Computer Communications 182 (2022) 117–128

Contents lists available at ScienceDirect

Computer Communications

journal homepage: www.elsevier.com/locate/comcom

A differential privacy-based classification system for edge computing in IoT

Wanli Xue [a], Yiran Shen [b,*], Chengwen Luo [d], Weitao Xu [c], Wen Hu [a], Aruna Seneviratne [a]

[a] University of New South Wales, Australia
[b] Shandong University, China
[c] City University of Hong Kong, Shenzhen Research Institute, China
[d] Shenzhen University, China

ARTICLE INFO

Keywords:
Edge-based classification
Privacy-preserving
Internet of Things
Decentralized
Attack the edge

ABSTRACT

The blooming Internet of Things (IoTs) devices have brought many new types of sensing applications and methods to the traditional cloud-enabled IoTs framework. Hence, the new network framework becomes bidirectional gradually, in which IoTs devices can also perform moderate computation tasks instead of being solely used as data harvesters. This new coming framework known as edge computing significantly improves the traditional cloud-enabled network latency and dependency by shifting part of computation back to "local". However, new security risks emerge when the edge computing shifts data and models back to the IoTs devices.

Acies, a differential privacy based privacy-preserving classification system for edge computing is proposed to secure the classification models offloaded to edge devices. Acies supports popular classifiers such as Nearest Neighborhood, Support Vector Machine and Sparse Representation Classifier with a variety of feature selection methods. According to our evaluation on different datasets, classification models with Acies can be private and maintain high utility. Acies achieves reliable privacy protection under reconstruction attacks with minimal impact on classification accuracy (2% $\backsim$ 5%) only. Acies outperforms the naive input dataset perturbation methods by up to 30% higher classification accuracy when the privacy requirements of the applications is high (the privacy budget ϵ is less than 2).

1. Introduction

With the pervasive implementation of Internet of Things (IoTs) technologies and growing performance of the host end devices, we envision the new generation of IoTs architecture will get relief from high dependency on availability of cloud resources, and the end devices will not only work as data collectors but also an edge platform for some sorts of computing. The report of Cisco Systems Inc. claims that there will be about 50 billion devices connected to the Internet by 2020[1]. The vast implementation of IoTs has brought new sensing capabilities and methodologies along with conventional cloud-enabled framework by shifting 'storage and process' to the cloud. It has substantially changed the way people arrange the information about themselves and the surrounding environment.

Edge computing framework has been proposed [2] and applied in a number of real-world applications like authenticating query [3]. Comparing with centralized process on cloud, edge computing pushes application logic and the underlying data much closer to where the data is generated to the system latency, availability and scalability. Edge computing can relieve the network dependency for real time applications, for instance, the users authentication. Besides, the carefully tuned trade-off between computing and communication responsibilities between edge servers, trusted servers and untrusted services can enlarge the fault-tolerance, churn, elasticity and many others scale to millions of users, which is more appropriate for current IoTs environment. However, this central-decentral offloading places new potential privacy risks therefore it forces to enhance the trust, privacy and autonomy requirements.

Edge computing greatly helps fill in the large gap by providing "low-latency offloading". In the conventional "cloud" circumstances, network latency problem has already been vastly noticed [4,5]. As the real world observations on Amazon EC2 service from different campuses, the network latency can be varying from 20 to 270 ms. Though the server-grade computing platforms support more complex algorithms, the network latency makes applications lean to a better design like *close at hand* [4]. On the contrary, offloading gives almost an order of magnitude improvement on overall response time and energy efficiency of the low-cost devices like cognitive assistance on wearable

* Corresponding author.
E-mail address: yiran.shen@sdu.edu.cn (Y. Shen).

https://doi.org/10.1016/j.comcom.2021.10.038
Received 20 February 2020; Received in revised form 31 January 2021; Accepted 23 October 2021
Available online 8 November 2021

glasses [5], by reducing network latency as well as communication bandwidth.

The privacy risk of edge computing. Technology advances such as dedicated connection boxes deployed in most homes, high capacity mobile end-user devices and powerful wireless networks are always coupled closely with concerns on trust, privacy, and autonomy. Edge computing pulls controlling of applications, users' data, and services away from central nodes (the "core") and closer to the other logical extreme (the "edge") of the Internet. Those out-of-range controls make significant contribution to the efficiency of edge computing, however, they also create new system security threats.

Privacy-preserving solutions. In this paper, we focus on the privacy leakage and the corresponding solutions for Machine Learning (ML) models. The privacy-preserving solutions designed for cloud computing cannot be directly migrated into edge computing scenarios due to their difference in network paradigm, i.e., the introduction of local (edge) servers. The existing solutions for privacy-preserving in cloud computing considered the data owner outsources computing capability for training ML model on professional cloud service providers without revealing the privacy of data. However, considering the real deployment circumstances of edge computing services, the edge servers are sometimes not trusted. The threat from malicious users has not been well addressed as it aims for more complicated scenarios. Most of recent solutions, such as in [6–8], only considered privacy-preserving training of ML classifiers for single user cases using common cloud services. Besides, some of them [6–10] were bespoken for some specific classifiers, e.g., Support Vector Machine (SVM) classifier. In addition, traditional solutions [10] to preserve the privacy of model is to inject random noises to perturb original training data, however, this strategy may lead to dramatic performance drop on the classification accuracy.

In this paper, we propose Acies,[1] to facilitate differential privacy theory in an innovate approach to effectively address generic ML privacy problems in the scope of edge computing. **Contribution.** The contributions of this paper are as followed:

- We point out and formulate the new privacy issue of machine learning in edge computing. A new threat model is proposed to analyze the issues brought by the new computational paradigm.
- A differential privacy based mechanism Acies is designed to preserve the privacy of the classifiers. It is orthogonal to the choice of classifiers, such as k-Nearest Neighborhoods (KNN), SVM, Sparse Representation Classification (SRC), and feature selection methods, such as Eigenface, Fisherface, and Singular Value Decomposition (SVD).
- Acies facilitates differential privacy protection in the feature selection stage and can be smoothly incorporated in most of the current classification services without any changes on classification work flow and system architecture.
- According to our extensive evaluation on multiple datasets of different classification tasks, Acies achieves reliable privacy protection against reconstruction attacks with only minimal impact on classification accuracy (2% – 5%) which, is significant (up to 30%) lower than naive approaches [10] by taking noise into account in the ML model training stage instead of adding the noise to the input data source directly.
- At last, we consider a new form of attack to the machine learning models running on the edge, termed as *accumulation attack*. We point out and prove that the training dictionary can be recovered when new users join in the system and propose the countermeasure to the accumulation attack basing on differential privacy.

Partial and preliminary results of this paper have appeared in our previous work [11]. Comparing to the conference version, this paper contains significant new contributions which are listed as follows.

[1] Acies is a Latin word origin as shape edge and vision.

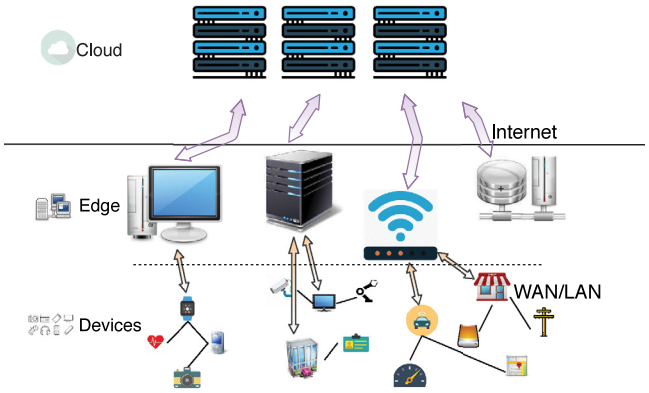


Fig. 1. Overview of edge computing.

- We detail and illustrate threat models with realistic attack examples in Sections 2.3 and 2.4.
- We extensively evaluate the new method with another two more public datasets from human activity recognition (HAR) and hand written digits (MNIST) in Section 4.2. We add one more measurement (RAM cost) to evaluate and discuss the potential overhead of the different classifier combination with all datasets (see Section 4.4).
- Figs. 9, 10, 11, 13 and Table 2 are new plots and tables used to illustrate and discuss new evaluation results. Figs. 1, 2, 3, 6 and equations are replenished to better illustrate the content.
- We propose a new attack model, accumulation attacks, and its corresponding protection solution based on differentially private in Section 5, which was not touched in the previously published work [11].

The rest of the paper is organized as following. Section 2 presents the background and privacy concerns of edge-based classifications. We also describe our proposed threat model with reconstruction attack sequentially. We present the proposed privacy-preserving ML scheme Acies in Section 3. Evaluations of Acies are presented in Section 4. We discuss a new potential and corresponding countermeasures in Section 5. Section 6 discusses related work and Section 7 concludes the paper.

2. Edge-based classification

2.1. Overview

Edge computing is also known as fog computing [12–16]. Fig. 1 is an overview of edge computing. Edge computing allows us to process data near the source and only send few results over the network to an intermediate data processor, which can address unreliable latency problem in traditional cloud-based computing.

2.2. Characteristics and constraints

There are couple of characteristics and constraints we need to consider when design an edge computing systems.

Dimension reduction. Training samples like features extracted from raw sensor recordings are used to train a robust classifier. IoTs devices (edge nodes) undertake some lightweight computing locally (we focus on classification tasks in this paper) to avoid latency caused by remote communication or network connection interruption. However, considering the fact that IoTs devices are resource-constrained, the complexity of trained ML models should be reduced through dimensionality reduction before being deployed on edge nodes. However, dimensionality reduction method should be carefully designed so that

the network latency can be addressed without noticeable sacrifice on classification accuracy.

Feature extraction is the first step of many ML applications, in which a vector of features are usually transformed from the raw sensor data (an image, an audio clip, gyro-accelerometer values or other sensor records). We take image classification as an example: image classification is computational prohibitive when deployed in edge computing environment. To enable the task efficient on resource limited IoTs devices, numerous feature extraction schemes have been investigated, such as Eigenfaces [17], Fisherfaces [18] and Singular Value Decomposition (SVD), to find the best representation that separates the classes in lower dimensional feature space. It is well known that the choice of features is critical to the traditional classifiers like kNN and SVM. This has led to the development of a wide variety of increasingly complex feature extraction methods, including nonlinear and kernel approaches.

The benefits of feature extraction are to save the computational cost and on-board storage with reduced dimensionality which helps those edge nodes meet the basic resource requirements [19]. For example, when using high-resolution face images, the corresponding ML models require high computational capability and large storage space. Specifically, a face image with typical resolution may contain over 300 K pixels (640 × 480). Although some algorithms rely on scalable methods such as linear programming, directly applying it to such high-resolution images is still beyond the capability of regular embedded computers. Appropriate feature dimension reduction can improve the classification efficiency without sacrificing too much recognition accuracy. The readers are encouraged to refer to [19] for more details.

Kernel based classifiers. In this paper, we focus on the classifiers that are widely adopted in IoTs applications such as kNN, SVM and SRC [10,19–21]. All those classifiers can be categorized as Kernel Logistic Regression models (KLR) [22]. In KLR models, the kernel function retains (a tiny fraction of) the training data (termed as "import points") which may result in the leakage of privacy of the training data.

2.3. Privacy concerns and threat model

Privacy concerns. Designing a system for resource-limited edge-based IoT environment sounds conceptually straightforward, but faces many underlying threats. For example, ML models are typically trained in a central cloud server before being offloaded to edge nodes. These edge nodes are owned by individual users who might be curious about the information in the models, or might not have financial and/or cybersecurity resources like the owner of central cloud server to safeguard the nodes against adversaries . Adversaries will have more chance to acquire useful information via various attack methods on the "less secure" edge servers such as data reconstruction attack [23]. Those reconstructed data can be identity-related, health status, GPS trajectory and so on, which can be further used for commercial purpose or endanger public safety.

Threat model. Different from traditional cloud computing threat model, we assume that the cloud server is secure while the edge (including edge servers and edge devices) is untrusted because the edge is located closer to the clients and difficult to be safeguarded physically like the Cloud. Furthermore, edge devices such as smartphones and smart home gateways are typically managed by clients without sophisticated cybersecurity knowledge to provide universal accessibilities, which makes them significantly easier to be compromised by the hackers. The overview of threat model and Acies is shown in Fig. 2.

In our threat model, a malicious user may obtain other users' private information through the networks between the edge server and the edge device. For example, Eve can access to the face authentication model placed in their building's edge server and may be interested in other residents' appearance. This is because among most of those kernel-based classification systems, there are still much private information remained in the model which can be obtained by the attacker via other machine learning tools [24]. Bring back to the

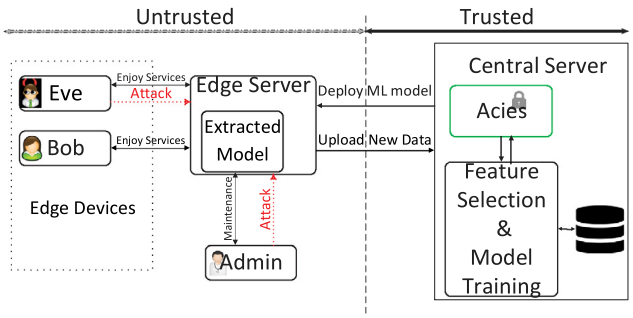


Fig. 2. Threat model and overview of Acies.

edge-computing scenario, it is widely accepted that this particularity of edge paradigms is a double-edged sword: on the one hand, it limits the impact of an attack to the local environment since there existing a 'command and control center' (the central server) which is apart from the geo-location. On the other hand, if one adversary can control one edge data center, it might be able to control almost all the services that are provided in that geographical location [25]. In addition, traditional encryption-based methods like secure multi-party encryptions are addressing this by using an encrypted channel between the edge devices and the servers, such that the attacker cannot access the classification model in plaintext. However, the edge devices might not meet the system requirements (e.g., CPU, RAM, etc.) for those encryptions and the high-power consumption is also not edge friendly. And in the worst case, the edge devices might be compromised before the key dissemination stage, thus the attacker can easily obtain the classification model and distil the private information. We assume the classification model is trained on central server with tailored ML algorithms (see Fig. 2). In order to provide better system performance such as response time, the ML models are offloaded to edge servers and/or devices.

Case study: smart building face recognition. Considering the face recognition based authentication system for smart building [26], to avoid the lengthy latency and temporally disabled connection, edge computing is more appropriate for authentication process [3]. Users' face images are uploaded and stored in central server. Then the Cloud (central server) trains the relative ML models for identification/authentication using uploaded face images. Daily face identification/authentication tasks will be offloaded to the edge nodes (local servers), which obtain trained ML models from the Cloud.

In this case, edge computing provides better system response time without the requirement of always-on Internet connection, but new risks emerge. Since the edge node deployed locally, there is potential threat from malicious attackers who are curious about the training data, i.e., the faces of users. The attacker could be a hacker who takes control of local edge nodes (which typically have lower security protection levels than the Cloud), or any common user (the owner of one of the edge nodes) who is just curious about how other users look like.

2.4. Reconstruction attack

To better illustrate the privacy issues of the edge computing based classification system we conduct some preliminary experiments by launching reconstruction attacks on those edge-based classifiers. We again take the face recognition case as an example.

When training the ML models, a central server extracts the features such as Eigenface, Fisherface or SVD vectors (all are available in the OpenCV library) from training data (raw face images). Then, it combines the features with a ML model before offloading them to edge nodes so that the edge nodes can use the ML model to obtain the

Fig. 3. Reconstruction attack on SVD, Eigenface, and Fisherface feature extraction methods and raw image as comparison (from left to right).

classification results based on a new input (image). The classification task can be formulated as,

$$y = f(x) \tag{1}$$

where $f(\cdot)$ represents the model or the classification function, x is the input data and y is the output. After the feature calculation, Eq. (1) becomes:

$$y = (R \cdot f)(R \cdot x) \tag{2}$$

where R is the feature extraction matrix.

By storing x in matrix $X \in \mathbb{R}$, each column of X represents one data record (with dimension n), and the m is the total number training data instance. If we have 38 people in total and capture 32 face images each person, we will produce an X that has $38 \times 32 = 1{,}216$ columns. A raw face image can have 32,256 pixels (extended YaleB face dataset is 192×168 image resolution), and the dimension of X become $32{,}256 \times 1216$. With the appropriate feature selection algorithms (which also typically reduces the dimension of X), the training data dimension can decrease to k (e.g., 300), and $k \ll n$. Thus, the compressed X (300×1216) is then used to train ML models.

Since extracted model will be offloaded to edge devices as discussed earlier in Section 2.3 and it contains the information of sensitive raw training data instance ($A = R \cdot X$) in KLR models such as kNN, SVM and SRC, one can launch an attack to reconstruct sensitive raw training data instance by solving the following equation to reconstruct the raw training data X from the compressed/remaining information A in the kernel:

$$\hat{X} = \arg\min_{X} \|A - R\hat{X}\|_2^2 + \theta, \tag{3}$$

where A equals to $R \cdot X$ in Eq. (2), and is the information leakage from KLR models, $\hat{X}$ is the solution (the estimation of sensitive raw training data instance), and θ is an unknown error vector.

Fig. 3 shows the attack results for popular feature selection algorithms such as SVD, Eigenface and Fisherface. This reconstruction attack is similar to those privacy attacks reported in the literature [23,27,28]. For the benefit of space, we omit the technical details.

According to our results, popular features for face recognition such as Eigenface, Fisherface, and SVD (all available in OpenCV) are vulnerable even though the dimensions of the features (e.g. k=300) are orders-of-magnitude smaller than the raw image resolution (e.g., n=32,256). This creates new risks for edge computing architecture where ML models are offloaded to the edge from the Cloud.

Apart from face recognition authentication systems, an adversary can launch similar reconstruction attacks to other edge computing applications such as activity recognition. For instance, embedded accelerometer and gyroscope data in smartphones can be used for activities recognition [29–31]; gait information extracted from WiFi signal like Channel State Information (CSI) can be used for human identification [32,33]. We will use these IoTs application datasets to evaluate the proposed algorithm later in Section 4.

3. Protections for edge-based classification

3.1. Adopting differential privacy in edge-based classifiers

The idea of differential privacy [34,35] has been popularly used in many countermeasures to address the issue of information leakage of a publicly released dataset. In this paper, we propose to perturb the extracted classification models which may be deployed on untrusted platforms rather than injecting noises to the raw training data directly.

In this paper, we define the new type of differential privacy for extracted classification model: the feature re-compress/re-extract methods are a set of matrices, and we say that two of these matrices are differentially private if they satisfy below definition. The statement of the definition is based on its counterpart on dataset and written as follows:

Definition 1 (*ϵ-differentially Private Mechanism*)**.** A randomized function K gives ϵ - differential privacy if for all data sets D and D' differing on at most one row, and all $S \subseteq Range(K)$

$$Pr[K(D) \in S] \leq \exp(\epsilon) \times Pr[K(D') \in S], \tag{4}$$

where the probability space in each case is over the coin flips of K. Smaller ϵ yields a stronger privacy guarantee.

Differential privacy has several properties that make it particularly useful in applications such as classifications composability, group privacy and robustness to auxiliary information. Composability enables modular design of mechanisms: if all components of a mechanism are differentially private, then so is their composition. Group privacy implies graceful degradation of privacy guarantees if datasets contain correlated inputs, such as the ones contributed by the same individual. Robustness to auxiliary information means that privacy guarantees are not affected by any side information available to the adversary.

A common paradigm for approximating a deterministic real-valued function $P : D \rightarrow \mathbb{R}^d$ with a differentially private mechanism is via additive noise calibrated to P 's *sensitivity* S_P, which is defined as the maximum of the absolute distance $|P(d) - P(d')|$ where d and d' are adjacent inputs.

Definition 2 (*Global Sensitivity*)**.** For the given query function $P : D \rightarrow \mathbb{R}^d$, the global *sensitivity* of P is

$$\Delta P = \max_{n,v} |P(n) - P(v)|, \tag{5}$$

for $\forall n, v$ differing in at most one element.

Let $Lap(\lambda)$ be the Laplace distribution whose density function is $h(n) = \frac{1}{2\lambda} \exp(-\frac{|N-\mu|}{\lambda})$ where μ is a mean and $\lambda > 0$ is a scale factor. Dwork et al. [35] proved that, for the given query function P and a database N, a randomized mechanism M_P that returns $P(N)+Y$ where Y is drawn identical and independent distribution from $Lap(\frac{\Delta P}{\epsilon})$, gives ϵ-differential privacy [36].

3.2. Input data perturbation

Inspired by differential privacy, a naive approach (as shown in Algorithm 1) can be applied to protect privacy by injecting random noises into classifiers' training dataset. The algorithm starts by initializing the sensitivity (Line 3 in Algorithm 1) then the noise vector are computed and combined with the chosen target privacy budget (Line 5). Selecting a reasonable global sensitivity itself is another research question which is beyond the scope of this paper. We adopted the naive approach from the original definition (Definition 2). Readers can refer to [37] for sensitivity methods like $\ell_2 - sensitivity$ for vector-valued functions. In face recognition case, the sensitivity is 255. According to the parallel composability of differential privacy [38], the entire dataset X is differentially private after processed by the algorithm. SVD is used as a feature extraction example in the algorithm, though other classifiers and feature extraction methods, such as Eigenface and Fisherface, are also applicable here. The output of this algorithm, i.e., A, then can be used for any ML classifiers as the training set.

Fig. 4 presents examples of reconstruction attack results on different feature extraction methods (from top to bottom: Elgenface, Fisherface, SVD) under different rates of noise (ϵ={8, 5, 2, ln2}) that are added

ALGORITHM 1: Input Data Perturbation

Input: Normal input data/feature $X = \{x_1, x_2, x_3, ..., x_n\} \in \mathbb{R}^{n \times m}$, target privacy budget ϵ, extraction ratio k.
Output: Perturbed extracted/compressed feature set A.
Calculate sensitivity $s = \max X - \min X$;
for *each* $x_i \in X$ **do**
  Sample noise vector lp from Laplace(s/ϵ);
  $x_i' = x_i + lp$;
end
Calculate the Singular Value Decomposition (SVD) of perturbed features transpose $(X')^T$: $(X')^T = U \Lambda V^T$;
Choose the first k column of U as the extraction matrix $R \in \mathbb{R}^{k \times n}$;
Get $A \in \mathbb{R}^{k \times m}$ by multiply the extraction matrix transpose with perturbed input $A = R^T \times X'$;

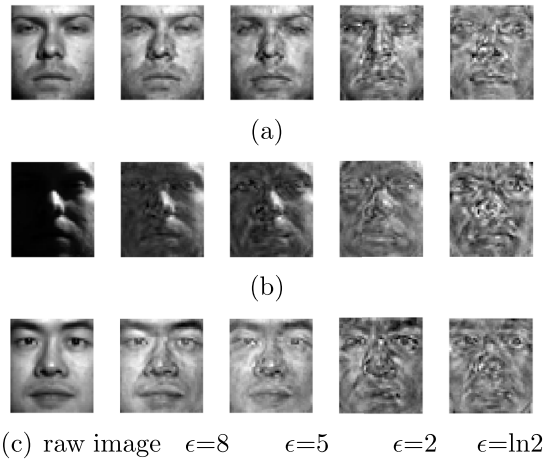


Fig. 4. Face images reconstruction with different feature extraction methods after protection with Algorithm 1. (a) SVD (b) Eigenface (c) Fisherface.

in the training data. In differential privacy, smaller ϵ provides higher privacy guarantee, however, it may leads to lower usability, i.e., lower recognition accuracy. From the results shown in Fig. 4 we can find that under small ϵ (such as ϵ=2 and ϵ=ln2), it is hard to recognize the identities of the face images after reconstruction attacks on the feature extraction methods of SVD and Eigenface. However, the added noises also have a huge impact on the recognition accuracy. In the literature, ϵ=ln2 is typically considered as providing acceptable level of privacy [39], but all ML algorithms lose their utility because the recognition accuracy drops to approximately 30% for kNN and SRC and 50% for SVM respectively as our evaluations in Section 4.4.

3.3. Acies - Feature Extraction Perturbation

To achieve high privacy guarantee while preserving acceptable utility of the classification model in edge computing, we propose a new classification model perturbation method **Acies** (Algorithm 2). The fundamental idea behind is to perturb the tailored feature extraction methods, to control the information leakage from the training data indirectly.

Considering linear regression purpose, matrix R and X from Algorithm 2 can be also regarded as two independent linear models. Hence, the models of R and X then can be written as $\sigma = R(\lambda)$ and $b = X(a)$, where λ and a are different inputs to each model and σ and b are their corresponding outputs. Under the context of edge computing, we mostly use the combined model ($R(X(\cdot))$) to handle the classification task, which refers to the feature selection process followed by those classifiers. Regardless of various classifiers, Acies takes linear

ALGORITHM 2: Feature Extraction Perturbation

Input: Normal input data/feature $X = \{x_1, x_2, x_3, ..., x_n\} \in \mathbb{R}^{n \times m}$, target privacy budget ϵ, extraction ratio k.
Output: Perturbed extracted/compressed feature set A .
Calculate the Singular Value Decomposition (SVD) of perturbed features transpose $(X)^T$: $(X)^T = U \Lambda V^T$;
Choose the first k column of U as the extraction matrix $R \in \mathbb{R}^{k \times n}$;
Calculate sensitivity $s = \max R - \min R$ **for** *each* $r_i \in R$ **do**
  Sample noise vector lp from Laplace(s/ϵ) ;
  $r_i' = r_i + lp$;
end
Get $A \in \mathbb{R}'^{k \times m}$ by multiply the extraction matrix transpose with perturbed input $A = (R')^T \times X$;

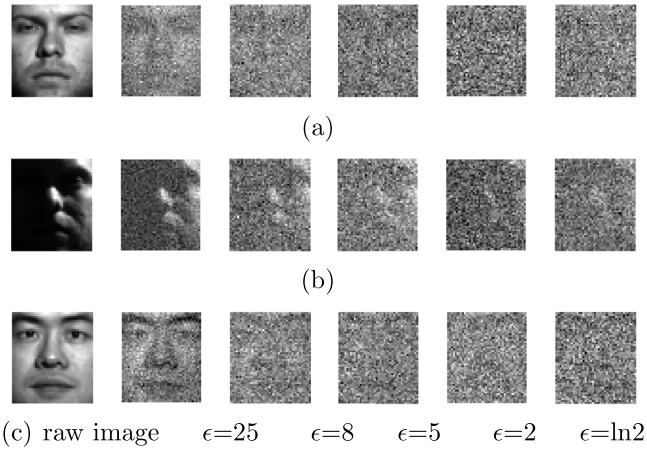


Fig. 5. Face images reconstruction with different feature extraction methods after protection with Algorithm 2. (a) SVD (b) Eigenface (c) Fisherface.

transformation to generate the $R(X(\cdot))$, which equals to A in Algorithm 2.

The outputs of Algorithm 2 are used to train classifiers which can be generally formulated as,

$$f = \arg\min \frac{1}{N} \sum_i L(x_i - f(A_i)), \tag{6}$$

where f is the target classifier, L is the corresponding loss function. Since the training set A is perturbed, the trained classifier then can also be considered perturbed.

The process of Algorithm 1 makes $X(\cdot)$ differentially private. Consequently, Algorithm 2 can make model $R(\cdot)$ to achieve ϵ-differential privacy. Based on the sequential composability of differential privacy [38], we can tell the output model A (see Algorithm 2 last line) is 2ϵ-differentially private if independently adding ϵ-differentially private noise on both $(R')^T$ (processed R) and X simultaneously. Similar to Algorithm 1, we use SVD as the feature selection in Algorithm 2, but it is applicable to other feature selection algorithms such as Eigenface and Fisherface.

We launch the reconstruction attacks to the features perturbed by Acies and present some preliminary results in Fig. 5. The face images are the same to those used in Fig. 4. The results show that the identities of the face images can be effectively protected when ϵ of R is equal to or below 8 which demonstrates significantly improved privacy preserving performance intuitively compared with the results shown in Fig. 4, i.e., Acies added significant lower level of noise to provide reliable privacy protection compared with the naive approach. We will evaluate the performance of the two approaches in details in Section 4.4.

Fig. 6 compares the differences between Acies (Algorithm 2) and the native approach (Algorithm 1). Although these two approaches

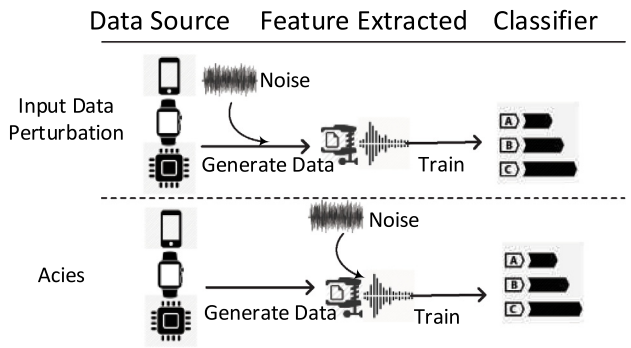


Fig. 6. Algorithms comparison. Noise is added to input data for input data perturbation. Acies adds noise when generating/extracted features. Both trained classifiers are privacy preserved.

have similar components, they have significant differences on how the Laplace noise is added to the ML models. The naive approach added the noise at the data source, however Acies added the noise while extracting the features. The noisy features are used for training and evaluation (or predication in the real-life scenario) simultaneously. Those classifiers (KNN, SVM and SRC) are continuously doing the l1 minimization, distance calculation, etc. on those noisy features. Since the same noisy feature extraction matrix is hold by both server and device sides, Acies does not make the final classification accuracy drop significantly on those classifiers (KNN, SVM and SRC), which are doing the l1 minimization, distance calculation, etc. processes.

4. Evaluation

4.1. Goals and metrics

In this section, we will first illustrate how Acies affects ML models' utility by evaluating the recognition accuracy of Acies with a number of IoTs application datasets, which include YaleB recognition (YaleB) [40], UCI Human Activity Recognition using smartphones (HAR) [41], hand written digits MNIST [42] and WiFi Channel State information (CSI) [32].

We then evaluate the performance of Acies on privacy protection. We use mutual information [43] as our evaluation metric, which measures the distance between the original training instance and the estimation of the instance obtained from reconstruction attacks. The mutual information of two variables A and B can be computed using the probability distributions,

$$I(A,B) = \sum_{a,b} p_{AB}(a,b) log \frac{p_{AB}(a,b)}{p_A(a) \cdot p_B(b)} \tag{7}$$

where $p_A(a)$ and $p_B(b)$ are marginal probability distribution and joint probability distribution $p_{AB}(a,b)$ are statistically independent if $p_{AB}(a,b) = p_A(a) \cdot p_B(b)$. When mutual information of two variables $I(A,B) = 0$, it implies that A and B are absolute independent. In the context of this paper, smaller $I(X,\hat{X})$ implies that X is better protected from reconstruction attacks.

At last, to illustrate Acies only introduce small system overhead to edge server/devices, we evaluate the time consumption and storage cost of different classification models. Experiments are conducted on a MacBook Pro (2.5 GHz i5) running Matlab2017a, which is regarded as edge server. We do not evaluate the consumption of training and feature extraction as they are computed on the Cloud which is resource rich.

4.2. Recognition accuracy of classification system with Acies

YaleB We choose the first 32 face images from each class as training dataset and the following 10 as testing. 300-dimensional feature vectors are extracted for different classifiers. We compute the classification accuracy of the two privacy protection methods over the 38 classes. The averaging classification accuracy results are presented in Fig. 7.

Compared with the input data perturbation approach shown on the left column of Fig. 7, Acies (the right column) achieves significantly higher recognition rate with smaller ϵ. Acies has different impacts on different feature extraction methods, for example, the accuracy of SVM with Fishface drops to 7% (note that this data point is not shown in Fig. 7(d)) while for other feature extraction methods, the change of ϵ has a less impact on accuracy.

CSI WiFi CSI data can be used to identify people based on the unique gait information. One data record is generated for a single person completing required activity. The raw CSI data is collected from 20 people, and each activity is repeated for 10 times. We follow the same feature selection method as in [32] and choose k=100 as dimensionality of the feature vector. We compute the recognition accuracy of SVD combining with different classifiers and the results are shown in Fig. 8. The baseline is the average of the classification results from the three different classifiers without any perturbation. From the results we can see that Acies has very little impact on classification accuracy for different values of privacy parameters ϵ.

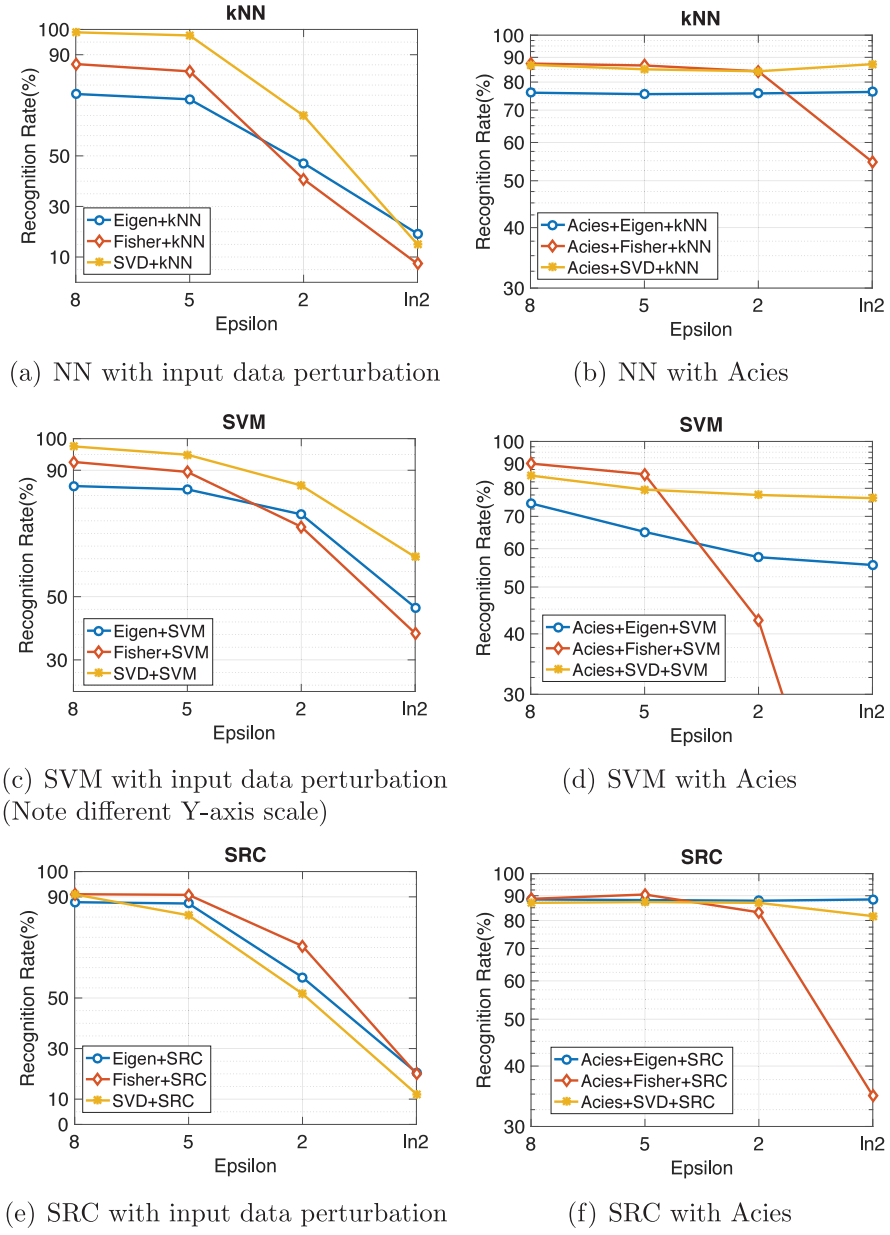
HAR HAR dataset is generated from the experiment carried out with a group of 30 people wearing a smartphone on the waist. The original purpose of the dataset is to classify human activities (walking, walking upstairs, walking downstairs, sitting, standing, laying) with data collected from the embedded accelerometer and gyroscope. HAR dataset consists of 561-dimensional feature set selected by the data contributors. In HAR, the training dataset contains 7352 data entries and testset contains 2947 data entries. After we evaluate the recognition accuracy with different number of extracted features, we choose k=50 as the default reduced dimensionality as there is no noticeable performance gain after that. We use the recognition accuracy computed without any perturbation as the baseline for the evaluation of Acies. The final evaluation results are shown in Fig. 9 and we observe that Acies has very limited impact on classification accuracy for all classifiers.

MNIST At last, we use MNIST dataset to evaluate the performance of Acies on classifying images other than faces. MNIST consists of hand written digits images from 0 to 9 with 28 × 28 pixels. We randomly choose 7000 images from MNIST, and use 6000 of them for training and 1000 of them for testing.

The raw data dimension for each image is 784 (28 × 28) in MNIST. We reduce the dimensionality of the image to k=70 for MNIST, because it is sufficient to produce acceptable recognition accuracy and there is no noticeable improvement after that. Fig. 10 illustrates the impact of Acies on classification accuracy. The results show that Acies preserves most of the utility and the recognition accuracy only decreases slightly (within 2%) for all classification models.

To take a closer look at the impact of Acies on the recognition accuracy, we show the confusion matrices of the classification results when applying Acies on the classification system with four different datasets in Fig. 11. For the benefit of clarity, we randomly selected 6 subjects from YaleB and CSI respectively. Those results in Fig. 11 could be seen as a subset of those reported in Figs. 7, 8, 9, 10. We can see that diagonals of the confusion matrices are always significantly darker than other entries which shows high recognition accuracy.

In summary, Acies shows good performance on preserving the utility of different classification models with various feature extraction methods on multiple datasets generally.



(a) NN with input data perturbation

(b) NN with Acies

(c) SVM with input data perturbation (Note different Y-axis scale)

(d) SVM with Acies

(e) SRC with input data perturbation

(f) SRC with Acies

Fig. 7. Recognition rates on extended Yale B database with input data perturbation (Algorithm 1) and Acies, for various feature transformations and classifiers.

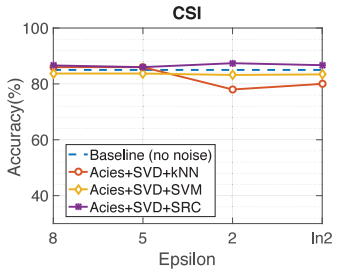


Fig. 8. Classification accuracy on CSI dataset.

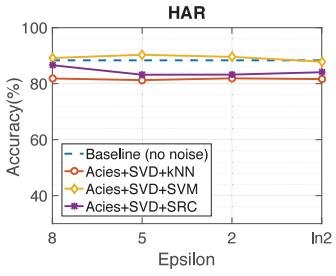


Fig. 9. Classification accuracy on HAR dataset.

Table 1
Time consumption for classification process with different extraction ratio.

Time consumption												
Data set/time (s)	YaleB			HAR			MNIST			CSI		
	5%	25%	100%	5%	25%	100%	5%	25%	100%	5%	25%	100%
kNN	0.006	0.01	0.02	0.01	0.02	0.05	0.3	0.4	1.1	0.05	0.13	0.71
SVM	0.9	1.1	1.2	0.02	0.06	0.08	0.12	0.14	0.17	0.32	0.36	1.87
SRC	0.3	0.67	1.58	0.5	1.1	2.1	0.5	0.8	1.5	0.3	0.9	2.3

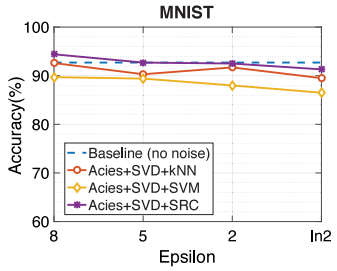


Fig. 10. Classification accuracy on MNIST dataset.

Table 2
SVM model storage required with different extraction ratio.

RAM cost					
Dimension reduction	1%	5%	10%	30%	100% *
YaleB	8.3MB	41.1MB	81.1MB	243.2MB	802MB
HAR	298KB	1.4MB	2.8MB	8.4MB	28MB
MNIST	34MB	171MB	338MB	0.9GB	3.3GB
CSI	803KB	3.2MB	6.2MB	18.1MB	62MB

4.3. Privacy analysis

In this section, we analyze the secrecy property of Acies under reconstruction attacks. We quantify how much information, in terms of mutual information, that an adversary can obtain from the classification models by launching the reconstruction attacks introduced in Section 2.4.

Fig. 12 shows the normalized mutual information revealed via launching reconstruction attacks on the classification models perturbed by input data perturbation (Algorithm 1) and Acies (Algorithm 2) respectively for all datasets with 30% compression ratio. The results show that the normalized mutual information degrades with the decrease of ϵ (the amount of noise increases) for both methods as expected. However, the normalized mutual information of Acies drops significantly quicker than that of input perturbation algorithm which implies that the performance of Acies on privacy protection is significantly better. For example, when ϵ is 8, the mutual information of Acies is less than 0.2, while that of input data perturbation is approximately 0.3. We have also evaluated Acies and input data perturbation using other compression ratios (e.g., 10% and 50%). The results are similar to those in Fig. 12, and the related plots are omitted for the benefit of space.

It is known that, there is a trade-off between the utility and privacy. We quantify the trade-off using classification accuracy and the normalized mutual information and present the results in Fig. 13. We select a valuable area for general application with classification accuracy higher than 80% and normalized mutual information below 0.15. These two values are derived based on experience on previous experiments on YaleB (classification and reconstruction). For different application purposes, the accuracy and privacy requirement may vary. With above (accuracy and normalized mutual information) parameter value selection, there is total only 2 available options for naive data perturbation methods with 12 combinations of different ML schemes and privacy budgets for each dataset. On the contrary, Acies can generally satisfy the requirement about utility and privacy for various datasets.

4.4. Resource consumption analysis

To evaluate the resources consumption of the classification systems with Acies, we measure the computation time and storage cost of different classification models with different datasets.

We conduct experiments on the same datasets as we used in the previous section, but for the benefit of space we select dimension reduction ratios of 5% and 25% as examples. As the results shown in Table 1, different dimension reduction ratios can affect the time consumption significantly for the same algorithms while time consumption is important for user experience (latency or system response time) issues. We take the classification using CSI as an example, the feature's original dimension is 10,800 (showed as 100%). When the feature dimension is compressed to 5%, the time consumption of classification model can be improved by 13, 6 and 8 times for kNN, SVM and SRC respectively. Similar results can also be observed from other datasets.

Storage size (or RAM) is another important resource of IoTs devices. When a classification task is included in the IoTs device's schedule, sufficient storage space is required necessarily for accommodating the large number of parameters in trained model which is offloaded from the Cloud. Even though modern ML techniques have been highly optimized to "extract" and only save the essential parameters in the model, the remaining can still be huge. Table 2 illustrates the storage used to accommodate the model trained with SVM (linear kernel) with different size of training set. We use the same setting as in Section 4.2 when running the classifiers. From the results in Table 2, we can observe that the dimension reduction of training dataset reduced the RAM consumption significantly. However, the RAM overhead is still considerable for some IoTs device platforms (e.g., 333MB for MNIST dataset when dimension reduction ratio is 10%.).

5. New potential attack and countermeasures at edge

Edge-computing network is a still new and under developing paradigm, and all those potential security and privacy issues are not fully researched. Differential privacy-related solutions are usually regarded as the "tiger balm" at this stage. However, those various implementations could still break the privacy guarantee in certain circumstance.

Put the face recognition application into real use, for some reason, the tailored model is updated and has to be re-deployed on edge server. For example, new resident moving into the smart building. Some issues like authority delegation or secure tunnel communication are beyond this paper's scope. We only focus on the problem derived by the classification model re-deploy.

According to the linear algebra theory, if the equation is underdetermined, and there is no other auxiliary information like high mutual coherent between R and X or X is sparse, it is difficult for attacker to reconstruct the X from $A = R \cdot X$. With only one pair of known A and R, the equation even becomes a One-Time-Pad (OTP) scenario, which achieve 'information-theoretically secure'. And for some situation, the attacker have no idea about how close the estimate data $\hat{X}$ to the raw data X which makes the reconstruction even harder practically [44]. Combined with our proposed Acies, this OTP and randomly perturbed

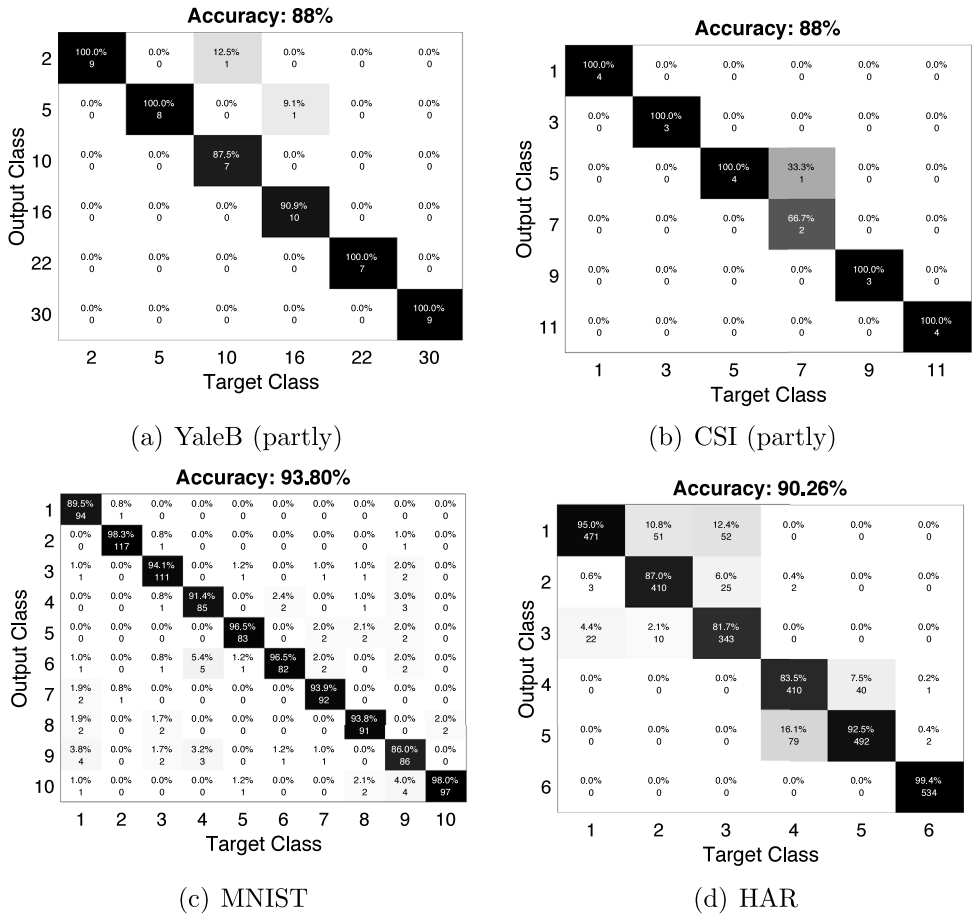


Fig. 11. Confusion matrices of different datasets (YaleB, CSI, MNIST, HAR). The x and y axis indicate the index of the sample.

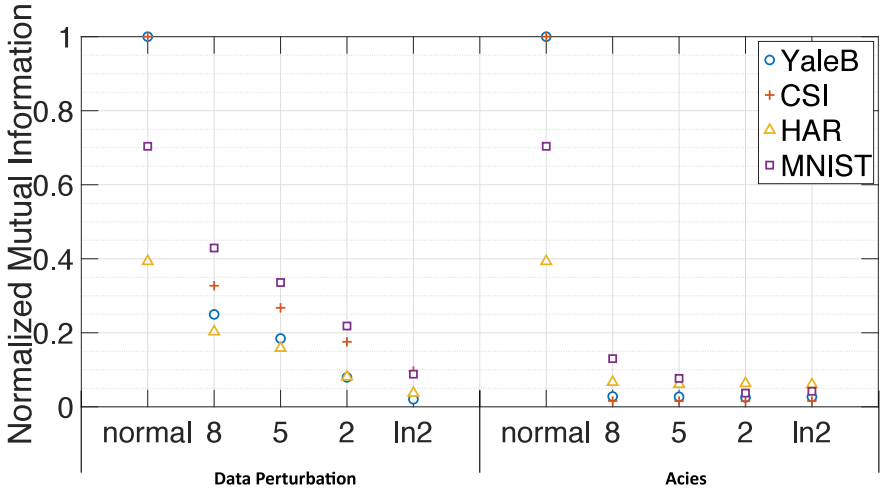


Fig. 12. Effect of reconstruction attack on different datasets.

(differential privacy) framework perfectly prevents the information leaking effectively at "one-time" using scenario.

Accumulation attack. When looking back to the essence of edge computing framework, we found an potential point could be the weakness. Since the edge server or edge devices (IoT devices) perform some computation task with *pre-processed model on hand*, there can exist situations that the model have to be re-sent and re-deployed. For example, re-train the model because new user (face images) involved in the application. Consequently, the new perturbed model has to be re-deployed on edge end. Unfortunately, in this case, the OTP condition is broken.

Every time new pre-processed model sent to edge node, the node (and attacker) can have one more equation contain information about X:

$$A' = R' \cdot (X + \Sigma) \tag{8}$$

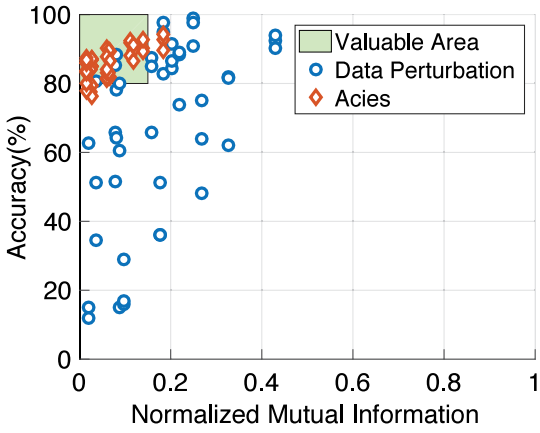


Fig. 13. Classification accuracy and normalized mutual information of two methods.

where Σ represents to the new information (e.g., training data, selected features) appended to the old training dataset, R' is the new feature extraction matrix and A' is the new model.

Even R' is still generated based on random-related mechanism (e.g., Acies), the attacker has accumulated more information (equations) about the previous users' data X. Take Bob's face images as an example, we assume attacker Alice always know the column number of Bob's δ-th face image in A (, A' and so on). Alice then can re-build the reconstruction equation:

$$\begin{aligned} a_\delta &= R \cdot x_\delta \\ a'_\delta &= R' \cdot x_\delta \\ &\dots \\ a'''_\delta &= R''' \cdot x_\delta \end{aligned} \tag{9}$$

where a_δ respond to the relative column in A, x_δ refers to the Bob's δ-th face image. With multiply times model updating, Alice can solve Eq. (3) and reconstruct the face image with high possibility.

Not to our surprise, Alice successfully reconstructed the Bob's face image that human can recognize. Accumulation attack is lunched against Acies and the result can be seen in Fig. 14. There is no standard criteria to define how much information is enough to recognize a face image in this paper, and there are more other situations like know the target person before or not. We omit the discussion and experiments in this paper. This part only illustrates there is a high possibility to 'acquire' other users' private information from the extracted model as long as the model is being long-term used. Random-based permutation even differential privacy (e.g., Acies) cannot stop Alice's success. No doubt that Alice can finally find Bob's δ-th face image under polynomial times trial and error with sufficient information.

Possible countermeasures. The fundamental idea of the accumulation attack is based on the fact that once the attacker accumulate 'sufficient' information from the application (e.g., steal from the compressed machine learning algorithm or eavesdrop the communication channel), the possibility for those information get reconstructed by attacked will increase (refer to Fig. 14). Applications implemented diversely will required different countermeasure which can fit the scenario. The generic possible countermeasure can also derive from preventing the information accumulation. Instead of reducing the model updating frequency (it is hard to be controlled practically), we can prevent the information get reconstructed by updating the old training data used in the application. For example, re-take the face photo regularly in the smart building scenario or use different training data each time when updating the model. Thus, the attacker is hard to reconstructed useful information from overlapped data (cannot accumulate valid information).

Fig. 14. Accumulation attack on Acies extraction methods after 1, n/2, 7/10n, n times model updating. n is the compression ratio used in the feature extraction.

In addition to that, perturb the raw training data will also improve the security against the accumulation attack. For instance, slightly injecting noise into raw training data will increase the difficulty of solving Eq. (8) to 13 mathematically. However, it is clear that adding differentially private noise into raw training data will obviously drop the model's utility (see left-side figures in Fig. 7). Understanding the guaranties and the effectiveness of differential privacy techniques against those attackers poses considerable challenges. And usually the mitigation outcomes highly depend on the data/model characteristics in practice. Research [45] has illustrated there is only negligible privacy risk even launched by the state-of-the-art membership inference attack while the training data is in skewed prior probability distribution setting. Although the non-private models are highly vulnerable in the balanced prior probability distribution setting, a small amount of privacy noise can mitigate the privacy risk well. This also holds in our attack. Attackers' capability will be restricted when the training data distribution is skewed in real-world scenario, where only a small fraction of the candidate population is included in the overlapped data. Adapted differential definition budget like personalized differential privacy [46], heterogeneous differential privacy [47], etc might be considered. Thus, more research efforts are required to put forward a reliable system with good privacy and utility trade-off.

6. Related work

In this paper, we perturb the classification models to preserve both the privacy and utility. Similar studies [6–10,44] have been conducted in the literature. However, the existing work is not applicable in edge computing environment as they focused on cloud computing scenarios which have significantly distinctive network architecture and application scenarios. The state-of-the-art privacy preserving mechanisms are either data-oriented (such as over partitioned dataset) [7,8] or focus on specific ML model [9,44,48], which require deliberate manipulation to make it as a fair comparison to the propose work.

The first category of the related work was the studies on addressing the privacy leakage when releasing the classifiers. Lin and Chen [6] proposed to control the released features to prevent the privacy leakage. For conventional classification process with SVM classifier, all support vectors must be kept in the classifier, which might violate privacy. To protect the sensitive content in the classifier, a post-processing privacy-preserving SVM classifier schema was used to protect the sensitive content of support vectors in the classifiers. Many researches had also been proposed to protect the privacy leakage of other classifiers [7–9,44] and they randomly perturbed the feature extraction methods in different classifiers. Those approaches required that all participants shared a common perturbation matrix to vertically/horizontally partition the private data/feature, where elements of the feature vector were spread among participants. The global SVM classification models were constructed from data distributed at multiple parties, without disclosing the data of each party to others. Solutions were sketched out for data that was arbitrarily partitioned. Those privacy-preserving classifiers are usually specific (to data or applications) in the context of edge computing. The released classifier is always limit to each dataset or application scenario. To achieve good utility of the released classifiers, complex process and fine-tuning computation are necessary which also increase the overhead. This issue can get worse when there is necessity

(e.g., adding new training data) to train a new classifier, the overhead increase is exponential.

Another category of related work was the cloud-enabled privacy-preserving collaborative learning (also known as participatory sensing). Pickle [10] protected the collaborate data privacy by asking each service user hold a different random perturb matrix R. The cloud then could build the regression model based on the previous distance information learned from user uploaded perturbed public data vector. By doing this, the Cloud could calculate the estimate distance among the collaborate data vectors contributed by independent users. In summary, Pickle perturbs the each user's training data by injecting random noise to original training data and then projecting to another domain, which was similar to the baseline method in this paper. The performance of Pickle is significantly worse than Acies in the context of edge computing.

Rappor [49], is a technology for crowdsourcing statistics from end-user client software, adopts the similar idea as Pickle. Rappor allows the forest of client data to be studied, without permitting the possibility of looking at individual trees. These privacy-preserving service [44,49] only provide statistic information though with strong privacy guarantee. As a result, only limited services are provided for users e.g., generate the distribution of the private data. Besides, if those statics information is used for further applications like training ML models, the accuracy will be far from the model trained with unprocessed data.

7. Conclusion

In this paper, we propose, Acies, a privacy-preserving system for classification on IoTs devices under edge computing environment. Instead of direct input data perturbation in the training set, which has huge impact on the classification accuracy, Acies is designed to perturb the feature extraction component to address the privacy issues when ML models are offloaded from the Cloud to edge nodes. By comparing the naive input data perturbation method on multiple datasets with different classification models, we can show that Acies achieves significantly better trade-off on privacy and utility preserving than naive approach: Acies provides trusted classification services with minimal impact on classification accuracy (2%–5%). And we discussed the future potential concerns (e.g., accumulation attack) in edge computing-based network paradigms and possible countermeasures at last for future research direction.

CRediT authorship contribution statement

Wanli Xue: Conceptualization, Methodology, Software, Writing – original draft. **Yiran Shen:** Data curation, Writing – original draft. **Chengwen Luo:** Conceptualization, Investigation. **Weitao Xu:** Data curation, Writing – review & editing. **Wen Hu:** Writing – review & editing, Supervision. **Aruna Seneviratne:** Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This research is partially supported by National Natural Science Foundation of China [Grant No. 61972263], Natural Science Foundation of Guangdong Province, China [Grant No. 2019A1515011608]. The work described in this paper was partially supported by Shenzhen Science and Technology Funding Fundamental Research Program, China (Project No. 2021Szvup126), a grant from Chow Sang Sang Group Research Fund sponsored by Chow Sang Sang Holdings International Limited, China (Project No. 9229062), and CityU APRC, China grant (Project No. 9610485).

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.comcom.2021.10.038.

References

[1] D. Evans, The internet of things: How the next evolution of the internet is changing everything, white paper, cisco systems, 2013, Online; accessed 29 Sep 2017.
[2] P. Garcia Lopez, A. Montresor, D. Epema, A. Datta, T. Higashino, A. Iamnitchi, M. Barcellos, P. Felber, E. Riviere, Edge-centric computing: Vision and challenges, ACM SIGCOMM Comput. Commun. Rev. 45 (5) (2015) 37–42.
[3] H. Pang, K.-L. Tan, Authenticating query results in edge computing, in: Data Engineering, 2004. Proceedings. 20th International Conference on, IEEE, 2004, pp. 560–571.
[4] K. Ha, P. Pillai, G. Lewis, S. Simanta, S. Clinch, N. Davies, M. Satyanarayanan, The impact of mobile multimedia applications on data center consolidation, in: Cloud Engineering (IC2E), 2013 IEEE International Conference on, IEEE, 2013, pp. 166–176.
[5] K. Ha, Z. Chen, W. Hu, W. Richter, P. Pillai, M. Satyanarayanan, Towards wearable cognitive assistance, in: Proceedings of the 12th Annual International Conference on Mobile Systems, Applications, and Services, ACM, 2014, pp. 68–81.
[6] K.-P. Lin, M.-S. Chen, Releasing the svm classifier with privacy-preservation, in: Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on, IEEE, 2008, pp. 899–904.
[7] O.L. Mangasarian, E.W. Wild, G.M. Fung, Privacy-preserving classification of vertically partitioned data via random kernels, ACM Trans. Knowl. Discov. Data 2 (3) (2008) 12.
[8] H. Yu, J. Vaidya, X. Jiang, Privacy-preserving SVM classification on vertically partitioned data., in: PAKDD, Vol. 3918, Springer, 2006, pp. 647–656.
[9] J. Qiang, B. Yang, Q. Li, L. Jing, Privacy-preserving svm of horizontally partitioned data for linear classification, in: Image and Signal Processing (CISP), 2011 4th International Congress on, Vol. 5, IEEE, 2011, pp. 2771–2775.
[10] B. Liu, Y. Jiang, F. Sha, R. Govindan, Cloud-enabled privacy-preserving collaborative learning for mobile sensing, in: Proceedings of the 10th ACM Conference on Embedded Network Sensor Systems, ACM, 2012, pp. 57–70.
[11] W. Xue, Y. Shen, C. Luo, W. Hu, A. Seneviratne, Acies: A privacy-preserving system for edge-based classification, in: 2018 17th IEEE International Conference on Trust, Security and Privacy in Computing and Communications/12th IEEE International Conference on Big Data Science and Engineering (TrustCom/BigDataSE), IEEE, 2018, pp. 914–919.
[12] S. Yi, C. Li, Q. Li, A survey of fog computing: concepts, applications and issues, in: Proceedings of the 2015 Workshop on Mobile Big Data, ACM, 2015, pp. 37–42.
[13] F. Bonomi, R. Milito, J. Zhu, S. Addepalli, Fog computing and its role in the internet of things, in: Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing, ACM, 2012, pp. 13–16.
[14] F. Bonomi, R. Milito, P. Natarajan, J. Zhu, Fog computing: A platform for internet of things and analytics, in: Big Data and Internet of Things: A Roadmap for Smart Environments, Springer, 2014, pp. 169–186.
[15] L.M. Vaquero, L. Rodero-Merino, Finding your way in the fog: Towards a comprehensive definition of fog computing, ACM SIGCOMM Comput. Commun. Rev. 44 (5) (2014) 27–32.
[16] T.H. Luan, L. Gao, Z. Li, Y. Xiang, G. Wei, L. Sun, Fog computing: Focusing on mobile users at the edge, 2015, arXiv preprint arXiv:1502.01815.
[17] M.A. Turk, A.P. Pentland, Face recognition using eigenfaces, in: Computer Vision and Pattern Recognition, 1991. Proceedings CVPR'91., IEEE Computer Society Conference on, IEEE, 1991, pp. 586–591.
[18] P.N. Belhumeur, J.a.P. Hespanha, D.J. Kriegman, Eigenfaces vs. fisherfaces: Recognition using class specific linear projection, IEEE Trans. Pattern Anal. Mach. Intell. 19 (7) (1997) 711–720.
[19] J. Wright, A.Y. Yang, A. Ganesh, S.S. Sastry, Y. Ma, Robust face recognition via sparse representation, IEEE Trans. Pattern Anal. Mach. Intell. 31 (2) (2009) 210–227.
[20] H.-S. Ham, H.-H. Kim, M.-S. Kim, M.-J. Choi, Linear SVM-based android malware detection for reliable IoT services, J. Appl. Math. 2014 (2014).
[21] D. Azariadi, V. Tsoutsouras, S. Xydis, D. Soudris, ECG signal analysis and arrhythmia detection on IoT wearable medical devices, in: Modern Circuits and Systems Technologies (MOCAST), 2016 5th International Conference on, IEEE, 2016, pp. 1–4.
[22] J. Zhu, T. Hastie, Kernel logistic regression and the import vector machine, J. Comput. Graph. Statist. 14 (1) (2005) 185–205.
[23] S.P. Kasiviswanathan, M. Rudelson, A. Smith, The power of linear reconstruction attacks, in: Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms, Society for Industrial and Applied Mathematics, 2013, pp. 1415–1433.

[24] R. Shokri, M. Stronati, V. Shmatikov, Membership inference attacks against machine learning models, 2016, arXiv preprint arXiv:1610.05820.
[25] R. Roman, J. Lopez, M. Mambo, Mobile edge computing, fog et al.: A survey and analysis of security threats and challenges, Future Gener. Comput. Syst. 78 (2018) 680–698.
[26] A. Pentland, T. Choudhury, Face recognition for smart environments, Computer 33 (2) (2000) 50–55.
[27] I. Dinur, K. Nissim, Revealing information while preserving privacy, in: Proceedings of the Twenty-Second ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, ACM, 2003, pp. 202–210.
[28] C. Dwork, F. McSherry, K. Talwar, The price of privacy and the limits of LP decoding, in: Proceedings of the Thirty-Ninth Annual ACM Symposium on Theory of Computing, ACM, 2007, pp. 85–94.
[29] D. Anguita, A. Ghio, L. Oneto, X. Parra, J.L. Reyes-Ortiz, A public domain dataset for human activity recognition using smartphones, in: ESANN, 2013.
[30] X. Su, H. Tong, P. Ji, Activity recognition with smartphone sensors, Tsinghua Sci. Technol. 19 (3) (2014) 235–249.
[31] D. Anguita, A. Ghio, L. Oneto, X. Parra, J.L. Reyes-Ortiz, Human activity recognition on smartphones using a multiclass hardware-friendly support vector machine, in: International Workshop on Ambient Assisted Living, Springer, 2012, pp. 216–223.
[32] J. Zhang, B. Wei, W. Hu, S.S. Kanhere, Wifi-id: Human identification using wifi signal, in: Distributed Computing in Sensor Systems (DCOSS), 2016 International Conference on, IEEE, 2016, pp. 75–82.
[33] Y. Zeng, P.H. Pathak, P. Mohapatra, Wiwho: wifi-based person identification in smart spaces, in: Proceedings of the 15th International Conference on Information Processing in Sensor Networks, IEEE Press, 2016, p. 4.
[34] C. Dwork, Differential privacy: A survey of results, in: International Conference on Theory and Applications of Models of Computation, Springer, 2008, pp. 1–19.
[35] C. Dwork, A. Roth, et al., The algorithmic foundations of differential privacy, Found. Trends Theor. Comput. Sci. 9 (3–4) (2014) 211–407.
[36] C. Dwork, F. McSherry, K. Nissim, A. Smith, Calibrating noise to sensitivity in private data analysis, in: TCC, Vol. 3876, Springer, 2006, pp. 265–284.
[37] K. Chaudhuri, C. Monteleoni, A.D. Sarwate, Differentially private empirical risk minimization, J. Mach. Learn. Res. 12 (Mar) (2011) 1069–1109.
[38] F.D. McSherry, Privacy integrated queries: an extensible platform for privacy-preserving data analysis, in: Proceedings of the 2009 ACM SIGMOD International Conference on Management of Data, ACM, 2009, pp. 19–30.
[39] A. Friedman, S. Berkovsky, M.A. Kaafar, A differential privacy framework for matrix factorization recommender systems, User Model. User-Adapted Int. 26 (5) (2016) 425–458.
[40] Yale, The extended yale face database B, 2001, URL http://vision.ucsd.edu/~iskwak/ExtYaleDatabase/ExtYaleB.html.
[41] D. Anguita, A. Ghio, L. Oneto, X. Parra, J.L. Reyes-Ortiz, A public domain dataset for human activity recognition using smartphones, in: ESANN, 2013.
[42] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proc. IEEE 86 (11) (1998) 2278–2324.
[43] D. Agrawal, C.C. Aggarwal, On the design and quantification of privacy preserving data mining algorithms, in: Proceedings of the Twentieth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, ACM, 2001, pp. 247–255.
[44] W. Xue, C. Luo, G. Lan, R. Rana, W. Hu, A. Seneviratne, Kryptein: a compressive-sensing-based encryption scheme for the internet of things, in: Information Processing in Sensor Networks (IPSN), 2017 16th ACM/IEEE International Conference on, IEEE, 2017, pp. 169–180.
[45] B. Jayaraman, L. Wang, D. Evans, Q. Gu, Revisiting membership inference under realistic assumptions, 2020, arXiv preprint arXiv:2005.10881.
[46] Z. Jorgensen, T. Yu, G. Cormode, Conservative or liberal? Personalized differential privacy, in: 2015 IEEE 31St International Conference on Data Engineering, IEEE, 2015, pp. 1023–1034.
[47] M. Alaggan, S. Gambs, A.-M. Kermarrec, Heterogeneous differential privacy, 2015, arXiv preprint arXiv:1504.06998.
[48] J. Vaidya, C. Clifton, Privacy-preserving k-means clustering over vertically partitioned data, in: Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2003, pp. 206–215.
[49] U. Erlingsson, V. Pihur, A. Korolova, Rappor: Randomized aggregatable privacy-preserving ordinal response, in: Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, ACM, 2014, pp. 1054–1067.