Deep Private-Feature Extraction

Seyed Ali Osia, Ali Taheri, Ali Shahin Shamsabadi, Kleomenis Katevas, Hamed Haddadi, and Hamid R. Rabiee, *Senior Member, IEEE*

Abstract—We present and evaluate *Deep Private-Feature Extractor (DPFE)*, a deep model which is trained and evaluated based on information theoretic constraints. Using the selective exchange of information between a user's device and a service provider, *DPFE* enables the user to prevent certain sensitive information from being shared with a service provider, while allowing them to extract approved information using their model. We introduce and utilize the *log-rank* privacy, a novel measure to assess the effectiveness of *DPFE* in removing sensitive information and compare different models based on their accuracy-privacy trade-off. We then implement and evaluate the performance of *DPFE* on smartphones to understand its complexity, resource demands, and efficiency trade-offs. Our results on benchmark image datasets demonstrate that under moderate resource utilization, *DPFE* can achieve high accuracy for primary tasks while preserving the privacy of sensitive information.

Index Terms—Feature extraction, privacy, information theory, deep learning

1 Introduction

The increasing collection of personal data generated by, or inferred from, our browsing habits, wearable devices, and smartphones, alongside the emergence of the data from the Internet of Things (IoT) devices are fuelling a wide range of novel applications and services. These include healthcare and wellbeing apps, financial management services, personalized content recommendations, and social networking tools. Many of these systems and apps rely on continuous data sensing and collection at the user side, and upload of the data to a service provider for consequent analysis.

While many of the data-driven services and apps are potentially beneficial, the underlying unvetted and opaque data collection and aggregation protocols can cause severe data security threats and privacy risks [1]. Additionally, the excessive resource utilization (i.e., significant bandwidth and energy overheads) [2], can be inefficient especially for tasks such as video and time-series analytics, as often large volumes of data are collected and uploaded to servers for processing, to gain specific insights while discarding the dataset. Collection and processing of private information on the cloud introduces a number of challenges and trade-offs, especially when scalability of data collection and uploading practice are taken into consideration. The collected data are often fed into machine learning models for extracting insights and features of commercial interest, where the information is exposed to data brokers and service providers. While certain features of the data can be of interest for specific applications (e.g., location-based services, or mobile health applications), the presence of additional information in the data can lead to unintended subsequent privacy leakages [3], [4].

Current solutions to this problem, such as cryptography [5], [6], or complete data isolation and local processing [7], are not efficient for big data and techniques relying on deep learning [8], [9]. In today's data-driven ecosystem, these privacy challenges are an inherent side effect of many big data and machine learning applications. Critically, there are no clear privacy measures indicating the losses and gains in privacy for various protocols.

In this paper, we focus on providing privacy at the first step of this ecosystem: the exchange of acquired user data between the *end user* and a *service provider*. We propose a novel solution based on a dynamic optimization between scalability and privacy. The proposed framework is based on the idea that when preparing data for subsequent analysis by service provider, the end user does not need to hide all the information by means of cryptographic methods, which can be resource-hungry or overly complex for the end-user device. Instead, it might suffice to remove the sensitive features of the information (e.g., the user identity in a face image), while at the same time preserving the necessary information (e.g., age) for further analysis. This is also the case in many surveillance applications where a central node is required to process user data that may be sensitive in some aspects.

Fig. 1 demonstrates our proposed hybrid framework in which the user and cloud collaborate to analyze the raw user data in a private and efficient manner. Our work relies on the assumption that the service provider releases a publicly verifiable feature extractor module based on an initial training set. The user then performs a minimalistic analysis and extracts a *private-feature* from the data and sends it to the service provider (i.e., *the cloud*) for subsequent analysis. The private-feature is then analyzed in the cloud and the result yields back to the user. The fundamental challenge in using this framework is the design of the feature extractor module that removes sensitive information properly, and

- *S. Ali Osia, A. Taheri, and H. R. Rabiee are with the Advanced ICT Innovation Center, Department of Computer Engineering, Sharif University of Technology, Tehran, Iran. E-mail: {osia, ataheri}@ce.sharif.edu, rabiee@sharif.edu.*
- *A. S. Shamsabadi is with the School of Electronic Engineering and Computer Science, Queen Mary University of London, London E1 4NS, United Kingdom. E-mail: shahinshamsabadi_ali@ee.sharif.edu.*
- *K. Katevas and H. Haddadi are with the Dyson School of Design Engineering, Imperial College London, London SW7 2AZ, United Kingdom. E-mail: k.katevas@qmul.ac.uk, h.haddadi@imperial.ac.uk.*

Manuscript received 7 Mar. 2018; revised 22 Aug. 2018; accepted 14 Oct. 2018. Date of publication 30 Oct. 2018; date of current version 5 Dec. 2019. (Corresponding author: Hamid R. Rabiee.)
Recommended for acceptance by M. Wang.
Digital Object Identifier no. 10.1109/TKDE.2018.2878698

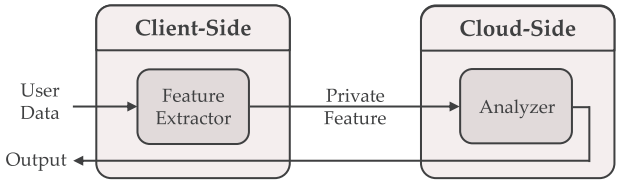


Fig. 1. The proposed hybrid framework for user-cloud collaboration.

on the other hand does not impact scalability by imposing heavy computational requirements on the user's device. We then propose a novel privacy measure to assess the impact of this approach on privacy versus utility.

The main contributions of this paper are: (i) Proposing a hybrid user-cloud framework for the user data privacy preservation problem which utilizes a private-feature extractor as its core component; (ii) Designing the private-feature extractor based on information theoretic concepts leading to an optimization problem; (iii) Proposing a deep neural network architecture to solve the optimization problem; and (iv) Proposing a measure to evaluate user privacy and verify the feature extractor module.[1]

In the rest of this paper, we first discuss the privacy issues in different aspects of machine learning and consider the problem of "user data privacy in interaction with cloud services" as the main purpose of this paper. To design the feature extractor module, we express our privacy preservation concerns in an optimization problem based on mutual information and relax it to be addressable by deep learning. We then present the Deep Private-Feature Extractor (*DPFE*), a tool for solving the aforementioned relaxed problem. We then propose a new privacy measure, the *log-rank* privacy, to verify the proposed feature extractor, measure its privacy, and evaluate the efficiency of the model in removing sensitive information. The log-rank privacy can be interpreted from different perspectives, including entropy, k-anonymity and classification error. We evaluate this framework under the facial attribute prediction problem by using face images. In this context, we remove the face identity information while keeping facial attribute information, and analyze the privacy-accuracy performance trade-off. Finally, we implement different private-feature extractors on mobile phone to compare the performance of different solutions and addressing the scalability concern.

2 Privacy in Machine Learning

Machine learning methods need to analyze sensitive data in many usecases to perform their desired tasks which may violate users' privacy. This fundamental dichotomy has been appearing in different problems. Public datasets and learning models pose threats to individuals' privacy. Moreover, sharing private data with a model or service provider may come with privacy threats. We may classify privacy concerns into *public dataset privacy*, *training phase privacy*, *training data privacy*, *model privacy* and *user data privacy* as depicted in Fig. 2. These concepts are discussed in the rest of this section. Since we are concerned with *user data privacy* in this paper, we discuss it with more details in Section 2.5.

2.1 Public Dataset Privacy

Training data is the crucial component of each learning system. Collecting and sharing rich datasets for data mining

1. All the code and models for the paper are available on https://github.com/aliosia/DPFE

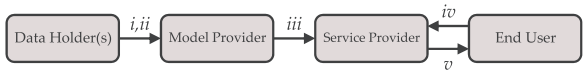


Fig. 2. Privacy concerns may exist when: (i) Data holder shares a public dataset: Anonymity of individuals are threatened; (ii) data holders participate in a model training procedure with their private data; (iii) a model provider shares a publicly-learned model: The privacy of the individuals' data used for training is at risk; (iv) an end user shares his/her data with the service provider: private information can be revealed to the service provider; and (v) a service provider shares query answers with the end user: An attacker can infer the model itself by launching repeated queries.

tasks can be highly beneficial to the learning community, although it might come with privacy concerns that make it a double-edged sword. Publishing a dataset that satisfies both parties by preserving the users' privacy and other useful information for data mining tasks, is a challenging problem and has long line of work. Agrawal and Srikant [10] were some of the first to address the privacy concern in data mining for sharing a generic dataset for learning tasks, in addition to considering users' privacy. They utilized a *randomization* technique, in which by adding noise to data they guaranteed its privacy. The resulting distribution of noisy data might have been different from the original distribution. To reconstruct the original distribution, a recovery method was introduced in the paper and extended by Agrawal et al. in [11]. By utilizing this method, it is possible to train a learning model on reconstructed data with the same distribution as the original data. Many works have followed this trend and extended this idea, however, this approach faces two important obstacles: curse of dimensionality and non-robustness to attacks [12], which make it inefficient for high dimensional data with side information.

k-anonymity is another popular option for addressing the problem of anonymous dataset publishing, first introduced by Sweeney [13]. Publishing a health database that contains patient sensitive information is one of the favored instance of k-anonymity usages. Assuming all data points have identity documents (IDs) that should be kept private, k-anonymity deals with transforming a dataset in a way that, having an individual data features, one cannot infer its ID among at least k identities. Many researches are presented to make a database k-anonymous [14], [15], [16], [17] and most of them are based on the generalization (e.g., removing the last digit of the patient zip code) or suppression of features (e.g., removing the name). Nevertheless, this approach deals with some important challenges when facing attacks [12], although [18], [19] and [20] tried to overcome these challenges. Furthermore, they are only well-suited for structured databases with high level features (e.g., relational databases) which makes them hard to deploy for other type of data (e.g., image and video). In [21], Newton et al. presented a k-anonymous image dataset by introducing the k-same algorithm. While they build the desired dataset by constructing average images among k identities, their proposed models are not always reliable.

2.2 Training Phase Privacy

A common problem of centralized learning is the collection of training data, especially when dealing with individual's sensitive data (e.g., health information). People are usually reluctant in sharing data that includes their habits, interests, and geographical positions. The upcoming solution to this problem is federated learning, where data holders keep their data private, while they communicate with a central node in order to train a learning model in a cooperative manner. The

authors in [22] tried to address this problem by using distributed stochastic gradient descent (*SGD*), where each party loads the latest parameters, update them using *SGD* and upload the new selected parameters to the central node that holds the global model. While in that case direct leakage of private data can be prevented, the uploaded gradients might still include sensitive information from the training data. Thus, a differentially private algorithm is required for sharing the gradients which is proposed in that work. This approach still has some major problems e.g., loose privacy bound addressed by [23] and potential threats by generative adversarial networks addressed by [24]. An alternative solution to this problem could be the use of cryptographic techniques like secure multi-party computation, recently used by [25]. However, these techniques are still not applicable on complex neural networks, due to their low efficiency and accuracy.

2.3 Training-Data Privacy

The growing popularity of public learning models raises the concern of privacy of the individuals involved in the training dataset. Differentially private algorithms brought us a rigorous answer to this problem, by providing a method to answer queries from a statistical database, without disclosing individuals' information, as formalized by [26]. An algorithm is called differentially private if the conditional likelihood ratio of presence and absence of an individual, given the transformed statistic, is close to one. Adding noise to the original statistic is one popular method leading to differential privacy. We can consider a learning model as a complex statistic of its training data which should not reveal information about the individuals. Answering complex queries by combining simple queries is the way various learning models, such as Principal Component Analysis and k-means, can be made differentially private (see the surveys by [27] and [28]). Recently differentially private deep models were proposed by [29]. The authors in [23] introduced a privacy preservation framework by utilizing differential privacy which is not specific to the learning model and possesses a state of the art privacy-accuracy trade-off.

2.4 Model Privacy

Model privacy is the concern of the service provider and deals with keeping the learning model private, while returning the inference results to the user. Throughout these years, less attention has been paid to the model privacy, although some works such as [30] studied this problem. In general, an adversary can infer the model parameters by making many queries to the learning model and aggregate the answers. The authors in [30] considered this approach for some basic models, e.g., logistic regression, multilayer perceptron and decision tree.

2.5 User Data Privacy

The increasing usage of cloud-based systems has triggered a situation where preserving privacy is a challenging but important task. That is, when the user data and the pretrained learning model is not accessible from the same place, inevitably user data must be sent to the service provider for further analysis. Usually, cryptographic schemes are prevalent in these situations, where two parties do not trust each other. Focusing on the deep models offered by a cloud service, [31] introduced this problem and proposed a homomorphic encryption method to execute the inference directly on the encrypted data. Even though this work is an interesting approach to the problem, a number of shortcomings makes it impractical. In fact, approximating a deep neural network with a low degree polynomial function may not be feasible without sacrificing accuracy. Furthermore, the complexity of the encryption is relatively high which makes it inefficient for the real-world online applications. Alternatives to fully homomorphic encryption have been suggested in [32] and [33]. They used garbled circuit protocol and addressed some of the discussed challenges, however they are limited to utilization of simple neural networks and had high computational costs.

Using learning algorithm to obtain privacy is an independent orientation of cryptographic techniques. Osia et al. considered the privacy challenge for users' identity by using deep models and layer separation over the cloud [8] and [9]. The authors in [34] utilized a similar method and tried to address the privacy challenge by separating only the first layer of the network. However, their definition of privacy is limited and fundamentally different from ours. In addition, their method is not flexible to fulfill the required privacy constraints in many applications, and they do not provide any metric for privacy evaluations. The privacy challenge in dealing with cloud server is also discussed by [35] and [36] using accelerometer sensor's data.

Although all of these methods are useful in specific domains, they lack a clear privacy measure for the general concept of privacy in different frameworks. In this paper, we address these issues by introducing a new privacy measure called log-rank privacy.

3 Problem Formulation

In this section, we address the user data privacy through solving an optimization problem. In the next section, we show how to solve this optimization problem with deep neural networks.

The user data privacy challenge rises when an end-user tries to share her sensitive data with the service provider to infer some knowledge about the data. Encryption-based solutions try to encode all the information such that only authorized users can access those information. However, these methods have important shortcomings which are discussed in Section 2.5. We address this challenge in a different manner from the encryption-based methods. The key intuition is that for many applications we can just remove the user's *sensitive* (unauthorized) information while retaining the ability to infer the *primary* (authorized) information. For instance, we may want to focus on hiding individuals' identities in a video surveillance system, but still allow to count the number of participants. In this scenario, a trivial solution is to censor people's faces in the frames, however this solution fails when the purpose is to measure the facial attributes such as emotion or gender. Henceforth, we address this problem as a *privacy preservation problem* and use the terms *primary* and *sensitive* information as the information needed to be preserved and removed, respectively. Assuming the service provider knows the primary and sensitive random variables, we abstract this concept as an optimization problem by utilizing mutual information (see Appendix A, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TKDE.2018.2878698 for information theoretic preliminaries).

Let $\mathbf{x}$ be the input, $\mathbf{z}$ the primary, and $\mathbf{y}$ the sensitive variables. We would like to extract a feature $\mathbf{f}$, by applying a

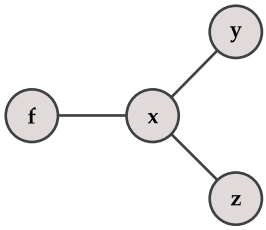


Fig. 3. Private-feature extraction probabilistic graphical model.

function **g** on **x**, which is informative about the primary variable and non-informative about the sensitive variable. We refer to the extracted feature as *private-feature*. More specifically, the desired private-feature is obtained through maximizing mutual information between the feature and primary variable $I(\mathbf{f};\mathbf{z})$, while minimizing mutual information between the feature and sensitive variable $I(\mathbf{f};\mathbf{y})$, as follows:

$$\begin{aligned} \max_{\mathbf{f}} \quad & I(\mathbf{f};\mathbf{z}) - \beta I(\mathbf{f};\mathbf{y}) \\ s.t. \quad & \mathbf{f} = g(\mathbf{x}), \end{aligned}$$

where $I(A;B)$ represents the mutual information between two random variables A and B.

Even though at the first glance it seems that the optimal solution of this problem is to set **f** equal to the best estimation of **z**, this is not applicable in many real world applications because: (a) the optimal model which perfectly predicts **z** can be too complicated, and hence using such a feature extractor in the client-side is impossible; and (b) the service provider may not share the whole model with the client for some reasons such as copyright issues. Assuming we can accurately estimate **f** by using a member of family of functions $\mathbb{G} = \{g(x;\theta)|\theta \in \mathbf{\Theta}\}$, then the optimization problem becomes:

$$\begin{aligned} \max_{\theta} \quad & I(\mathbf{f};\mathbf{z}) - \beta I(\mathbf{f};\mathbf{y}) \\ s.t. \quad & \mathbf{f} = g(\mathbf{x};\theta), \end{aligned} \tag{1}$$

where, **f** is a deterministic function of the input variable, parameterized by θ. The graphical model of this problem is shown in Fig. 3.

Optimizing mutual information has been widely used in many information theoretic approaches of machine learning problems. The authors in [37] formulated the *Infomax* and tried to address the problem of unsupervised deterministic invertible feature extraction by maximizing the mutual information between the input and feature. The authors in [38], relaxed the limiting invertibility constraint and used a variational approach which leads to the *IM* algorithm for maximizing the mutual information. Recently, the authors in [39] used a similar method to maximize the mutual information in generative adversarial networks. These works can be considered as the fundamental works in the problem of unsupervised feature extraction from information theoretic viewpoint, however, since we are utilizing a supervised approach, those methods cannot be applied to our case. Among works considering supervised feature extraction, the *information bottleneck* introduced in [40] is the most relevant work. In general, Information bottleneck provides an information theoretic framework for analyzing the supervised feature extraction procedure. Although their optimization problem almost looks similar to ours, there is a fundamental difference between the two approaches. More specifically, they use $I(\mathbf{f};\mathbf{x})$ instead of $I(\mathbf{f};\mathbf{y})$, meaning that irrelevant information to **z** should be removed by minimizing $I(\mathbf{f};\mathbf{x})$ in the process of feature extraction. Therefore, they can not directly consider the privacy constraints about **y**. Moreover, their optimization problem is solved through an analytical approach that assumes that the joint probability distribution $p(x,z)$, is known. However, in practice this distribution is often unavailable. Although their analytical method is impractical, nevertheless, their framework provides a powerful tool for analysis of the supervised feature extraction methods.

Similar to the information bottleneck optimization problem, the private-feature extraction problem (Equation (1)) is non-convex and can not be solved through the known convex optimization algorithms. To overcome this challenge, it is common to bound the optimization problem, and then by using the iterative methods similar to [38] and [41], obtain the desired results. To this end, we first obtain the lower and upper bounds for $I(\mathbf{f};\mathbf{z})$ and $I(\mathbf{f};\mathbf{y})$ respectively, and then try to maximize the lower bound of Equation (1). Henceforth, we assume **y** to be a discrete sensitive variable in order to address the classification privacy problem.

Lower Bound For $I(\mathbf{f};\mathbf{z})$. We derive a variational lower bound for mutual information by first expressing Lemma 1 and then proving Theorem 2.

Lemma 1. *For any arbitrary conditional distribution* $q(z|f)$*, we have:*

$$\begin{aligned} I(\mathbf{f};\mathbf{z}) &\geq \mathbb{E}_{\mathbf{f},\mathbf{z}} \log \frac{q(z|f)}{p(z)} \\ &= H(\mathbf{z}) + \mathbb{E}_{\mathbf{f},\mathbf{z}} \log q(z|f) \end{aligned} \tag{2}$$

Proof. See Appendix B.1, available in the online supplemental material. □

Theorem 2. *The lower bound* $\mathcal{L}$ *for* $I(\mathbf{f};\mathbf{z})$ *is given by:*

$$\mathcal{L} = H(\mathbf{z}) + \max_{\phi} \mathbb{E}_{\mathbf{f},\mathbf{z}} \log q(z|f;\phi). \tag{3}$$

Proof. For all members of a parametric family of distributions $\{q(z|f;\phi)|\phi \in \Phi\}$, the right hand side of Equation (2) can be considered as the lower bound for mutual information. The equality happens when $q(z|f)$ is equal to $p(z|f)$. Therefore, if we consider a rich family of distributions for q in which a member can approximate $p(z|f)$ well enough, we can obtain a tight enough lower bound for mutual information by maximizing the right hand side of Equation (2) with respect to ϕ. By utilizing the definition of Entropy, we obtain $\mathcal{L}$ as the desired lower bound. □

Upper Bound for $I(\mathbf{f};\mathbf{y})$. A two-step procedure can be used to find an upper bound for the mutual information. First, we use the Lemma 3 and Jensen inequality to prove Theorem 4, and obtain $\mathcal{U}_1$ as the primitive upper bound for $I(\mathbf{f};\mathbf{y})$. Then we use kernel density estimation (*KDE*) (see [42]) and use Lemmas 5 and 6 to obtain $\mathcal{U}_2$ as the desired upper bound for $I(\mathbf{f};\mathbf{y})$ through Theorem 7.

Lemma 3. *Assume* **y** *is a discrete random variable with* $\{y_a|1 \leq a \leq c_y\}$ *as its range, then:*

$$I(\mathbf{f};\mathbf{y}) = \sum_a p(y_a) \int p(f|y_a) \log \frac{p(f|y_a)}{\sum_b p(y_b)p(f|y_b)}\, df.$$

Proof. By substituting $\frac{p(f,y)}{p(f)p(y)}$ with $\frac{p(f|y)}{p(f)}$ and then $p(f)$ with $\sum_b p(y_b)p(f|y_b)$ in the main formula of $I(\mathbf{f};\mathbf{y})$, we obtain the desired formula. □

By utilizing Jensen inequality (see Appendix A, available in the online supplemental material) and manipulating Lemma 3, we can compute $\mathcal{U}_1$ as a primitive upper bound for mutual information, as follows.

Theorem 4. *The upper bound $\mathcal{U}_1$ for $I(\mathbf{f};\mathbf{y})$ is given by:*

$$\mathcal{U}_1 = \sum_a \sum_{b:b\neq a} p(y_a)\ p(y_b)\ D_{kl}\big[p(f|y_a)\|p(f|y_b)\big]. \quad (4)$$

Proof. See Appendix B.2, available in the online supplemental material. □

Since computing $\mathcal{U}_1$ by Equation (4) is not tractable, we use an approximation technique to obtain the upper bound. By employing the kernel density estimation, we can efficiently estimate $p(f)$ [43]. We then utilize the Silverman's rule of thumb [44] and use a Gaussian kernel with the desired diagonal covariance matrix. Next, by normalizing each dimension of the feature space to have zero mean and unit variance, we acquire a symmetric Gaussian kernel with *fixed* covariance matrix, σI, where σ is a constant depending on the dimensionality of the feature space and the size of the training data. This kind of normalization is a common process in machine learning [45] and is impartial of relations among different dimensions including independency and correlation. Finally, conditioning on $\mathbf{y}$, for each y_a we can think of $p(f|y_a)$ as a Gaussian Mixture Model (*GMM*) (see [42]) and use the following lemmas from [46] to obtain a reliable upper bound.

Lemma 5. *[46] For two multidimensional Gaussian distributions, p and q, with μ_p and μ_q as their expected values and the same covariance matrix σI, we have:*

$$D_{kl}(p\|q) = \frac{1}{2\sigma}\|\mu_p - \mu_q\|_2^2.$$

Lemma 6. *[46] For two given GMMs $p = \sum_a \pi_a p_a$ and $q = \sum_b \omega_b q_b$, we have:*

$$D_{kl}(p\|q) \leq \sum_{a,b} \pi_a\ \omega_b\ D_{kl}(p_a\|q_b).$$

where for each a and b, p_a and q_b are Gaussian distributions forming the mixtures.

We can use Theorem 4, Lemma 5 and Lemma 6 to derive the desired upper bound for $I(\mathbf{f};\mathbf{y})$.

Theorem 7. *Having large training data, the upper bound $\mathcal{U}_2$ for $I(\mathbf{f};\mathbf{y})$ is given by:*

$$\mathcal{U}_2 = \frac{1}{\sigma N^2} \sum_{\substack{(i,j):\\ y_i\neq y_j}} \|f_i - f_j\|_2^2, \quad (5)$$

where f_i is the extracted feature from data x_i and its corresponding label y_i. The sum is over pairs of points with different $\mathbf{y}$ labels.

Proof. See Appendix B.3, available in the online supplemental material. □

In other words, $\mathcal{U}_2$ is an upper bound that is proportional to the average euclidean distance between each pairs of feature vectors having different $\mathbf{y}$ labels. This value is practically hard to estimate, especially when we use *SGD*, and have large number of classes. Therefore, as stated in Theorem 8 and Corollary 9, we use an equivalent relation for $\mathcal{U}_2$ which is easier to optimize.

Theorem 8. *Constraining the variance of each dimension of feature space to be 1, we have:*

$$\mathcal{U}_2 = \frac{1}{\sigma N^2} \sum_{\substack{(i,j):\\ y_i=y_j}} (c - \|f_i - f_j\|_2^2). \quad (6)$$

where c is a constant function of feature space dimension and number of training data.

Proof. See Appendix B.4, available in the online supplemental material. □

Corollary 9. *We can optimize the right hand side of Equation (6) instead of Equation (5) to obtain $\mathcal{U}_2$.*

Considering Corollary 9 together with Theorem 8, we realize that for a random pair of feature points, we should decrease their distance if the y labels are different, and increase their distance if they are the same. This is very similar to the *Contrastive* loss idea presented in [47] which is a popular loss function for *Siamese* architecture [48]. Siamese networks are used for metric learning purposes and tends to form a feature space in which similar points are gathered near each other. This is the opposite of what we aim to achieve: increase the distance of similar points and decrease the distance of dissimilar points.

By utilizing the suggested lower and upper bounds, we can substitute the original private-feature extraction problem (Equation (1)) with the following relaxed problem.

$$\begin{aligned} \min_{\theta,\phi} & \sum_{f_i,z_i} -\log q(z_i|f_i;\phi) \\ & + \frac{\beta}{2\sigma N^2}\left[\sum_{(i,j):y_i\neq y_j} \|f_i - f_j\|_2^2 + \sum_{(i,j):y_i=y_j} (c - \|f_i - f_j\|_2^2)\right] \\ s.t.\ & f_i = g(x_i;\theta). \end{aligned} \quad (7)$$

Considering the above equation, we should optimize an objective function that consists of two loss functions: the loss of the primary variable preservation modeled by a classification loss (first term), and the loss of sensitive variable elimination modeled by a contrastive loss (second term). Thus, the general training framework of the private-feature extractor contains three main modules: feature extractor, primary variable predictor and sensitive variable remover, as shown in Fig. 4. Note that according to the second term of Equation (7), the loss function of removing sensitive variable is defined on the pairs of samples, and as a result the $\mathbf{y}$-remover module also operates on pairs of features.

We propose a general deep model along with *SGD*-based optimizers to solve the optimization problem in Equation (7), as explained in the next section.

4 Deep Architecture

By utilizing the latest breakthroughs in the area of deep neural networks, we can practically find good local optimums of

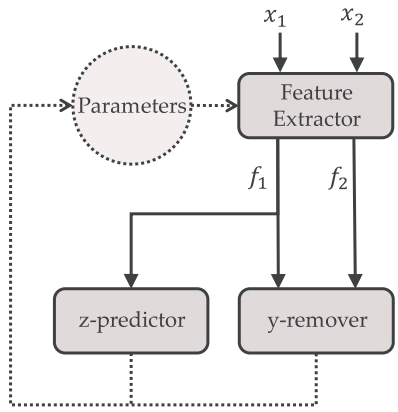


Fig. 4. The private-feature extraction framework. z and y are the primary and sensitive variables, respectively. x_1 and x_2 are two independent samples, and f_1 and f_2 are their corresponding features. **z**-predictor just uses f_1 to compute the first term of loss function, whereas **y**-remover uses both f_1 and f_2 to compute the second term of loss function (see Equation (7)). Solid lines show data flow and dotted lines indicate affection.

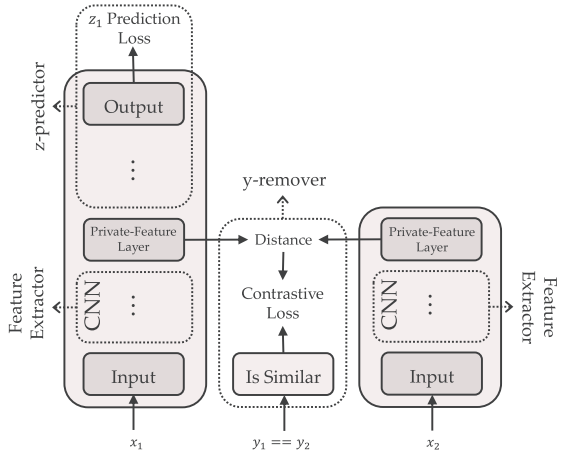


Fig. 5. Deep CNN architecture for private-feature extraction (*DPFE* architecture). x_1 and x_2 are independent random samples and y_1 and y_2 are their corresponding sensitive labels. **y**-remover first checks the equality of sensitive labels and then apply the information removal loss function.

non-convex objective functions through *SGD* based algorithms, and accurately estimate complex non-linear functions. Today, a large portion of state of the art learning models are deep. Therefore, having a general framework for privacy preserving deep inference is necessary. In this paper, we focus on image data (in the context of identity versus gender, expression, or age recognition), and propose a deep architecture based on CNN (Fig. 5) to optimize the objective function of the relaxed problem (Equation (7)). It is worth mentioning that the proposed framework can be generalized for other applications using feed forward networks or CNNs.

We call the proposed *Deep Private-Feature Extractor* architecture; *DPFE*. We consider two consecutive CNNs; one as the feature extractor and the other as the primary variable predictor as shown in Fig. 5. In this figure, z_1 prediction loss and contrastive loss are modeling the first and second term of Equation (7), respectively and their weighted sum builds the ultimate loss function which can be minimized using *SGD* based algorithms.

A simple strategy for having two consecutive CNNs is the *layer separation mechanism* introduced in [49]. We can also employ batch normalization layer [50] and normalize each dimension of the feature space, as stated in Section 3. In the following, we first introduce the layer separation mechanism, and then proceed with dimensionality reduction and noise addition issues that can enhance the preservation of privacy.

4.1 Layer Separation Mechanism

To deploy our framework, we can start from a pre-trained recognizer of primary variable (e.g., a deep gender recognition model), and make it private to the sensitive variable (e.g., identity). In order to do this, we choose the output of an arbitrary intermediate layer of the pre-trained model as the preliminary private-feature and simply partition the layers of the model into two sets: the elementary and the secondary layers that form the feature extractor and the primary variable predictor, respectively. In this way, the same model can be easily fine-tuned by just appending the contrastive loss function and continuing the optimization process, leading to the *private-feature* as the intermediate layer's output. This procedure is shown in Fig. 5.

One may argue that separating the layers of a deep model is sufficient to obtain an ideal private-feature in the intermediate layer due to the nature of deep networks. In general, the higher layers of a deep architecture provide a more abstract representation of the data and drop the irrelevant information including the sensitive information [51] and preserve the primary variable [52]. Therefore, there is no need to fine-tune the model with the suggested *DPFE* architecture. However, this argument can easily be rejected by considering the counter example provided by deep visualization techniques. For example, [53] provided a method to reconstruct the input image from intermediate layers of a deep network. Osia et al. used this method in [49] and demonstrated that the original face image can be reconstructed from some intermediate layers of the gender recognition model. Thus, there is no guarantee that the intermediate layers drop the sensitive information (identity in this case).

4.2 Dimensionality Reduction

Imagine that the extracted private-feature has a low dimension (in Section 5 we use a 10-dimensional feature space). In this case, we will benefit from the following advantages:

- We can highly decrease the communication cost between user and the service provider, because instead of sending the raw input data to the cloud, the user will only send the private-feature of the input.
- As shown in Section 5, we need to estimate an expectation to measure the privacy, thus lower dimension will help us to avoid the curse of dimensionality during the approximation process.
- Reducing the dimension of the private-feature will intrinsically improve privacy as suggested by [49] and [36].

Nevertheless, a potential disadvantage of the dimensionality reduction is that it can negatively affect on the accuracy of the primary variable prediction. However, we show in our experiments that the adverse effect of dimensionality reduction is negligible.

Reducing the dimensionality can be done as a preprocessing step on the pre-trained network. In fact, after choosing the intermediate layer, we can first execute the following operations: (i) Embed an auto-encoder with a low dimensional hidden layer on top of the chosen layer; (ii) Fine-tune the model to obtain the new primary variable predictor, and

(iii) Choose the auto-encoder's hidden layer as the new intermediate layer which is low dimensional. Consequently, we can fine-tune the model with *DPFE* architecture to get a low dimensional private-feature.

4.3 Noise Addition

As mentioned earlier, many of the privacy preservation methods, from randomization technique to differentially private algorithms, rely on noise addition to gain privacy as it increases the uncertainty. We can utilize this technique after finishing the training procedure, in the test phase, when the dimensionality reduction is employed and the granularity of the sensitive variable is finer than the primary variable (e.g., identity is finer than gender).

Adding noise to the private-feature will smooth out the conditional distributions of both primary and sensitive variables and form a trade-off between privacy (of sensitive variable) and accuracy (of primary variable). This trade-off can be helpful in real world applications, because one can choose the desired point on privacy-accuracy curve, based on the the importance of privacy or accuracy in a specific application. We will discuss this trade-off in detail in Section 6.

5 Privacy Measure

In this section, we propose a method for evaluating the quality of privacy algorithms. Considering the problem formulation by mutual information (Equation (1)), one may suggest the negative of mutual information between the extracted private-feature and the sensitive variable ($-I(\mathbf{f};\mathbf{y})$), as a privacy measure. Since $I(\mathbf{f};\mathbf{y}) = H(\mathbf{y}) - H(\mathbf{y}|\mathbf{f})$ and $H(\mathbf{y})$ is constant, this approach is equivalent to considering $H(\mathbf{y}|\mathbf{f})$ as the privacy measure. However, this measure has two shortcomings: (i) it is difficult to obtain an efficient estimation of $p(y|f)$; and (ii) there is no intuitive interpretation of this measure for privacy. In order to resolve these problems, we can relax the definition of uncertainty. We achieve this by partitioning the conditional probabilities by their rank order and build a lower bound for the conditional entropy:

$$H(\mathbf{y}|\mathbf{f}) = \int p(f) \sum_{y_a=1}^{c_y} p(y_a|f) \log \frac{1}{p(y_a|f)} df.$$

It is known that among some numbers which sums into one, the r'th highest value is lower than or equal to $\frac{1}{r}$. So if we consider $r_{f,a}$ as the rank of $p(y_a|f)$ in the set of $\{p(y_j|f)|j \in \{1,\ldots,c_y\}\}$ sorted descending, we have:

$$\begin{aligned} H(\mathbf{y}|\mathbf{f}) &\geq \int p(f) \sum_{y_a=1}^{c_y} p(y_a|f) \log r_{f,a} df \\ &= \mathbb{E}_{p(f,y)} \log \mathbf{r} \triangleq \mathcal{L}_{rank} \end{aligned} \tag{8}$$

which leads to the following definition by dividing all formulas by $\log c_y$, in order to have a normalized measure between zero and one.

Definition 10 (Log-Rank Privacy). *The* log-rank privacy *of a discrete sensitive variable* $\mathbf{y}$, *given the observed feature vector* $\mathbf{f}$, *is defined as:*

$$LRP(\boldsymbol{y}|\boldsymbol{f}) = \frac{1}{\log c_y} \mathbb{E}_{p(f,y)} \log \boldsymbol{r}, \tag{9}$$

where $\boldsymbol{r}$ *is a random function of* $\boldsymbol{f}$, *and* $\boldsymbol{y}$ *corresponds to the rank of* $p(y|f)$ *in the set of* $\{p(y_j|f)|j \in \{1,\ldots,c_y\}\}$ *which has been sorted in a descending order.*

Assuming we have an estimation of $p(y|f)$, log-rank privacy can be empirically estimated by the sample mean of training in the rank logarithm:

$$\hat{LRP}(\mathbf{y}|\mathbf{f}) = \frac{1}{N \log c_y} \sum_{i=1}^{N} \log\Big(rank\big(p(y_i|f_i), S_i\big)\Big)$$
$$S_i = \{p(y_j|f_i)|j \in \{1,\ldots,c_y\}\}$$

where $rank(a, S)$ for $a \in S$ is the rank of a in the descending ordered set S. In the following, we provide some intuition about the log-rank privacy and its relation to entropy, k-anonymity and classification error.

20-Questions Game Interpretation. Consider the 20-questions game, in which we want to guess an unknown object by asking yes/no questions from an oracle. As stated in [54], the entropy is equivalent to the minimum number of questions one could ask in order to find the correct answer. Now consider the situation where we cannot ask *any* kind of yes/no questions, but only the questions in which we guess to be a candidate for the final answer, e.g., 'is the answer a chair?'. Moreover, assume that if we can guess the correct answer after k questions, we would be penalized by $\log k$; so that the wrong guesses are punished more at the beginning. Evidently, the optimal strategy to have the minimum expected penalty is to guess the objects in the same order with their probabilities. Using this strategy, the expected penalty would be equal to the *log-rank privacy*.

k-Anonymity and Expected Rank. k-anonymity deals with the number of entities we are equally uncertain about. Expected rank can be considered as a soft interpretation of this number, relaxing the equal uncertainty with the weighted sum of ranks. Thus, the rank variable expectation can be thought as the expected number of entities that we are in doubt about.

Classification Error Extension. One could suggest using *classification error (zero-one loss)* as the privacy measure, as it represents the deficiency of the classifier. Using this measure is equal to considering zero and one penalty for the correct and wrong guesses of the *first* question, respectively. Thus, two situations where we can find the correct label in the second and tenth question are considered equal and both penalized by one. The log-rank privacy handles this issue by penalizing different questions using their ranks' logarithm and can be considered as an extension of classification error.

Sensitivity Analysis. Empirically approximating an expected value by drawing samples from a probability distribution is a common method in machine learning [42]. For comparing empirical estimation of log-rank privacy with entropy, we need to estimate the order of probabilities in the former, while the exact values of probabilities are needed in the later. In general, approximating the log-rank privacy is less sensitive to the error of the density estimation and can gain lower variance. Detailed sensitivity analysis is out of scope of this paper and will be considered in future work.

6 Evaluation

In this section, we evaluate the proposed private-feature extractor by considering the problem of *facial attribute* prediction. We use each face image as an input and infer its

facial attributes such as gender, expression, or age, in a supervised manner. We extract a feature for facial attribute prediction, which at the same time is non-informative with respect to the *identity* of the person (sensitive attribute). In all of our experiments, we used the *CelebA* face dataset, presented in [55], which includes 40 binary facial attributes, such as gender (male/female), age (young/old), and smiling (yes/no) with the corresponding identity labels. In the following, first we explain the experiment setting and then we discuss the results.

6.1 Experiment Setting

In our evaluations, we used the layer separation mechanism followed by dimensionality reduction and noise addition. We selected the state of the art pre-trained facial attribute prediction model presented in [56] and called it the *original model*.[2] Then, we chose an attribute set from *CelebA* binary attributes (e.g., {gender & age}) as the primary variable to preserve its information (the sensitive variable is *identity* in all simulations). Next, we selected an intermediate layer (e.g., layer conv7) of the chosen network. Since this layer can also be a high dimensional tensor, we embedded a linear auto-encoder and applied batch normalization on its hidden layer to obtain the normalized intermediate features. Finally, by fine-tuning the network, we may have an attribute prediction model with low dimensional intermediate feature, which we refer to as the *Simple model* in the rest of the paper. While the low-dimensional feature preserves the information of attributes (see Theorem 2), it does not necessarily omit the sensitive information. Hence, we should fine-tune the network with the proposed *DPFE* architecture (Fig. 5) to remove identity information from the intermediate features. We refer to this model as the *DPFE model*. These steps are depicted in Procedure 1. We implemented all the models with the *Caffe* framework [57], utilizing the *Adam* optimizer [58], and a contrastive loss function.

Algorithm 1. DPFE Training Phase

Input: training data, intermediate layer, attribute set
$M_0 \leftarrow$ attribute prediction model
$L \leftarrow$ intermediate layer of M_0 (e.g., *conv7*)
$|L| \leftarrow$ size of L's output
$A \leftarrow$ attribute set (e.g., *{Gender & Age}*)
$AE \leftarrow$ linear auto-encoder with input/output size of $|L|$
$H \leftarrow$ hidden layer of AE (private-feature layer)
Initialize AE with PCA weights on L's output
$M_1 \leftarrow$ embed AE to M_0 on top of L
$S_{L,A} \leftarrow$ fine-tune M_1 on A
$z \leftarrow A$, $y \leftarrow Identity$
$P_{L,A} \leftarrow$ fine-tune $S_{L,A}$ with *DPFE* architecture
Output: $S_{L,A}$: simple model, $P_{L,A}$: *DPFE* fine-tuned model, H: private-feature layer

We evaluated each fine-tuned model based on the following criteria:

- Accuracy of the facial attribute prediction: achieving higher accuracy implies that the primary variable information is well preserved.
- Identity privacy: we evaluate the privacy of the feature extractor, using two different measures. First, the log-rank privacy measure, introduced in Section 5. Second, we utilize 1NN identity classifier and consider its misclassification rate which must be high in order to preserve privacy (although this condition is not sufficient as discussed in Section 5. We also use the deep visualization technique presented in [53] to demonstrate that the higher layers of the deep network may not be reliable.

To show the generality of the proposed method, we consider four different intermediate layers (conv4-2, conv5-1, conv6-1 and conv7) together with five attribute sets (listed below), and the results for twenty *Simple* and twenty *DPFE* models.

- G: {gender}
- GA: {gender, age}
- GAS: {gender, age, smiling}
- GASL: {gender, age, smiling, big lips}
- GASLN: {gender, age, smiling, big lips, big nose}

In what follows, we first explain the accuracy-privacy trade-off based on the log-rank privacy measure and 1NN misclassification rate (Section 6.2). We then present the visualization result (Section 6.3), and finally address the complexity issue of the private-feature extractor by implementing the proposed framework on a smartphone (Section 6.4).

6.2 Accuracy *versus* Privacy

To evaluate *Simple* and *DPFE* models, we designed the following four experiments and assessed different models based on their accuracy-privacy trade-off:

1) We compared *Simple* and *DPFE* models to show the superiority of *DPFE* fine-tuning;
2) We assessed the effect of different intermediate layers to indicate the appropriateness of higher layers;
3) We evaluated the effect of extending attribute set and showed that preserving privacy becomes harder;
4) We considered mean and standard deviation of Rank-privacy measure to guarantee privacy.

Algorithm 2. DPFE Test Phase

Input: test data, intermediate and private-feature layers, attribute set, model
$H \leftarrow$ private-feature layer
$A \leftarrow$ attribute set
$M \leftarrow$ model
$C \leftarrow$ covariance matrix of H in M
for $r \in \{ratios\}$ **do**
  $N_r \leftarrow$ Gaussian noise layer with covariance rC
  $M_r \leftarrow$ embed N_r as an additive noise on H in M
  $H_r \leftarrow$ output of $H + N_r$
  $p_r \leftarrow$ *identity* privacy of H_r
  $a_r \leftarrow$ average accuracy of M_r on A
end for
plot accuracy-privacy curve using $\{(a_r, p_r) | r \in \{ratios\}\}$
Output: accuracy-privacy trade-off

In order to adjust the accuracy-privacy trade-off, we used the noise addition mechanism. After the training phase, we estimate the covariance matrix of the feature space, scale it with different ratios and use it as a covariance matrix of a Gaussian noise. By increasing the amount of noise, the accuracy of the primary variable prediction decreases but the

2. We used a similar implementation from https://github.com/camel007/caffe-moonwhich used the tiny darknet architecture from https://pjreddie.com/darknet/tiny-darknet.

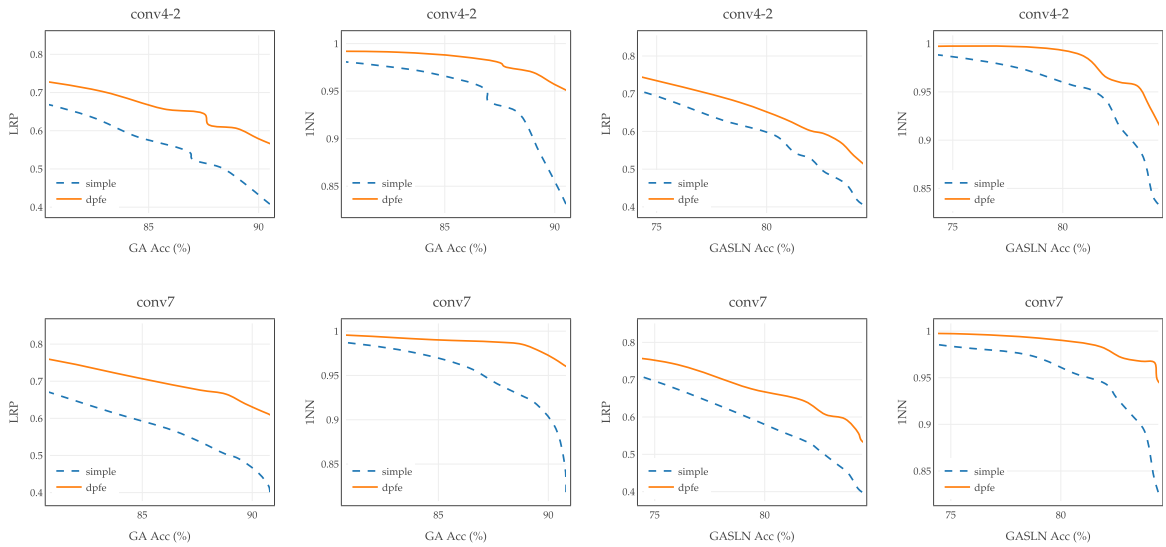


Fig. 6. *DPFE versus* Simple models: Fine-tuned models with *DPFE* architecture achieve Acc-Priv superiority to corresponding Simple models in all layers and attribute sets.

privacy of the sensitive variable increases. As a result, we can build the accuracy-privacy trade-off curves in a manner similar to the trade-off in rate-distortion theory (see [54]). The evaluation steps are shown in Procedure 2. The accuracy-privacy curves of different models can be compared based on the following definition.

Definition 11 (Acc-Priv Superiority). *For two models that try to preserve privacy of a sensitive variable and maintain accuracy of a primary variable, the one which always results in higher value of privacy for a fixed value of accuracy, is* Acc-Priv superior.

Considering Equation (7), it seems that the relative importance of accuracy and privacy can be controlled by changing the values of parameter β. However, this is not feasible in practice due to the challenges in the training stage. For example, training with a constant β and consequent noise addition mechanism, it is possible to set different accuracy-privacy strategies by utilizing a single trained model. This is not the case when we have various models by considering different values for β. We used cross validation, in order to choose a suitable fixed value for β in our experiments.

We computed the accuracy-privacy trade-off on the test data with 608 identities. Setting noise to zero, for all intermediate layers and attribute sets, *Simple* and *DPFE* models reached the same accuracy level as the *original* model with an error margin of less than 0.5 percent.[3] Therefore, we can conclude that all *Simple* and *DPFE* models preserve the facial attribute information, and we may concentrate on their privacy performance.

Effect of DPFE Fine-Tuning. In order to verify the superiority of *DPFE* fine-tuning over *Simple* fine-tuning, we compared the accuracy-privacy curve of different models, fine-tuned with DPFE or Simple architectures. Fig. 6 shows the results for the combination of two layers and two attribute sets, with different privacy measures. In all cases, DPFE models have the Acc-Priv superiority over Simple models. In other words, for a fixed value of accuracy, DPFE consistently achieves higher levels of privacy.

3. In order to report the accuracy of an attribute set, we consider the average accuracy of predicting each binary attributes in the set.

Effect of Higher Layers. Comparison of the accuracy-privacy curves of different layers on the same attribute set is depicted in Fig. 7. The results illustrate the Acc-Priv superiority of higher layers for two attribute sets and for both privacy measures. This observation is inline with our earlier assumptions about the higher layers.

Effect of Attribute Set Extension. The accuracy-privacy trade-off of the *DPFE* fine-tuned models for different attribute sets with conv7 as the intermediate layer, are shown in Fig. 8. The results show that as we enlarge the attribute set and restrict the model with preserving the information, then preserving privacy becomes more challenging due to the intrinsic correlation of the identity with facial attributes.

Guaranteeing Privacy. As discussed in Section 5, instead of log-rank, we could also consider the rank itself by analyzing its mean and variance. This idea is depicted in Fig. 9 for Simple and *DPFE* models. The results show that the *DPFE* model has Acc-Priv superiority over the Simple model. More importantly, it forces the conditional distribution of the sensitive variable to converge to an uniform distribution, at least in the rank-mean and standard deviation sense. In fact, the mean and the standard deviation of rank measure for the discrete uniform distribution are 0.5 and 0.28, respectively. As shown in Fig. 9, when privacy increased, the statistics for the *DPFE* model converge to their corresponding values for the uniform distribution. If we consider the normal distribution for the rank variable, we can provide an (ϵ, δ) privacy guarantee, similar to the method used in differential privacy [26]. For example, as depicted in Fig. 9, we can achieve the gender accuracy of up to 90 percent with a rank-mean of 0.3 and standard deviation of 0.25. Hence, with a probability of 0.88 percent we can claim that the rank-privacy is greater than 0.1, and we have achieved 10 percent anonymity.

6.3 Visualization

Visualization is a method for understanding the behavior of deep networks. It provides an insightful intuition about the flow of information through different layers. We used an auto-encoder objective visualization technique [53] to validate the sensitive information removal in DPFE. The reconstruction of images is done by feeding the private-feature to

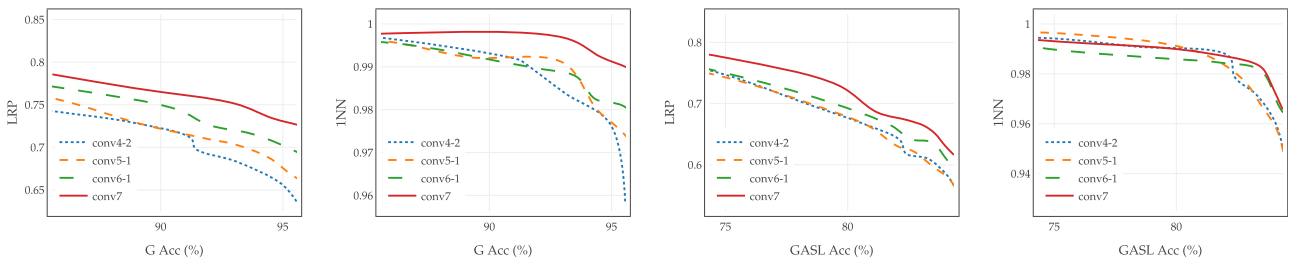


Fig. 7. Layer Comparison: In general, higher layers achieve Acc-Priv superiority to lower layers. In this figure, all models are fine-tuned with the *DPFE* architecture.

the Alexnet decoder proposed in [53]. Therefore, we may *visually* verify the identity removal property of the private-feature by comparing the original and reconstructed images. These images are shown in Fig. 10 for different layers of the original and *DPFE* fine-tuned models.

The results can be analyzed in two aspects: accuracy of desired attributes and privacy of identities. From the privacy perspective, the identity of the people in the reconstructed images of the original model can be readily observed in the last layers (e.g., conv7), while that is not the case for *DPFE* models. Therefore, just relying on the output of higher layers in the original model can not assure acceptable privacy preservation performance, while the *DPFE* models assure the privacy of identities. Regarding the accuracy, we can observe and detect the facial attributes in both models.

6.4 Complexity *versus* Efficiency

Although higher intermediate layers may achieve better accuracy-privacy trade-off, in some cases, such as low-power IoT devices or smartphones, their computational complexity may not be acceptable. Therefore, due to the limited resources on these devices (both memory and computational power) a privacy-complexity trade-off should also be considered. In order to address this problem, we evaluated the original architecture without dimensionality reduction on a smartphone and measured its complexity in different layers. The results are shown in Fig. 11. By gradually reducing the complexity of the private-feature extractor (considering lower intermediate layers in the layer separation mechanism), we also managed to reduce the inference time, memory and CPU usage, while hiding the user's sensitive information.

We evaluated the proposed implementation on a modern handset device, as shown in Table 1. We evaluated the intermediate layers cumulatively, and compared them with the on-premise solution (full model). We used Caffe Mobile v1.0 [57] for Android to load each model and measured the inference time (Fig. 11a) and model memory usage (Fig. 11b) of each of the 17 configurations. We configured the model to only use one core of the device's CPU, as the aim of this experiment was a comparison between the different configurations on a specific device.

Results show a large increase in both inference time and memory use when loading the on-premise solution due to the increased size of the model, proving the efficiency of our solution. More specifically, and by considering the layer conv4_2 as a baseline, we experienced a 14.44 percent inference time and 8.28 percent memory usage increase in conv5_1, 43.96 percent inference time and 22.10 percent memory usage increase in conv6_1, 90.81 percent inference time and 35.05 percent memory usage increase in conv7, and 121.76 percent inference time and 54.91 percent memory usage increase in all layers (on premise). CPU usage also increases per configuration, however due to the multitasking nature of an android device, it is challenging to isolate the CPU usage of a single process and naturally the results fluctuates. Moreover, use of the lower intermediate layers can significantly reduce the complexity of private-feature extractors, especially when dealing with implementing complex deep architectures e.g., *VGG-16* on edge devices and smartphones [59].

Analyzing the complexity of different layers can lead us to considering accuracy-privacy-complexity trade-offs. As an example, consider Fig. 7 and suppose we want to preserve the gender information. Comparing conv7 with conv4-2 and setting the accuracy to 95 percent, we obtain 10 percent more log-rank privacy with the cost of about 90 percent more inference time. In this way we can choose the right strategy based on the importance of accuracy, privacy and complexity. Also by using the dimensionality reduction we can highly decrease the communication cost (compare the size of an image to size of 10 floating point numbers), although in this case we should consider the effect of dimensionality reduction on the complexity which is negligible.

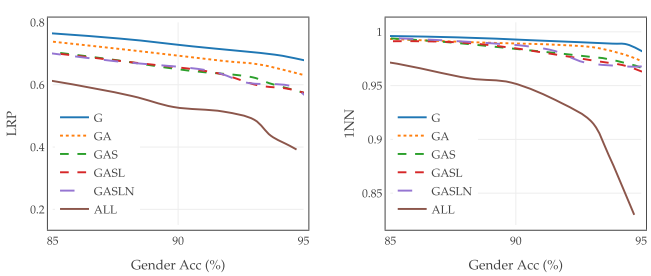


Fig. 8. Comparison of Gender accuracy-privacy trade-offs when putting more preservation constraints on the model. The intermediate layer is set to conv7.

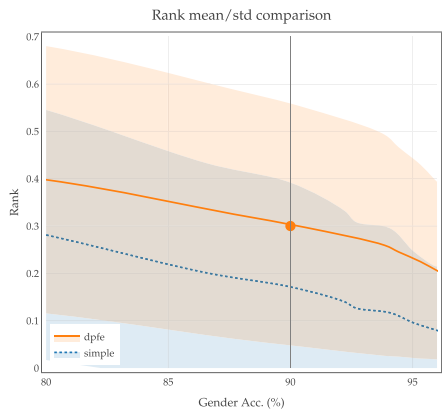


Fig. 9. Comparison of mean and standard deviation of Rank variable for *DPFE* and Simple models for layer conv7.

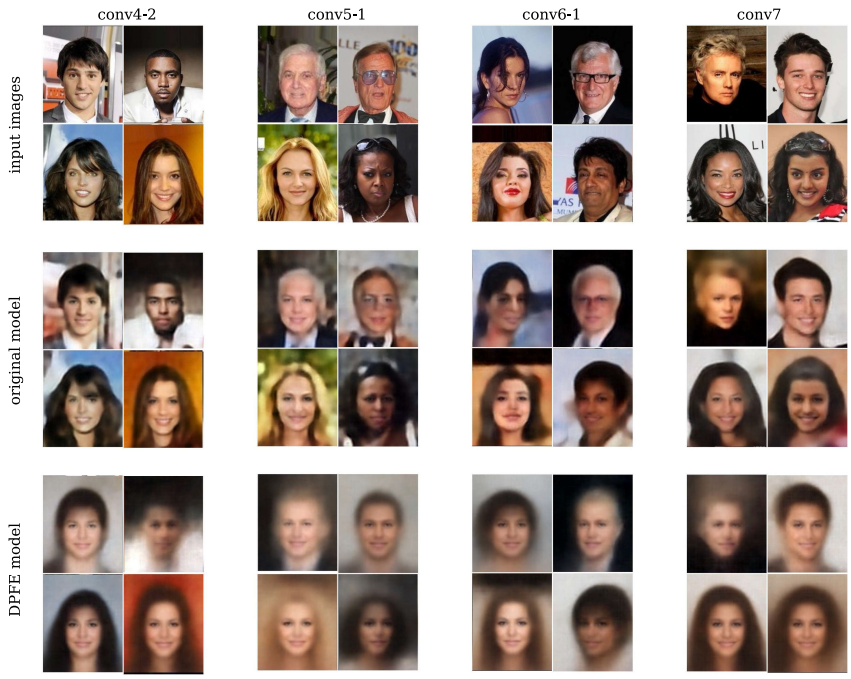
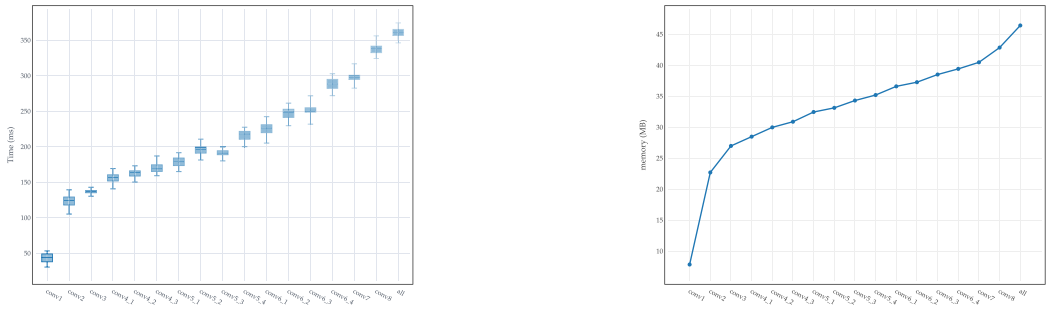


Fig. 10. Visualization of different layers for different models: The top to bottom rows show input images, reconstructed images from original model, and reconstructed images from *DPFE* model. The second row shows that separating layers of a deep model and relying on specificity of higher layers does not provide identity privacy.



(a) Layers time comparison

(b) Layers memory usage comparison

Fig. 11. Comparison of different layers on mobile phone.

We conclude that our algorithm can be implemented on a modern smartphone. By choosing a proper privacy-complexity trade-off and using different intermediate layers, we were able to significantly reduce the cost when running the model on a mobile device, while at the same time preserving important user information from being uploaded to the cloud.

7 Discussion and Future Work

In this paper, we proposed a hybrid framework for privacy preservation of user data. This framework consists of a feature extractor and an analyzer module. The feature extractor provides a user with a private-feature which does not contains the user's desired sensitive information, but still maintains the required information to the service provider, so it can be used by the analyzer module in the cloud. In order to design the feature extractor, we used an information theoretic approach to formulate an optimization problem and proposed a novel deep architecture (*DPFE*) to solve it. To measure quality of the private-feature extractor, we proposed a new privacy measure called *log-rank privacy*. Finally, we considered the problem of facial attribute prediction from face image, and attempted to extract a feature that contains facial attributes information while elimination the identity information. Moreover, by implementing *DPFE* on a mobile phone, we showed that we can achieve a reasonable trade-off between facial attribute prediction accuracy, privacy of identity and computational efficiency.

Our framework suffers from some deficiencies which can be addressed in future works. First, the sensitive variable (e.g., identity) should be discrete. Continuous sensitive variables may be considered in future works and differential privacy might be useful in this context. Second, the current

TABLE 1
Device Specification

Google (Huawei) Nexus 6P	
Memory	3 GB LPDDR4 RAM
Storage	32 GB
CPU	Octa-core Snapdragon 810 v2.1
GPU	Adreno 430
OS	Android 7.1.2

version of the deep architecture is limited to CNNs and feed forward networks. Considering time series data and extending the framework to other deep architectures such as recurrent neural networks might pose new challenges which can be addressed in future works. Moreover, reducing the dimensionality and choosing the optimal dimension is hard and can be analyzed in the future. It is also interesting to analyze the log-rank privacy measure which can also have many potential applications in the privacy domain. Another interesting future direction is to exploit the log-rank privacy in design of learning to rank algorithms.

Acknowledgments

The authors acknowledge constructive feedback from David Meyer, Hassan Hafez, Sina Sajadmanesh, and Amirhossein Nazem. Ali Shahin Shamsabadi, Kleomenis Katevas, and Hamed Haddadi were supported by the EPSRC Databox grant (Ref: EP/N028260/1) and a Microsoft Azure for Research grant (CRM:0740917).

References

[1] A. Acquisti, L. Brandimarte, and G. Loewenstein, "Privacy and human behavior in the age of information," *Sci.*, vol. 347, no. 6221, pp. 509–514, 2015.

[2] N. Vallina-Rodriguez, J. Shah, A. Finamore, Y. Grunenberger, K. Papagiannaki, H. Haddadi, and J. Crowcroft, "Breaking for commercials: Characterizing mobile advertising," in *Proc. Internet Meas. Conf.*, 2012, pp. 343–356.

[3] M. Haris, H. Haddadi, and P. Hui, "Privacy leakage in mobile computing: Tools, methods, and characteristics," arXiv:1410.4978, Oct. 2014.

[4] H. Haddadi and I. Brown, "Quantified self and the privacy challenge," *Technol. Law Futures*, pp. 1–2, 2014.

[5] F. D. Garcia and B. Jacobs, "Privacy-friendly energy-metering via homomorphic encryption," in *Proc. Int. Workshop Security Trust Manag.*, 2010, pp. 226–238.

[6] C. Fontaine and F. Galand, "A survey of homomorphic encryption for nonspecialists," *EURASIP J. Inf. Security*, vol. 2007, no. 1, 2007, Art. no. 013801.

[7] P. Garcia Lopez, A. Montresor, D. Epema, A. Datta, T. Higashino, A. Iamnitchi, M. Barcellos, P. Felber, and E. Riviere, "Edge-centric computing: Vision and challenges," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 45, no. 5, pp. 37–42, 2015.

[8] S. A. Osia, A. S. Shamsabadi, A. Taheri, H. R. Rabiee, N. Lane, and H. Haddadi, "A hybrid deep learning architecture for privacy-preserving mobile analytics," arXiv:1703.02952, Mar. 2017.

[9] S. A. Osia, A. S. Shamsabadi, A. Taheri, H. R. Rabiee, and H. Haddadi, "Private and scalable personal data analytics using a hybrid edge-cloud deep learning," *IEEE Comput.*, vol. 51, no. 5, pp. 42–49, May 2018.

[10] R. Agrawal and R. Srikant, "Privacy-preserving data mining," *ACM SIGMOD Rec.*, vol. 29, no. 2, 2000, pp. 439–450.

[11] D. Agrawal and C. C. Aggarwal, "On the design and quantification of privacy preserving data mining algorithms," in *ACM Symp. Principles Database Syst.*, 2001, pp. 247–255.

[12] C. C. Aggarwal and S. Y. Philip, "A general survey of privacy-preserving data mining models and algorithms," in *Proc. Privacy-Preserving Data Mining*, 2008, pp. 11–52.

[13] L. Sweeney, "k-anonymity: A model for protecting privacy," *Int. J. Uncertainty, Fuzziness Knowl.-Based Syst.*, vol. 10, no. 05, pp. 557–570, 2002.

[14] B. C. Fung, K. Wang, and P. S. Yu, "Top-down specialization for information and privacy preservation," in *Proc. IEEE Int. Conf. Data Eng.*, 2005, pp. 205–216.

[15] K. Wang, P. S. Yu, and S. Chakraborty, "Bottom-up generalization: A data mining solution to privacy protection," in *Proc. IEEE Int. Conf. Data Mining*, 2004, pp. 249–256.

[16] R. J. Bayardo and R. Agrawal, "Data privacy through optimal k-anonymization," in *Proc. IEEE Int. Conf. Data Eng.*, 2005, pp. 217–228.

[17] K. LeFevre, D. J. DeWitt, and R. Ramakrishnan, "Mondrian multidimensional k-anonymity," in *Proc. IEEE Int. Conf. Data Eng.*, 2006, pp. 25–25.

[18] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam, "L-diversity: Privacy beyond K-anonymity," *ACM Trans. Knowl. Discovery Data*, vol. 1, no. 1, 2007, Art. no. 3.

[19] N. Li, T. Li, and S. Venkatasubramanian, "T-closeness: Privacy beyond k-anonymity and l-diversity," in *Proc. IEEE Int. Conf. Data Eng.*, 2007, pp. 106–115.

[20] D. Rebollo-Monedero, J. Forne, and J. Domingo-Ferrer, "From t-closeness-like privacy to postrandomization via information theory," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 11, pp. 1623–1636, Nov. 2010.

[21] E. M. Newton, L. Sweeney, and B. Malin, "Preserving privacy by de-identifying face images," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 2, pp. 232–243, Feb. 2005.

[22] R. Shokri and V. Shmatikov, "Privacy-preserving deep learning," in *Proc. ACM Conf. Comput. Commun. Security*, 2015, pp. 1310–1321.

[23] N. Papernot, M. Abadi, U. Erlingsson, I. Goodfellow, and K. Talwar, "Semi-supervised knowledge transfer for deep learning from private training data," in *Proc. Int. Conf. Learn. Representations*, 2017.

[24] B. Hitaj, G. Ateniese, and F. Pérez-Cruz, "Deep models under the GAN: Information leakage from collaborative deep learning," in *Proc. ACM SIGSAC Conf. Comput. Commun. Security*, 2017, pp. 603–618.

[25] P. Mohassel and Y. Zhang, "Secureml: A system for scalable privacy-preserving machine learning," in *Proc. IEEE Symp. Security Privacy*, 2017, pp. 19–38.

[26] C. Dwork, "Differential privacy," in *Proc. Int. Conf. Automata Languages Program.*, 2006, pp. 1–12.

[27] C. Dwork, "Differential privacy: A survey of results," in *Proc. Int. Conf. Theory Appl. Models Comput.*, 2008, pp. 1–19.

[28] Z. Ji, Z. C. Lipton, and C. Elkan, "Differential privacy and machine learning: A survey and review," arXiv:1412.7584, Dec. 2014.

[29] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *Proc. ACM SIGSAC Conf. Comput. Commun. Security*, 2016, pp. 308–318.

[30] F. Tramèr, F. Zhang, A. Juels, M. K. Reiter, and T. Ristenpart, "Stealing machine learning models via prediction APIs," in *Proc. USENIX Security Symp.*, 2016, pp. 601–618.

[31] R. Gilad-Bachrach, N. Dowlin, K. Laine, K. Lauter, M. Naehrig, and J. Wernsing, "Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 201–210.

[32] B. D. Rouhani, M. S. Riazi, and F. Koushanfar, "Deepsecure: Scalable provably-secure deep learning," arXiv:1705.08963, pp. 1–6, 2018.

[33] C. Juvekar, V. Vaikuntanathan, and A. Chandrakasan, "Gazelle: A low latency framework for secure neural network inference," arXiv:1801.05507, Jan. 2018.

[34] H. Dong, C. Wu, Z. Wei, and Y. Guo, "Dropping activation outputs with localized first-layer deep network for enhancing user privacy and data security," *IEEE Trans. Inf. Forensics Security*, vol. 13, no. 3, pp. 662–670, Mar. 2018.

[35] C. Liu, S. Chakraborty, and P. Mittal, "Deeprotect: Enabling inference-based access control on mobile sensing applications," arXiv:1702.06159, Feb. 2017.

[36] M. Malekzadeh, R. G. Clegg, and H. Haddadi, "Replacement autoencoder: A privacy-preserving algorithm for sensory data analysis," in *Proc. 3rd ACM/IEEE Int. Conf. Internet-of-Things Des. Implementation*, 2018, pp. 165–176.

[37] A. J. Bell and T. J. Sejnowski, "An information-maximization approach to blind separation and blind deconvolution," *Neural Comput.*, vol. 7, no. 6, pp. 1129–1159, 1995.

[38] D. Barber and F. Agakov, "The IM algorithm: A variational approach to information maximization," in *Proc. 16th Int. Conf. Neural Inf. Process. Syst.*, 2003, pp. 201–208.

[39] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel, "Infogan: Interpretable representation learning by information maximizing generative adversarial nets," in *Proc. Neural Inf. Process. Syst.*, 2016, pp. 2172–2180.

[40] N. Tishby, F. Pereira, and W. Bialek, "The information bottleneck method," in *Proc. 37th Annu. Allerton Conf. Commun. Control Comput.*, 1999, pp. 368–377.

[41] A. A. Alemi, I. Fischer, J. V. Dillon, and K. Murphy, "Deep variational information bottleneck," arXiv:1612.00410, Dec. 2016.

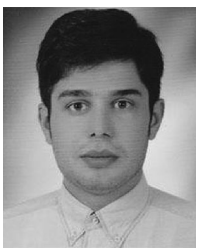
[42] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Secaucus, NJ, USA: Springer, 2006.

[43] T. Duong and M. L. Hazelton, "Convergence rates for unconstrained bandwidth matrix selectors in multivariate kernel density estimation," *J. Multivariate Anal.*, vol. 93, no. 2, pp. 417–433, 2005.

[44] B. W. Silverman, *Density Estimation for Statistics and Data Analysis*. Boca Raton, FL, USA: CRC Press, 1986, vol. 26.
[45] Y. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller, "Efficient backprop," in *Neural Networks: Tricks of the Trade*. Berlin, Germany: Springer, 1998, pp. 9–50.
[46] J. R. Hershey and P. A. Olsen, "Approximating the kullback leibler divergence between gaussian mixture models," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2007, pp. IV-317–IV-320.
[47] R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality reduction by learning an invariant mapping," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2006, pp. 1735–1742.
[48] S. Chopra, R. Hadsell, and Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2005, pp. 539–546.
[49] S. A. Osia, A. S. Shamsabadi, A. Taheri, K. Katevas, H. R. Rabiee, N. D. Lane, and H. Haddadi, "Privacy-preserving deep inference for rich user data on the cloud," arXiv:1710.01727, 2017.
[50] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 448–456.
[51] R. Shwartz-Ziv and N. Tishby, "Opening the black box of deep neural networks via information," arXiv:1703.00810, Mar. 2017.
[52] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" in *Proc. Neural Inf. Process. Syst.*, 2014, pp. 3320–3328.
[53] A. Dosovitskiy and T. Brox, "Inverting visual representations with convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 4829–4837.
[54] T. M. Cover and J. A. Thomas, *Elements Inf. Theory*. Hoboken, NJ, USA: Wiley, 2012.
[55] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *Proc. Int. Conf. Comput. Vis.*, 2015, pp. 3730–3738.
[56] E. M. Rudd, M. Günther, and T. E. Boult, "Moon: A mixed objective optimization network for the recognition of facial attributes," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 19–35.
[57] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," arXiv:1408.5093, pp. 675–678, 2014.
[58] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," arXiv:1412.6980, Dec. 2014.
[59] Y.-D. Kim, E. Park, S. Yoo, T. Choi, L. Yang, and D. Shin, "Compression of deep convolutional neural networks for fast and low power mobile applications," arXiv:1511.06530, Nov. 2015.

Seyed Ali Osia received the BSc degree in software engineering from the Sharif University of Technology, in 2014. He is currently working toward the PhD degree at the Department of Computer Engineering, Sharif University of Technology. His research interests includes statistical machine learning, deep learning, privacy, and computer vision.

Ali Taheri received the BSc degree in software engineering from Shahid Beheshti University, in 2015, and the MSc degree in artificial intelligence from the Sharif University of Technology, in 2017. His research interests includes deep learning and privacy.



Ali Shahin Shamsabadi received the BS degree in electrical engineering from the Shiraz University of Technology, in 2014, and the MSc degree in electrical engineering (digital) from the Sharif University of Technology, in 2016. Currently, he is working toward the PhD degree at the Queen Mary University of London. His research interests include deep learning and data privacy protection in distributed and centralized learning.

Kleomenis Katevas received the BSc degree in informatics engineering from the University of Applied Sciences of Thessaloniki, and the MSc and PhD degrees from the Queen Mary University of London. He is currently a postdoctoral researcher with the Imperial College of London. His research interests includes Mobile & Ubiquitous Computing, Applied Machine Learning, Crowd Sensing and Human-Computer Interaction.

Hamed Haddadi received the BEng, MSc, and PhD degrees from University College London. He was a postdoctoral researcher at the Max Planck Institute for Software Systems in Germany, and a postdoctoral research fellow at the Department of Pharmacology, University of Cambridge, and The Royal Veterinary College, University of London, followed by few years as a lecturer and consequently senior lecturer in digital media at the Queen Mary University of London. He is currently a senior lecturer (associate professor) and the deputy director of research in the Dyson School of Design Engineering, and an academic fellow of the Data Science Institute, at The Faculty of Engineering at Imperial College of London. He is interested in user-centered systems, IoT, applied machine learning, and data security & privacy.

Hamid R. Rabiee (SM '07) received the BS and MS degrees (with great distinction) in electrical engineering from CSULB, Long Beach, CA, in 1987 and 1989, the EEE degree in electrical and computer engineering from USC, Los Angeles, CA, in 1993, and the PhD degree in electrical and computer engineering from Purdue University, West Lafayette, IN, in 1996. From 1993 to 1996, he was a member of technical staff at AT&T Bell Laboratories. From 1996 to 1999, he worked as a senior software engineer at Intel Corporation. He was also with PSU, OGI, and OSU universities as an adjunct professor of electrical and computer engineering from 1996-2000. In September 2000, he joined the Sharif University of Technology, Tehran, Iran. He was also a visiting professor at the Imperial College of London for the 2017-2018 academic year. He is the founder of the Sharif University Advanced Information and Communication Technology Research Institute (AICT), ICT Innovation Center, Advanced Technologies Incubator (SATI), Digital Media Laboratory (DML), Mobile Value Added Services Laboratory (VASL), Bioinformatics and Computational Biology Laboratory (BCB) and Cognitive Neuroengineering Research Center. He has also been the founder of many successful High-Tech start-up companies in the field of ICT as an entrepreneur. He is currently a professor of computer engineering at the Sharif University of Technology, and director of AICT, DML, and VASL. He has been the initiator and director of many national and international level projects in the context of the Iran National ICT Development Plan and the UNDP International Open Source Network (IOSN). He has received numerous awards and honors for his industrial, scientific, and academic contributions. He has acted as chairman in a number of national and international conferences, and holds three patents. He is also a member of the IFIP Working Group 10.3 on Concurrent Systems, and a senior member of the IEEE. His research interests include statistical machine learning, Bayesian statistics, data analytics and complex networks with applications in complex networks, multimedia systems, cloud and IoT privacy, bioinformatics, and brain networks.