]. The Gaussian noise included in the noisy data $\overline{I}$ can provide rigorous guarantees of differential privacy using less local noise. This is widely used in the noisy aggregation scheme of federated learning (FL) [34, 39]. We test the utility u, the specified privacy p_1, and the intuitive privacy p_2 of this noisy data $\overline{I}$ using the similar methodology as DP baseline.
- **DNN** method encodes the raw data I into features F using a deep encoder with multiple convolutional and pooling layers, and expose features F to the service provider [16, 17]. The utility u of DNN features is measured by the inference accuracy in a classifier UC with multiple fully-connected layers. The specified privacy p_1 is tested by the inference accuracy over the private attribute in another privacy classifier PC with multiple



(a) MNIST (T_1) (b) CIFAR-10 (T_2) (c) ImageNet (T_3)

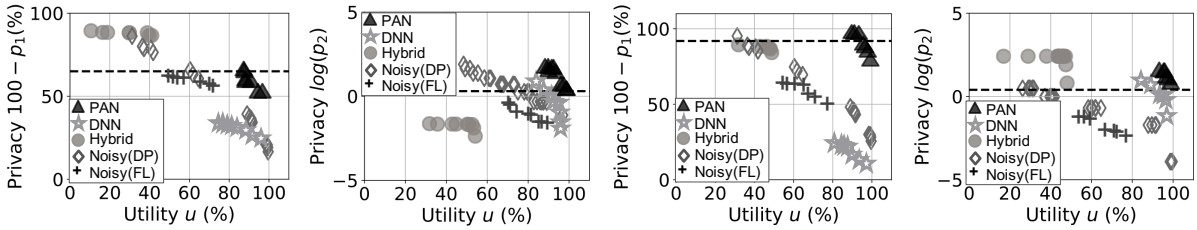
(d) UbiSound (T_4) (e) Har (T_5) (f) StateFarm (T_6)

Fig. 3. Performance comparison of PAN_1 with four baselines across six different applications ($T_1, T_2, T_3, T_4, T_5, T_6$). The X-axis shows the utility u (*i.e.*, task inference accuracy) tested by the simulated service provider. And Y-axis is the intuitive reconstruction privacy p_2 tested by the simulated malicious attacker, normalized by *log* operation. The dashed line sets a privacy $log(p_2)$ benchmark validated by PAN's privacy reconstructor.

fully-connected layers. And the intuitive privacy p_2 is tested by the reconstruction error in a decoder D with multiple deconvolutional and unpooling layers, *i.e.*, $p_2 = |I_i - D(F_i)|^2, I_i \in T_{test}$.

- **Hybrid** DNN method further perturbs the above DNN features through additional lossy processes, *i.e.*, principal components analysis (PCA) and adding Laplace noise [33] with varying noise factors $\{0.1, 0.2, ...0.9\}$, before delivering them to the service provider. The utility u, the specified privacy p_1, and the intuitive privacy p_2 of the perturbed features F' is respectively tested by a task classifier, a private attribute classifier, and a decoder, using the same methodology of the DNN baseline.
- **PAN** automatically transform raw data I into features, *i.e.*, $E(I)$, using the learned Encoder $E(\cdot)$. In particular, we evaluate the following two types of PAN, that are trained to defend against different types of privacy attackers for different benchmark tasks/datasets (Table 1):
  - **PAN_1** is trained with one privacy attacker, *i.e.*, the Privacy Reconstructor (PR), by setting $\lambda_3 = 0$ in the adversarial training objective (Eq.(5)). We train the PAN_1 on six datasets (*i.e.*, $T_1 \sim T_6$).
  - **PAN_2** is trained with two privacy attackers, *i.e.*, Privacy Discriminator (PD) and Privacy Reconstructor (PR), for application datasets accompanied with both utility labels and private attribute labels (*i.e.*, T_5 and T_6).

  The utility u of both PAN_1's and PAN_2's Encoder output are tested by the task inference accuracy in the service provider's utility discriminator (UD) using a classifier. The specified privacy p_1 of PAN_2's Encoder output is evaluated by the inference accuracy in the attacker's privacy discriminator (PD) using a classifier. As for the intuitive reconstruction privacy p_2 in PAN_1 and PAN_2, we select the most powerful decoder as the privacy reconstructor (PR) to evaluate it, *i.e.*, $p_2 = |I_i - PR(E(I_i))|^2, I_i \in T_{test}$.



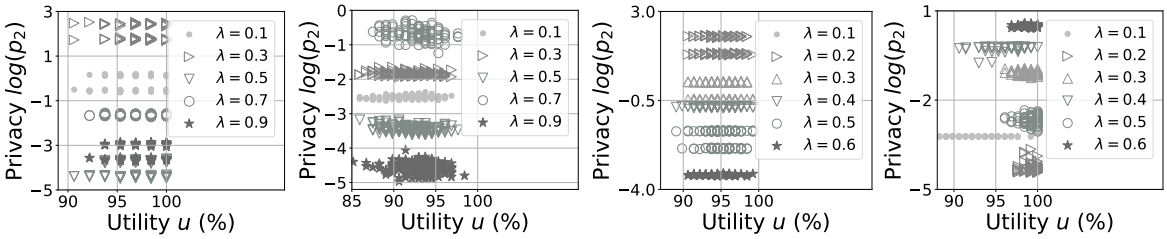
(a) p_1, Har (T_5) (b) p_2, Har (T_5) (c) p_1, StateFarm (T_6) (d) p_2, StateFarm (T_6)

Fig. 4. Comparison of PAN_2 with four baselines on Har (T_5) and StateFarm (T_6). X-axis is the utility u (*i.e.*, task inference accuracy). Y-axis in (a) and (c) represents the specified privacy p_1 (*i.e.*, private attribute inference accuracy) normalized by $100 - p_1(\%)$. And Y-axis in (b) and (d) is the intuitive privacy p_2 normalized by log operation. The dashed line set the privacy benchmark validated by PAN's privacy discriminator in (a) and (c) and privacy reconstructor in (b) and (d).

4.3 Utility vs. Privacy Tradeoffs

This subsection evaluates PAN in terms of the utility u by the service provider and the privacy p_1 and p_2 by the malicious attackers, compared with four privacy-preserving baselines (see § 4.2). Figure 3 and Figure 4 summarize the Pareto fronts of the testing privacy-utility tradeoffs by four baselines and PAN. In this set of experiments, we train the PAN_1 on the six application datasets ($T_1 \sim T_6$) based on utility labels, and train the PAN_2 on Har (T_5) and StateFarm (T_6) datasets accompanied with both utility labels and private attribute labels (see Table 1).

First, PAN's Encoder output achieves the best privacy-utility tradeoff, compared to those encoded by other four baselines. In Figure 3, we see the performance of PAN_1's Encoder output lies in the upper right corner with maximized utility u and maximized privacy $log(p_2)$ on the digit recognition applications (T_1), and lie around the upper right side with maximized utility u and competitive privacy $log(p_2)$ compared with other four baselines on image classification applications (T_2 and T_3) and audio sensing applications (T_4). In Figure 4, the PAN_2 is also in the upper right corner with maximized utility u and maximized privacy $(100 - p_1)\%$ or $log(p_2)$ on both human activity recognition application (T_5) and driver behavior prediction application (T_6). Here we transform the expected minimized privacy $p_1\%$ to the maximized privacy $(100 - p_1)\%$. When we consider the $\lambda_1 u + \lambda_2(100 - p_1) - \lambda_3 log(p_2)$ as a quantifiable metric of utility-privacy tradeoff, both the PAN_1's and PAN_2's Encoder output achieve the best tradeoff value according to the default relative importance $\lambda_1 = 0.4, \lambda_2 = 0.3$ and $\lambda_3 = 0.3$. While the DNN method provides unacceptable low privacy, and Hybrid DNN, Noisy (DP) and Noisy (FL) methods offer high privacy at the cost of utility degradation. Second, the utility (*i.e.*, task inference accuracy) of PAN's Encoder output is at least as good as and sometimes even better than the other four baseline methods across different applications. Specifically, the task inference accuracy by PAN_1 is $83.1 \sim 99.8\%$ on MNIST (T_1), $70.5 \sim 89.3\%$ on CIFAR-10 (T_2), $81.8 \sim 99.2\%$ on ImageNet (T_3), $83.6 \sim 98.1\%$ on UbiSound (T_4), $78.6 \sim 98.5\%$ on Har (T_5), and $77.8 \sim 98.6\%$ on StateFarm (T_6), maintaining at a high level. And the task inference accuracy by PAN_2 is $85.1 \sim 99.3\%$ on Har (T_5) and $87.3 \sim 99.8\%$ on StateFarm (T_6). It is even better than the utility of standard DNN features. Although, with carefully-calibrated Gaussian noise distribution in Noisy (FL), we observe better utility when using Noisy (FL) method than that using the Noisy (DP) method. The task inference accuracy in Noisy (DP), Noisy (FL) and Hybrid DNN baselines is seriously unstable, ranging from 12.8% to 96.7% on different applications, because of the injected noises. Also, we see the utility of PAN_2 on Har and StateFarm is slightly improved than the PAN_1 case. It implies the PAN_2 with two adversaries learns better features than PAN_1 with only one generative model-based adversary. Third, PAN's Encoder output in both PAN_1 and PAN_2 cases considerably improves the privacy than the DNN method and achieves the competitive privacy compared with other three baselines. Moreover, the PAN's



(a) PAN_1 on MNIST (T_1) (b) PAN_1 on UbiSound (T_4) (c) PAN_2 on Har (T_5) (d) PAN_2 on StateFarm (T_6)

Fig. 5. We can tune the utility-privacy tradeoff in both PAN_1 and PAN_2 by selecting different Lagrange multipliers λ in the adversarial training phase (Eq.(5)).

privacy p_1 and p_2 quantified by PAN's privacy discriminator (PD) and privacy reconstructor (PR) (the dashed lines in Figure 3,4) is comparable with that measured by the third-party attackers (solid black triangles in Figure 3,4). We train the third-party attackers using the binary version of PAN's Encoder. This result demonstrates the strong adversary ability of PAN's privacy discriminator and privacy reconstructor.

Summary. First, although the PAN_1 and PAN_2 cannot always outperform the baseline methods in both utility and privacy, it achieves the best Pareto front for the utility-privacy tradeoffs across various adversaries and applications. Second, the utility, *i.e.*, inference accuracy, of PAN's Encoder output is even better than taht of the standard DNN. We will revisit this surprising result in § 4.6 and § 5.

4.4 Impact of the Lagrangian Multipliers

An important step in PAN's training is to determine the Lagrangian multipliers $\lambda_1, \lambda_2, and \lambda_3$ in the adversarial training stage (see Eq.(5)). We verify that we are able to tune PAN's utility-privacy tradeoff point through setting different λ_1, λ_2, and λ_3 in adversarila training phase (see Eq.(1)), shown in Figure 5. Let $\lambda_3 = 0$ and $\lambda_2 = 1 - \lambda_1$, we show evaluate the influence of Lagrangian multiplier on PAN_1 tradeoff performance over two typical applications (*e.g.* T_1 and T_4), with five discrete choices of $\lambda_1 \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$. As for PAN_2, empirically assuming $\lambda_3 = 0.2$ and $\lambda_2 = 1 - 0.2 - \lambda_1$, we compare six discrete choices of $\lambda_1 \in \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6\}$. And we see that the optimal choice of Lagrange multiplier, among above optional space, is $\lambda_1 = 0.3$ for PAN_1 on digit classification (T_1: MNIST), is $\lambda_1 = 0.7$ for PAN_1 on non-speech sound recognition (T_4: Ubisound), is $\lambda_1 = 0.2$ for PAN_2 on human activity detection (T_5: Har), and $\lambda_1 = 0.6$ for PAN_2 on driver behavior classification (T_6: StateFarm).

Summary. The Lagrange multipliers λ_1, λ_2, and λ_3 bring flexibility to PAN to satisfy different requirements of utility-privacy tradeoffs according to the relative importance between utility and privacy budgets across various tasks/applications. And we note it is exhaustive to search the optimal λ_1, λ_2 and λ_3, since we can always search it from a finer-grained discrete space (*e.g.* $\{0.0001, 0.0002, ..., 0.799\}$). An alternative in the future work is to leverage the automated search technique, *e.g.* deep deterministic policy gradient algorithm, for efficient searching.

4.5 Performance on Smartphone

We next evaluate PAN on a commercial off-the-shelf smartphone with six Android applications.

4.5.1 Resource Cost of PAN's *Encoder on Smartphone.* This subsection evaluates the run-time execution cost (*e.g.* latency, storage, and energy consumption) of the learned PAN's Encoder for encoding different formats of data on the Xiaomi Mi6 smartphone. Specifically, we deploy the learned Encoder on the smartphone to interrupt and encode the incoming testing sample into features (*i.e.*, Encoder output). And then the Encoder output is

Table 2. Resource cost of PAN's Encoder for data encoding across five Android applications on Xiaomi Mi6 smartphone.

Applications	Encoder's Cost on Xiaomi Mi6 Smartphone		
	Latency (ms)	Storage (KB)	Energy (mJ)
Digit recognition (T_1: MNIST)	26	135	0.8
Image classification (T_2: CIFAR-10)	31	198	1.6
Image classification (T_3: ImageNet)	102	310	3.2
Acoustic event recognition (T_4: Ubisound)	42	213	1.8
Human activity prediction (T_5: Har)	27	269	0.9

Table 3. Performance on the driver behavior recognition Android App. Utility u is the driver behavior recognition accuracy, specified privacy p_1 is the driver identity classification accuracy, and intuitive privacy p_2 is the data reconstruction error.

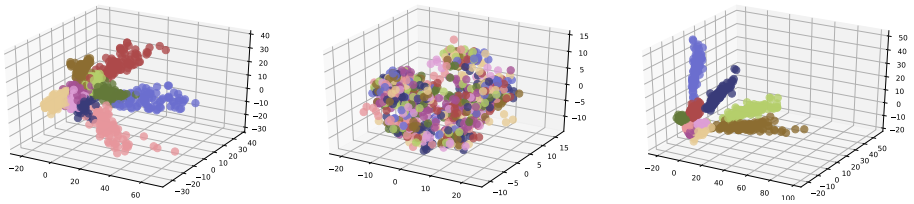
Input to App	Utility u (%)	Specified privacy p_1 (%)	Intuitive privacy $log(p_2)$
Case A: Raw image	98.2	93.6	0
Case B: DNN features	96.1	65.6	0.035
Case C: PAN's Encoder output	99.1	23.1	0.163

fed into the corresponding Android Apps for task recognition and privacy validation. In this experiment, the task classifier is embedded in the corresponding Android App, and the privacy validation models (*i.e.*, private attribute classifier and privacy reconstructor) are executed on the cloud to attack the Encoder output collected by Android APP. We summarize the on-device execution cost of PAN's Encoder to encode five formats of data in Table 2. In particular, we load the PAN's Encoder (parameters and architecture files) in the smartphone cache to speedup processing, since it only occupies $\leq 310KB$ storage. And the multiply-accumulate (MAC) operations of the Encoder network are run using smartphone CPU [37]. PAN's Encoder occupies only $135 \sim 310KB$ of memory, takes $26 \sim 102ms$ of encoding latency, and incurs $0.8 \sim 3.2mJ$ of energy cost for each encoding pass of raw data.

Summary. PAN's Encoder does not incur notable high resource cost. Therefore it is compact to deploy on the resource-constrained mobile platforms as a data preprocessing middleware. In particular, it takes low memory usage since the Encoder only contains convolutional layers, without storage-exhaustive fully-connected layers. The execution delay is only several milliseconds [27]. And the energy cost is less than $\leq 3.2mJ$, which is insignificant compared with Xiaomi Mi6's battery capacity, *i.e.*, $3350mAh$.

4.5.2 Case studies on driver behavior recognition App. The user inputs data to an Android App to recognize driver behavior. Meanwhile, he wants to hide the private attributes (*e.g.* the driver identity) and other agnostic private information (*e.g.* the driver race and car model). Therefore, he leverages PAN to encode the raw image into features and only deliver the Encoder output to the Android App. We artificially play an example trace of the driver behavior during the study with 80 driver images from 10 drivers, selected from $4,424$ testing samples of StateFarm (T_6). We consider 3 cases of the input data to the driver behavior recognition Android App: Case A: raw data, Case B: features generated by a standard DNN; and Case C: PAN's Encoder output. Table 3 shows the evaluation results on the driver behavior recognition App for three cases. In Case A with raw image input, the driver classification accuracy (utility) by App is 98.2%, while the adversary's accuracy of predicting the private attribute, *i.e.*, driver identity, is 93.6%. In Case B of DNN feature input, the utility of classifying driver behavior is 96.1%, and the private driver identity inference accuracy p_1 by the malicious attacker is 65.6%. It indicates that both the raw data and DNN feature cases reveal private attribute-correlated information. As for Case C with PAN's Encoder output, it incurs an improvement (0.9%) in driver behavior recognition accuracy, and reduce the private identity prediction accuracy p_1 by 70.5%. Meanwhile, the intuitive reconstruction privacy p_2 in Case C is 0.163, which is larger than that in Case B (0) and Case C (0.035). The more significant reconstruction error implies the less private information leakage risk.

Summary. This outcome demonstrates PAN's Encoder improves utility with quantified privacy guarantees.



(a) DNN (b) Hybrid DNN (c) PAN

Fig. 6. Visualization of features learned by DNN, Hybrid method, and PAN's Encoder on StateFarm (T_6) datasets. Different color in each figure standards for one task class.

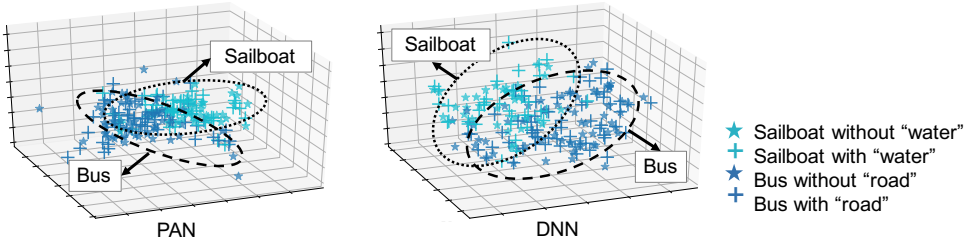


Fig. 7. Detail visualization of feature learning by PAN and DNN on two categories of images from ImageNet (T_3). The target task is to classify the "sailboat" and "bus" image samples. The background "water" in "sailboat" images and the background "road" in "bus" images are redundant (private) to recognize the target "sailboat" and "bus".

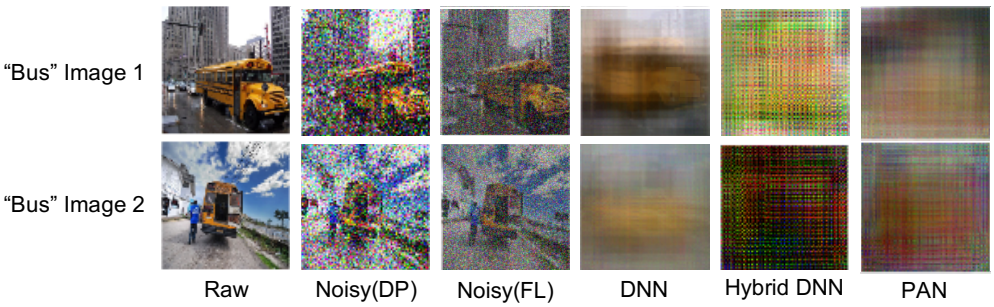


Fig. 8. Visualization of reconstruction privacy. From left to right: raw "bus" images from ImageNet (Raw), images with Laplace noise (DP), images with carefully-calibrated Gaussian noise (FL), and images reconstructed from DNN's features, Hybrid DNN's features, and PAN's Encoder output.

4.6 Visualization of PAN's Encoder Output

In this subsection, we further visualize the PAN's Encoder output in terms of feature distribution and the reconstruction privacy to seek insight into answering the following questions: what is the impact of PAN on the learned features, how does PAN disentangle the feature components relevant to privacy from those relevant to utility, and how well does PAN preserve the reconstruction privacy of raw data?

4.6.1 Visualization of PAN's Encoder Output on Feature Space. Fig. 6 and Fig. 7 visualize how the feature manifold is derived by DNN, DNN(resized), and PAN. First, in Fig. 6, PAN's Encoder output is highly separable as DNN method do on the feature space, which indicates its utility for task recognition. While the manifold driven by the Hybrid baseline with PCA and noise addition processes on DNN features is blurry, this is why the resized features by Hybrid DNN method hurt utility. Moreover, the feature distribution (manifold) formed by PAN is the most constrictive one compared to that from the DNN and Hybrid baselines, which leads to the improved utility. Second, PAN to push the features away from redundant private information, for privacy, makes the manifold more constrictive, so that enhances the utility. Specifically, to zoom in on two categories of images from ImageNet (T_3) for more details about how PAN and DNN form the feature manifold to achieve utility-privacy tradeoff, as shown in Fig. 7. The target task is to classify the two categories, "sailboat" and "bus". The private background information in the "sailboat" raw image is "water", and the private information in the "bus" image is "road". We see PAN pushes features towards the constrictive space dominated by the samples without redundant (private) information, *i.e.*, "sailboat without water" and "bus without road", which guarantees privacy, avoids over-fitting, and improves utility as well. While the DNN method may capture the background (private) information "water" and "road" and retain them in the feature manifold to help the target task classification of "sailboat" and "bus", therefore hurts privacy. We defer the theoretical interpretation of this result to § 5.

4.6.2 Visualization of PAN's *Encoder output on Reconstruction Privacy.* Fig. 8 visualizes the reconstruction privacy of PAN's Encoder output, in comparison to the baseline approaches, using two "bus" image samples from ImageNet. We adopt the same architectures of encoder (*i.e.*, 12 conv layers, 5 pooling layers, and 1 batch-normalization layer) and privacy reconstructor (*i.e.*, the encoder turned upside down) to decode the features generated by DNN, Hybrid DNN, and PAN for fair comparison. We see the images reconstructed from the DNN features convey the target object "bus" information and the private background "road" information, indicating a high risk of private background leakage. Adding noise to the images in DP and FL or adding noise to the features in Hybrid DNN baselines obfuscate both utility-related "bus" information and the privacy-correlated background "road" information, compromising task detection accuracy (utility) at the cost of privacy. The PAN, instead, only muddles the utility-irrelevant private information "road", making background information reconstruction impossible without compromising the utility.

5 MANIFOLD BASED INTERPRETATION

Our evaluation reported above shows that PAN is able to train an Encoder that improves utility and accuracy at the same time. This section attempts to provide a theoretical interpretation of this surprising result.

We resort to the manifold perspective of the deep model. It is common in literature to assume that the high-dimensional raw data lies on a lower dimensional manifold [4]. A DNN can also be viewed as a parametric manifold learner utilizing the nonlinear mapping of multi-layer architectures and connection weights. We decompose the input data into two orthogonal lower dimensional manifolds: $I = I^{OD} + I_{\perp}^{OD}$. Here, the component I^{OD} is the manifold component that is both necessary and sufficient for task recognition (*e.g.* driver behavior). Thus, ideally, we want our training algorithm to rely on this information for task recognition solely. Formally, for the utility discriminator (UD), this implies that $prob(y|I) = prob(y|I^{OD})$. And the other manifold component $I_{\perp}^{OD}$, orthogonal to I^{OD}, may or may not contain information for the objective class, but it is dispensable for task detection. In practice, the real data does have redundant correlations. Thus $I_{\perp}^{OD}$ may be learned for task recognition, but unnecessary. However, revealing $I_{\perp}^{OD}$ is likely to contain some sensitive information (*e.g.* driver identity information and background information) thus hurt the privacy. If we assume that there does exist a sweet-spot tradeoff between utility and privacy, that we hope to find, then it must be the case that I^{OD} is not sensitive.

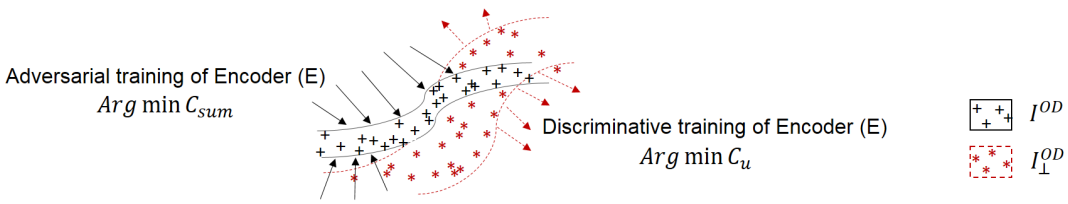


Fig. 9. A new manifold pushed by PAN to form the feature extractor, *i.e.*, Encoder, for better utility and privacy. The utility-specified discriminative learning objective (Eq.(2)) push it to contain I^{OD} and $I_{\perp}^{OD}$, and the privacy-imposed adversarial training objective (Eq.(5)) pushes it away from the sensitive component $I_{\perp}^{OD}$.

The features F learned by standard discriminative learning to minimize the classification error based on information from I, will mostly likely overlap (non-zero projection) with both I^{OD} and $I_{\perp}^{OD}$. And the overlap with $I_{\perp}^{OD}$ compromises the privacy (as evident from our experiments). Meanwhile, the projection of manifold I^{OD} on $I_{\perp}^{OD}$ is significant as it might capture other extra sensitive features, which will help task recognition accuracy. Apart from privacy, the redundant correlation in $I_{\perp}^{OD}$ is also likely only be spurious in training data. Thus, merely minimizing classification loss can lead to over-fitting.

This is where we can skill two birds with one stone via an adversarial process. In PAN, the Encoder $E(\cdot)$ is trained by the utility-specified discriminative learning objective (Eq.(2)) and privacy-imposed adversarial learning objective (Eq.(5)), to remove extra sensitive information in features F' as shown in Fig. 9. The transformed manifold formulated by Encoder $E(\cdot)$ is forced by discriminative learning objective (Eq.(2)), just like the traditional approach, to contain information from both I^{OD} as well as $I_{\perp}^{OD}$. However, the adversarial training objective (Eq.(5)) will push features F' away (or orthogonal) from $I_{\perp}^{OD}$. In this way, we get privacy as well, since F' as a function of I which has two manifolds, being orthogonal to $I_{\perp}^{OD}$ forces it to only depend on I^{OD}.

Meanwhile, from a generalization perspective, in the training data, the spurious information from $I_{\perp}^{OD}$ that might over-fit the training data is iteratively removed by the adversarial training objective (Eq.(5)), leading to enhanced generalization. For example, as shown in Fig. 7, if we want to discriminate between "bus" and "sailboat", the background information "road" in the image can help in most cases but can also mislead when the test image contains a "sailboat" being transported on the "road". Therefore, by considering the background information, a standard DNN may not generalize well. In contrast, because the background may contain sensitive information and contribute to reconstruction error, PAN is likely to train the Encoder to remove information about the background and as a result, improve the task accuracy .

The above interpretation highlights the possibility that utility and privacy are not completely competing objectives in practice. We believe that a rigorous formalism and thorough investigation of this phenomena is necessary to shed more insight and derive better designs.

6 RELATED WORK

Our work is inspired by and closely related to the following works.

Data Privacy Protection in Machine Learning based Services: Randomized noise addition [18] and Differential privacy [1, 8] techniques are widely used by service providers to remove personal identities in the released datasets. They provide strong privacy guarantees but often lead to a significant reduction in utility (as shown in § 4.3). GAP [19] learns a privatization scheme to sanitize the datasets for limiting the risk of inference attacks on personal private attributes. Erdogdu *et al.* design a privacy mapping scheme for continuously released time-series of user data to protect the correlated private information in the dataset [11]. Federated learning [34, 39] techniques prevent inference over the sensitive data exchanged between distributed parties during training by

noisy aggregation of multiple parties' resulting models. However, all of the above techniques are tailored to datasets or the statistics of datasets, which are unsuitable to our problem settings, *i.e.*, run-time data privacy protection in the online inference phase. Meanwhile, applying the statistic information to a new context-aware case is still an open problem.

PAN is a very different approach toward preserving the privacy at run-time. As the raw data is generated, it is intercepted by a trained Encoder and the encoded features are then fed into the untrusted service.

Data Utility-Privacy Tradeoff using Adversarial Networks: Adversarial networks have been explored for data privacy protection, in which two or more players defend against others with conflicting utility/privacy goals. Seong *et al.* introduce an adversarial game to learn the image obfuscation strategy, in which the user and recogniser (attacker) strive for antagonistic goals: dis-/enabling recognition [32]. Wu *et al.* [41] propose an adversarial framework to learn the degradation transformation (*e.g.* anonymized video) of video inputs. This framework optimizes the tradeoff between task performance and privacy budgets. However, both practices only consider protecting privacy against attackers that perform "vision" recognition on specific data format (*e.g.* image), which is insufficient across diverse data modalities in ubiquitous computing. On the contrary, PAN allow users to specify the utility and privacy quantification towards different data formats according to application requirements. And we have evaluated the PAN's usability across image, audio, and motion data formats in § 4. OLYMPUS [36] learns a data obfuscator (*i.e.*, AutoEncoder) to jointly minimize privacy and utility loss, where the privacy and utility requirements are modeled as adversarial networks.

Although the above works share the idea of adversarial learning with ours, they use the generative model to obfuscate the raw data in a homomorphic way. In contrast, PAN uses the Encoder to learn a downsampling transformation, *e.g.*, features, from the raw data, and send the features, rather than any forms of obfuscated/synthetic data, to service providers. A byproduct of this encoding is that PAN has more efficient data communication from the mobile to the service provider.

Deep Feature Learning for Utility or Privacy: Edwards *et al.* propose the adversarial learned representations that are both fair (independent of sensitive attributes) and discriminative for the prediction task [10]. However, they target at fair decision by quantifying the dependence between representation and sensitive variables thus provide no privacy guarantees. Our work is closely related to [3] that employs a variational GAN to learn the representations that hide the personal identity and preserve the facial expression. However, it employs a generative model to minimize the reconstruction error for realistic image synthesis, which is vulnerable to agnostic privacy hacking by reverse engineering. In contrast, we maximize the reconstruction error for intuitive privacy-preserving. Also, discriminative and generative models are widely studied for latent feature learning, improving task inference but facilitating data reconstruction [35, 43]. They would make intuitive privacy protection even harder. Osia *et al.* [33] employ a combination of dimensionality reduction, noise addition, and Siamese fine-tuning to preserve privacy. Importantly both its dimensionality reduction and Siamese fine-tuning are based on discriminative training. Specifically its Siamese fine-tuning seeks to reduce the intra-class variation in features amongst training samples for the intended classification service. While the authors show these methods improve privacy, there is no systematic way to make tradeoffs between privacy and utility. In contrast, PAN presents a rigorous mechanism to discover good tradeoffs via a combination of discriminative, generative, and adversarial training. Mohammad *et al.* [29] present a privacy-preserving transformation called Replacement AutoEncoder (RAE). Like the Encoder in PAN, RAE also intends to eliminate sensitive information from the features while keeping the task-relevant information. Importantly it assumes that features/data relevant to an intended task do not overlap with those revealing sensitive information. As a result, it transforms the data by simply replacing the latter with features/data that are irrelevant to the intended task and do not reveal sensitive information. With that assumption, RAE eschews the hard problem of making a good tradeoff between utility and privacy and is also solely based on discriminative training. Furthermore, RAE does not reduce the amount of data that have to be sent to the service provider and would use significantly higher resources in transforming the data in which RAE will need to detect

sensitive features/data, replace them, and then reconstruct the (modified) raw data. In contrast, PAN only needs to run the dimensionality-reducing Encoder on the raw data and send the features to the service provider, although PAN may require significantly more computational resources in training the Encoder, which is done off-line, in the cloud.

7 CONCLUDING REMARKS

This paper addresses the privacy concern when mobile users send their data to an untrusted service provider for classification services. We present PAN, an adversarial framework to automatically generate deep features from the raw data with quantified guarantees in privacy and utility. We report a prototype of PAN on Android platforms and cloud servers. Evaluation using Android applications and benchmark datasets show that PAN's Encoder output attains a notably better privacy-utility tradeoff than known methods. To our surprise, it achieves even better utility than standard DNNs that are completely ignorant of privacy. We surmise that this surprising result can be understood from the perspective of manifold.

We also see three directions that the work reported in this paper can be extended. First, the PAN framework can accommodate other choices of context-aware utility, such as the sequence prediction, and privacy quantification, such as the information theory-based privacy, according to app requirements. Second, it can also integrate multiple utility discriminators and privacy attackers to train the Encoder, given the appropriate datasets accompanied by utility and privacy labels. Third, our experience shows that the training of multiple adversarial models in PAN must be carefully synchronized to avoid model degradation caused by difference in their objectives and convergence speeds. Therefore, more heuristics and insights for guaranteeing and accelerating training convergence are much needed.

ACKNOWLEDGMENTS

This work is supported in part by National Key R&D Program of China #2018YFB1003605, Natural Science Foundation of China (NSFC) #61472312, Shaanxi Fund 2018JM6125, Open Fund of State Key Laboratory of Computer Architecture, The Youth Innovation Team of Shaanxi Universities, and Natural Science Foundation (NSF) Grant #1611295. The idea behind PAN was conceived during Sicong Liu's yearlong visit to Rice University with support from China Scholarship Council to which the authors are grateful. The authors also thank the anonymous reviewers for their constructive feedback that has made the work stronger.

REFERENCES

[1] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. 2016. Deep learning with differential privacy. In *Proceedings of SIGSAC*. 308–318.

[2] Jaspreet Bhatia, Travis D Breaux, Liora Friedberg, Hanan Hibshi, and Daniel Smullen. 2016. Privacy risk in cybersecurity data sharing. In *Proceedings of ACM Workshop on ISCS*. ACM, 57–64.

[3] Jiawei Chen, Janusz Konrad, and Prakash Ishwar. 2018. Vgan-based image representation learning for privacy-preserving facial expression recognition. In *Proceedings of CVPR Workshops*. 1570–1579.

[4] Jen-Tzung Chien and Ching-Huai Chen. 2016. Deep discriminative manifold learning. In *Proceeding of ICASSP*. 2672–2676.

[5] Andrew Collette. 2018. HDF5 for Python. http://www.h5py.org/.

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *Proceedings of CVPR*.

[7] Cynthia Dwork, Moni Naor, Toniann Pitassi, and Guy N Rothblum. 2010. Differential privacy under continual observation. In *Proceedings of STC*. ACM, 715–724.

[8] Cynthia Dwork, Aaron Roth, et al. 2014. The algorithmic foundations of differential privacy. *Journal of Foundations and Trends in Theoretical Computer Science* (2014), 211–407.

[9] Cynthia Dwork, Adam Smith, Thomas Steinke, and Jonathan Ullman. 2017. Exposed! a survey of attacks on private data. *Annual Review of Statistics and Its Application* 4 (2017), 61–84.

[10] Harrison Edwards and Amos Storkey. 2015. Censoring representations with an adversary. *arXiv preprint arXiv:1511.05897* (2015).

[11] Murat A Erdogdu, Nadia Fawaz, and Andrea Montanari. 2015. Privacy-utility trade-off for time-series with application to smart-meter data. In *Proceedings of Workshops at AAAI*.

[12] Alessandro Giusti, Dan C Ciresan, Jonathan Masci, Luca M Gambardella, and Jurgen Schmidhuber. 2013. Fast image scanning with deep max-pooling convolutional neural networks. In *Proceedings of ICIP*. 4034–4038.

[13] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in Neural Information Processing Systems*. 2672–2680.

[14] Google. 2018. android.util.LruCache. https://developer.android.com/reference/android/util/LruCache.html.

[15] Google. 2018. TensorFlow Mobile. https://www.tensorflow.org/mobile/.

[16] GoogleCloud. 2018. Data Preparation. https://cloud.google.com/ml-engine/docs/tensorflow/data-prep.

[17] GoogleNow. 2018. Google Now Launcher. https://en.wikipedia.org/wiki/Google_Now.

[18] Jianping He and Lin Cai. 2017. Differential private noise adding mechanism: Basic conditions and its application. In *American Control Conference (ACC), 2017*. IEEE, 1673–1678.

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

[20] Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167* (2015).

[21] Kaggle. 2019. State Farm Distracted Driver Detection. https://www.kaggle.com/c/state-farm-distracted-driver-detection.

[22] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[23] Alex Krizhevsky, Nair Vinod, and Hinton Geoffrey. 2014. The CIFAR-10 dataset. https://goo.gl/hXmru5.

[24] Rudolf Kruse, Christian Borgelt, Frank Klawonn, Christian Moewes, Matthias Steinbrecher, and Pascal Held. 2013. Multi-layer perceptrons. Springer, 47–81.

[25] Yann LeCun. 1998. The MNIST database of handwritten digits. https://goo.gl/t6gTEy.

[26] Mu Li, Tong Zhang, Yuqiang Chen, and Alexander J Smola. 2014. Efficient mini-batch training for stochastic optimization. In *Proceedings of SIGKDD*. ACM, 661–670.

[27] Sicong Liu, Yingyan Lin, Zimu Zhou, Kaiming Nan, Hui Liu, and Junzhao Du. 2018. On-demand deep model compression for mobile devices: a usage-driven model selection framework. In *Proceedings of ACM MobiSys*.

[28] Aravindh Mahendran and Andrea Vedaldi. 2015. Understanding deep image representations by inverting them. , 5188–5196 pages.

[29] Mohammad Malekzadeh, Richard G Clegg, and Hamed Haddadi. 2018. Replacement autoencoder: A privacy-preserving algorithm for sensory data analysis. In *Proceedings of IEEE IoTDI*. 165–176.

[30] Ricardo Mendes and João P Vilela. 2017. Privacy-preserving data mining: methods, metrics, and applications. *IEEE Access* 5 (2017), 10562–10582.

[31] Fausto Milletari, Nassir Navab, and Seyed-Ahmad Ahmadi. 2016. V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *Proceedings of 3DV*. 565–571.

[32] Seong Joon Oh, Mario Fritz, and Bernt Schiele. 2017. Adversarial image perturbation for privacy protection a game theory perspective. In *Proceedings of ICCV*. 1491–1500.

[33] Seyed Ali Ossia, Ali Shahin Shamsabadi, Ali Taheri, Hamid R Rabiee, Nic Lane, and Hamed Haddadi. 2017. A hybrid deep learning architecture for privacy-preserving mobile analytics. *arXiv preprint arXiv:1703.02952* (2017).

[34] Nicolas Papernot, Shuang Song, Ilya Mironov, Ananth Raghunathan, Kunal Talwar, and Úlfar Erlingsson. 2018. Scalable private learning with pate. *Proceddings of ICLR* (2018).

[35] Alec Radford, Luke Metz, and Soumith Chintala. 2015. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434* (2015).

[36] Nisarg Raval, Ashwin Machanavajjhala, and Jerry Pan. 2019. Olympus: sensor privacy through utility aware obfuscation. *Proceedings of PET* (2019).

[37] Liu Sicong, Zhou Zimu, Du Junzhao, Shangguan Longfei, Jun Han, and Xin Wang. 2017. UbiEar: Bringing Location-independent Sound Awareness to the Hard-of-hearing People with Smartphones. *Journal of IMWUT* (2017).

[38] Google Brain team. 2018. TensorFlow. https://www.tensorflow.org/tutorials/.

[39] Stacey Truex, Nathalie Baracaldo, Ali Anwar, Thomas Steinke, Heiko Ludwig, and Rui Zhang. 2018. A hybrid approach to privacy-preserving federated learning. *arXiv preprint arXiv:1812.03224* (2018).

[40] UCI. 2017. Har: Dataset for Human Activity Recognition. https://goo.gl/m5bRo1.

[41] Zhenyu Wu, Zhangyang Wang, Zhaowen Wang, and Hailin Jin. 2018. Towards privacy-preserving visual recognition via adversarial training: A pilot study. In *Proceedings of ECCV*.

[42] Matthew D Zeiler, Dilip Krishnan, Graham W Taylor, and Rob Fergus. 2010. Deconvolutional networks. In *Proceedings of CVPR*.

[43] Guoqiang Zhong, Li-Na Wang, Xiao Ling, and Junyu Dong. 2016. An overview on data representation learning: From traditional feature learning to recent deep learning. *Journal of Finance and Data Science* (2016), 265–278.