IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 9, 2018, accepted November 28, 2018, date of publication December 3, 2018,
date of current version December 27, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2884511

PPDCA: Privacy-Preserving Crowdsourcing Data Collection and Analysis With Randomized Response

YAO-TUNG TSOU, (Member, IEEE), AND BO-CHENG LIN
Department of Communications Engineering, Feng Chia University, Taichung 407, Taiwan

Corresponding author: Yao-Tung Tsou (yaodong1014@gmail.com)

This work was supported by the Ministry of Science and Technology, Taiwan, under Grants MOST 107-2221-E-035-020-MY3 and MOST 106-2622-8-002-011-TA.

ABSTRACT Randomized response mechanisms for guaranteeing crowdsourcing data privacy have attracted scholarly attention; aggregators can ensure privacy by collecting only randomized data, and individuals can have plausible deniability regarding their responses. With these mechanisms, analysts employed by organizations can still make predictions and conduct analyses using the randomized data. Existing randomized response-based data collection solutions have severely restricted functionality and usability, resulting in impractical and inefficient systems. Therefore, we developed a randomized response-based privacy-preserving crowdsourcing data collection and analysis mechanism. We designed a complementary randomized response (C-RR) method to guarantee individuals' data privacy and to preserve features from the original data for analysis. We formalized a machine learning framework; our proposed method uses randomized data in the form of binary vectors to generate a learning network. Extensive experiments on real-world data sets demonstrated that our heavy-hitters estimation scheme, which applies C-RR and our data learning model, significantly outperformed existing estimation schemes in terms of data analysis.

INDEX TERMS Randomized response, local differential privacy, data analysis, randomized data.

I. INTRODUCTION

Because of the rapid development of the Internet of Things (IoT) and social networks, various forms of data are becoming increasingly important for both organizations and individuals. Organizations (*e.g.*, computer companies and academic institutes) aspire to learn valuable information from collected individual data to improve the quality levels of various services or to establish commercial strategies. Individuals may wish to release their data to the public or interested third parties to obtain rewards or advanced services, but almost all individuals demand that their private data (*e.g.*, web browsing history, service usages, and visited locations) not be revealed. Consequently, people face a puzzle between maximizing the quality of experiences and minimizing the leakage of private information.

Randomized response mechanisms that satisfy local differential privacy [25] have drawn considerable interest from the privacy research community, and this is because randomized response mechanisms can address the aforementioned puzzle. For example, RAPPOR [28], which is included as part of Google Chrome, constantly collects users' responses to questions such as the default homepage of the browser and the default search engine in a differentially private manner to forestall the unwanted or malicious hijacking of user settings. At the Apple Worldwide Developers Conference (WWDC) in 2016, Apple [3] announced its implementation of differential privacy in iOS 10 for discovering popular emojis, identifying excessive electricity consumption, and understanding memory usage in Safari. Microsoft [5] deploys a differentially private data collection mechanism in its Windows Insiders program to gather application usage statistics. Samsung [27] also proposed a differentially private system that enables collection of both categorical reports (*e.g.*, screen deployment) and numerical reports (*e.g.*, remaining battery, time of device usage). Both companies and users can benefit from differentially private data sharing. Companies can promote their commercial strategies or service optimization; users can have a premium experience by consuming software or services without losing their individual privacy.

The concept of randomized response is to analyze collected information without collecting data that can be traced to specific users. In contrast to the earliest differential

privacy mechanisms [9], [10], which outsource obfuscated reports or data sets but still collect actual sensitive information from individuals, randomized response methods avoid gathering sensitive information from individuals from the very beginning. In particular, any randomized response mechanism that satisfies local differential privacy can rigorously guarantee individual data privacy. More precisely, individuals have ''plausible deniability'' regarding the sensitive information that belongs to them, regardless of attackers' background knowledge.

Several studies (*i.e.*, [17] and [28]) have used randomized response approaches to protect individuals' data privacy, but such studies have been marred by the drawback of inaccurate data analysis. The fundamental cause of their inaccurate data analysis is the design of their data randomization algorithms and their data decoding or learning models. To address this drawback, we developed **P**rivacy-**P**reserving crowdsourcing **D**ata **C**ollection and **A**nalysis (PPDCA), an advanced, randomized-response-based data collection and analysis mechanism that guarantees local differential privacy using a complementary randomized response (C-RR) and decodes collected data in accordance with TensorFlow [2]. Additionally, we proved that PPDCA meets the definition of local differential privacy and demonstrated that PPDCA outperforms state-of-the-art methods [17], [28] in accuracy of data analysis.

The remainder of this paper is organized as follows. Section II describes related work. We present the proposed system model, data aggregation protocol, attack model, and notations in Section III. We then describe the details of our method, PPDCA, and ε-differential privacy analysis in Sections IV and V; we present our learning model construction in Section VI. Subsequently, we describe extensive experiments in Section VII. Finally, we present the concluding remarks in Section VIII.

II. RELATED WORK

In 1965, Warner [24] proposed randomized response as a proven efficient mechanism that satisfies local differential privacy [25] for a situation in which no trusted curator exists. Randomized response uses secret coin flips as random events to decide the answers to sensitive questions, such as ''Are you a homosexual?'' An individual would respond to this question truthfully only if the coin flip returns heads. Otherwise, the individual will flip a second coin to determine the answer and respond ''Yes'' if heads and ''No'' if tails. Individuals hold their own private data and release them to curators in a differentially private manner. For ease of presentation, we refer to individuals' private data as client strings d. In this case, the data set D is viewed as a sequence of d, and the neighboring data sets D_1 and D_2 are also viewed as two distinct sequences of strings d_1 and d_2. Thus, a local randomized algorithm $\mathcal{A}$ delivers ε-differential privacy [9] if for all $S_{\mathcal{A}} \subset \text{Range}(\mathcal{A})$ and every pair of d_1 and d_2,

$$Pr[\mathcal{A}(d_1) \in S_{\mathcal{A}}] \leq e^{\varepsilon} \times Pr[\mathcal{A}(d_2) \in S_{\mathcal{A}}],$$

where the probability of each result is taken in terms of the coin flips of the local randomized algorithm $\mathcal{A}$ and ε is called the ''privacy budget.'' The privacy budget determines the extent of privacy leakage. A relatively small ε provides relatively rigorous data privacy protection but allows relatively low data analysis accuracy.

Because the local differentially private model does not require a trusted curator [20], it has received growing attention recently. In real-world applications, market researchers aspire to know which behaviors occur the most frequently among all events, which is called the ''heavy-hitters problem.'' To address the heavy-hitters problem, Erlingsson *et al.* developed randomized aggregatable privacy-preserving ordinal response (RAPPOR) [28]. The actual client-side string in [28] is represented as a bit vector using a Bloom filter [8] and released in a noisy version after a multilayer randomized response. The major contribution of RAPPOR is its sensitive decoding framework for learning statistics, which can not only address the heavy-hitters problem but also reconstruct the population of client-side strings.

Several studies on data privacy protection have been conducted on models of local differentially private learning since the development of RAPPOR. In 2016, Fanti *et al.* [14] proposed an extended version of RAPPOR. They developed a new data decoding algorithm to allow aggregators to determine the joint distribution and decode data efficiently under a precise data dictionary. However, in achieving these two goals, they sacrificed the capability to accurately rebuild data. More precisely, aggregators could only observe a few clients' strings with high frequency after decoding.

Qin *et al.* [32] proposed the method LDPMiner, which first uses a partial ε of differential privacy to generate a candidate set of heavy hitters and then exploits the remainder of ε to refine the results. Although LDPMiner expands the applicability of RAPPOR, it focuses on heavy-hitter reconstruction in set-valued data instead of categorical data.

To determine the optimal parameters for RAPPOR, Wang *et al.* [26] proposed the OLH mechanism. However, OLH is only applied to reconstruct heavy hitters for a small domain of users' data, whereas RAPPOR and our method do not have this limitation. In 2017, S2M and S2Mb were proposed by Sei and Ohsuga [30], who used mean square errors and Jensen–Shannon divergence to demonstrate that both can achieve utility similar to RAPPOR. Subsequently, Yang *et al.* [17] developed MLDP, which was built on a fog computing architecture, to protect data with differential privacy; MLDP uses regression methods of machine learning to obtain limited analytic results.

With the aim of protecting individuals' privacy, Apple's differential privacy team [3] designed scalable local differential privacy mechanisms and provided analyses to show the trade-offs between device bandwidth, server computation, data privacy, and data utility. More precisely, they designed the Private Hadamard Count Mean Sketch algorithm, which has the advantage that users can transmit adjustable bits based on users' device bandwidths for minimizing transmission

cost. Reference [3] focused on practical deployments of learning systems considering the trade-offs between device bandwidth, server computation, data privacy, and data utility. Nevertheless, [3] does not strive to significantly improve the accuracy in reconstructing clients' data. Specifically, the approach proposed by Apple still suffers from the inaccuracy of reconstructed data, just as RAPPOR does.

LoPub [29] utilized expectation maximization and Lasso regression to provide multivariate joint distribution estimation and correlation identification under the circumstance that each client individually reports the data in a local differentially private manner. In contrast to LoPub, the goal of our method is the same as that of [17], [26], and [28]; our method focuses on the challenge of accurate heavy-hitter estimation. Additionally, LoPub used the indicator of the distance metric AVD (average variant distance) to quantify the utility of their elaborated synthetic data set. Although LoPub addressed the publication of privacy-preserving crowdsourced data, we cannot make a horizontal comparison with their evaluation because it used a different indicator. In particular, our privacy-preserving data collection in a differentially private manner is based on a supervised multilayer perceptron within a fog computing architecture and alleviates the computational burdens on cloud servers and on traditional servers. If the system reports only the results of the query or aggregation to the server, the efficiency of communication can be significantly improved [17].

Notably, Hitaj *et al.* [6] demonstrated that the privacy-preserving machine learning model is susceptible to the Generative Adversarial Network (GAN) attack problem in its collaborative mode. In particular, a collaborative learning model shares a subset of model parameters among users. In this case, an attacker employing a GAN can deceive any victim into releasing their private information. More precisely, the attacker defined in [6] simply runs a collaborative learning algorithm and reconstructs sensitive information stored on the victim's device. The attacker can also influence the learning process and deceive the victim into releasing more detailed information even when model parameters are obfuscated with differential privacy. By contrast, our model involves centralized learning; that is, we do not share our learning model parameters with users. Therefore, our model does not suffer from this type of attack.

In privacy-preserving crowdsourcing data collection and analysis, researchers strive to prevent attackers from obtaining individuals' information in memory, on hard disk, or in other forms. Numerous studies [12], [15], [16], [18], [21], [22], [31] have proposed well-designed data collection and analysis methods over encrypted domains, which are based on sophisticated cryptography systems that are horizontal to our method. In many of these studies, the major concerns are computational cost and secret key management overhead. Particularly, the effectiveness of these methods relies on foresight regarding attackers' background knowledge. By contrast, our method satisfying the definition of ε-differential privacy can have a rigorous privacy guarantee, regardless of the attackers' background knowledge.

According to our survey of state-of-the-art methods, the original RAPPOR is the most effective and relevant method for addressing client-side data collection and analysis, and the computing architecture of MLDP is the most similar to that of our machine learning model.

PPDCA, which is an extension of our previous work [4], addresses three perspectives that are different from those of the aforementioned methods: (1) C-RR is elaborated to provide a rigorous data privacy guarantee for individuals while preserving high-utility analyses; (2) a Tensorflow machine learning model is enabled to engage in high-utility learning and prediction for randomized data; and (3) through experiments, PPDCA was proven to be effective and to outperform RAPPOR and MLDP in data prediction accuracy.

III. PROBLEM STATEMENT

In this section, the definition of PPDCA, including the system model, data aggregation protocol, attack model, and notations, are formulated and described in detail.

A. SYSTEM MODEL

As shown in Fig. 1, our system is based on a fog/edge computing data aggregation architecture that involves five roles: client, fog node, fog center, mobile edge cloud, and cloud server. Each role is described as follows.

- Client: The client's behaviors are monitored by IoT devices; behaviors are analyzed elsewhere. To solve privacy concerns, the original behavioral data are

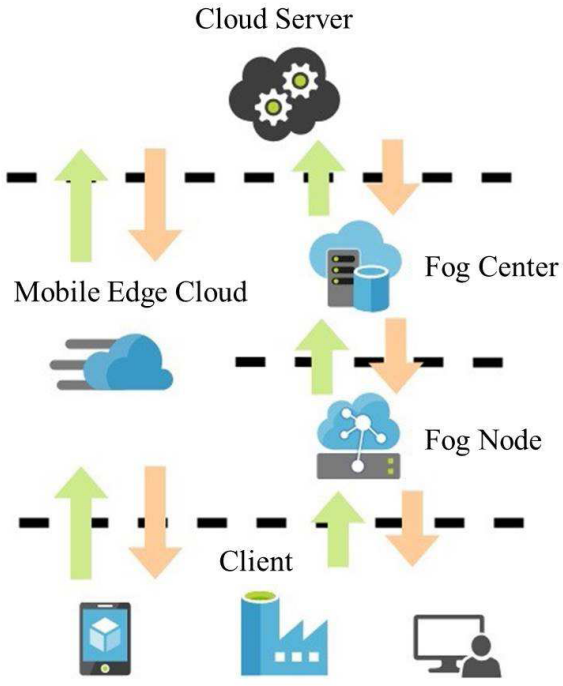

FIGURE 1. System model.

obfuscated using C-RR and transmitted to the nearest fog node.

- Fog node: The fog node is responsible for computing and analyzing randomized data from the client side using a machine learning model. The learning model can perform data prediction and return the required query answers when the query is initiated from the fog center.
- Fog center: The fog center is a bridge for communication between the cloud server and the fog node. It is responsible for three tasks: (1) collecting query requests from the cloud server and sending requests to fog nodes; (2) aggregating query results from the fog nodes and sending the results to the originating server; and (3) updating the parameters of the learning network model and sending the updated versions to fog nodes.
- Mobile edge cloud: The mobile edge cloud mainly conducts mobile edge computing for clients on the mobile network. It is responsible for computing and analyzing randomized data from the client side using a machine learning model to predict data and respond to required queries.
- Cloud server: The cloud server functions as a data manager or service provider. It has the ability to access query results. It can also process and compute data for different query requests.

B. DATA AGGREGATION PROTOCOL

In general, to avoid privacy leaks and centralized attacks, a data aggregation protocol must be able to provide data aggregation in one round of communication without leaking sensitive information to any entity. Fig. 2 shows the process of data aggregation. The entire aggregation process can be split into two processes. The first is data collection, and the second is query processing.

1) PROCESS OF DATA COLLECTION

Because collected data are protected in terms of data privacy and defended against centralized attacks, the data are transmitted to the fog node after being randomized, and the data prediction model is trained on an honest-but-curious fog node. When a user initiates a query request, the fog center generates the matching query results from the prediction model. The detailed processes are described as follows:

Data collection. The client-side entities collect all useful information according to demand.

Data disturbance. The data V collected from client sides are encoded through C-RR to produce the randomized data S', thus satisfying local differential privacy. Finally, the S' are transmitted to nearby or specific fog nodes.

Data prediction model training. The fog nodes add the received S' to the learning model M and repeatedly train M until the verification accuracy meets the established requirements.

Shared weights of model updating. The fog center and fog nodes constitute a distributed learning network architecture. The weight of learning model $w(t)$ in this architecture is shared; we call the weight defining the feature map the

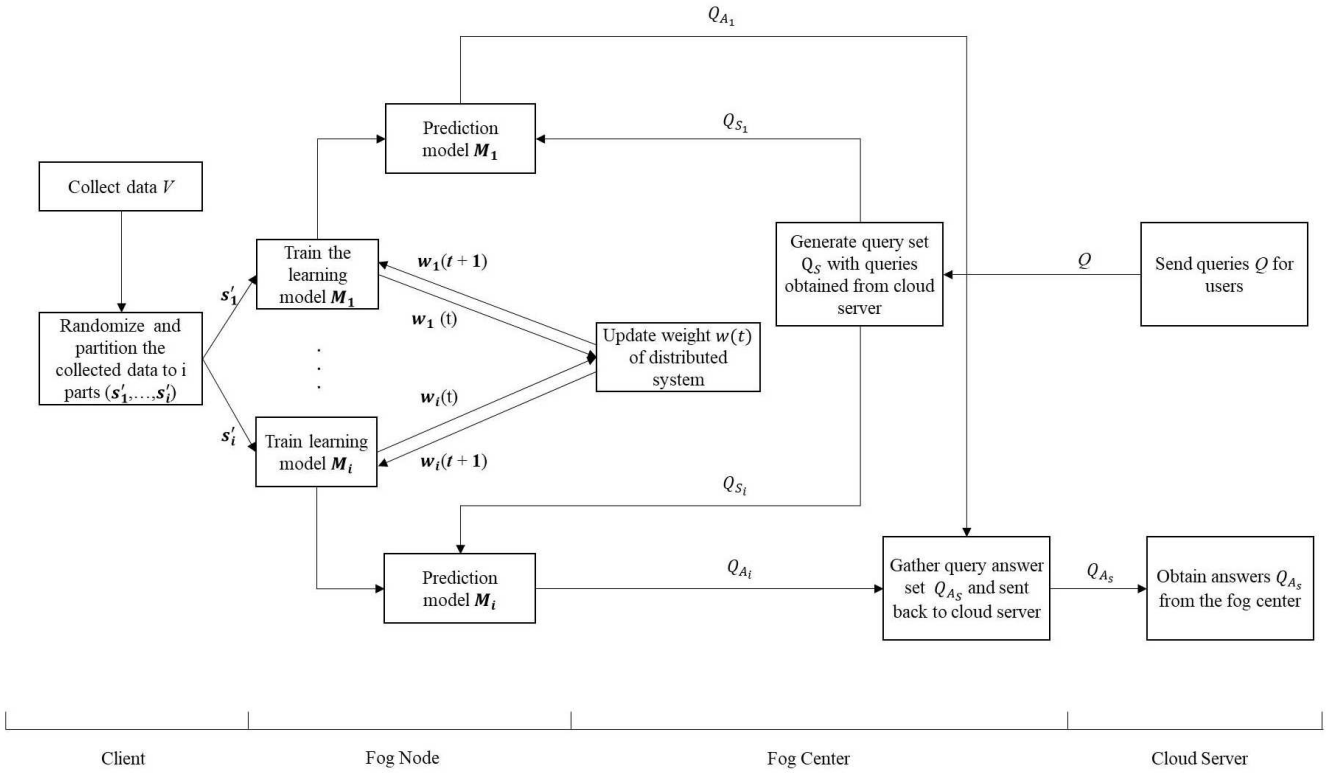

FIGURE 2. Data aggregation protocol.

"shared weight." Therefore, the shared weight $w(t)$ of a trained learning model in a fog node must be uploaded to the fog center. When the fog center receives the weight from the fog node, it updates the weight to a new shared weight $w(t+1)$ and sends $w(t+1)$ to the learning model of a fog node.

2) PROCESS OF QUERY RESULT

A query request initiated from the cloud server is transmitted to a fog node, and a query result provided by the fog node is transmitted back to the cloud server. The detailed processes are described as follows:

Generate the query request. When a user launches a query request Q using a cloud service, the cloud server must notify the fog center to cope with the query request. After collecting a set of query requirements Q_S, the fog center sends Q_S to the corresponding fog nodes.

Generate query results. The fog nodes receive the query request Q_{S_i} sent from the fog center and analyze the query results using the prediction model M_i.

Respond query results. The fog center transmits the gathered query result set Q_{A_S} to the cloud server after a specific period. Because the cloud server knows the originator of each query, it can send each query result Q_{A_i} as a response to the corresponding originator of each query.

C. ATTACK MODEL

In crowd sensing and collection modes, client-side private data can be disclosed through many methods. Assuming that the cloud server and fog computing center are honest-but-curious entities, they may disclose client-side private data unintentionally by releasing data analyses or may violate privacy intentionally by gathering sensitive data. Various attack types are available: for example, the attacker may poach data stored on cloud/fog servers or attempt to eavesdrop on communication between clients and cloud servers. The fog nodes and mobile edge cloud are semitrusted. Specifically, they are curious about the aggregated data but are not able to collude with each other. For remedying these attacks, we recommend a local privacy preservation method implemented on each client; we further recommend sanitizing any data item before it is outsourced by the client. A local privacy-preserving method satisfying the definition of ε-differential privacy (called local differential privacy) can have a rigorous privacy guarantee, regardless of the attackers' background knowledge.

D. NOTATIONS

- $h-$ number of hash functions.
- $k-$ size of Bloom filter.
- $p, q,$ and $f-$ probability parameters for the degree of data privacy.
- $b_i, b_i', b_i^*, s_i,$ and $s_i'-$ resultant bits of Bloom filter, PRR, COP, IRR, and COI, respectively.
- $\varepsilon-$ privacy budget of differential privacy.
- c^*- probability of generating 1 in the response b_i^*, when the bit b_i of Bloom filter B is set to 1.
- z^*- probability of generating 1 in the response s_i', when the bit b_i of Bloom filter B is set to 1.
- $z'-$ probability of generating 1 in the response s_i', when the bit b_i of Bloom filter B is set to 0.
- z_1- probability of generating 1 in the response s_i', if $b_i^* = 1$ and $s_i = 1$.
- z_2- probability of generating 1 in the response s_i', if $b_i^* = 0$ and $s_i = 1$.
- z_3- probability of generating 1 in the response s_i', if $b_i^* = 1$ and $s_i = 0$.
- z_4- probability of generating 1 in the response s_i', if $b_i^* = 0$ and $s_i = 0$.

IV. COMPONENTS OF PPDCA

In this section, we describe two key components of PPDCA: encoding of data through C-RR and prediction of randomized data using machine learning mechanisms.

A. DATA ENCODING THROUGH C-RR

To improve the accuracy of data analysis, an intuitive approach is to reserve numerous features from the original data without compromising data privacy. Accordingly, the concept of C-RR is to use the permanent randomized response (PRR), complementary PRR (COP), instantaneous randomized response (IRR), and complementary IRR (COI) to reserve the features from b_i within s_i' while maintaining the randomness of the encoded data.

In the following, the process of C-RR is illustrated from the view of coin flips. Initially, each client generates a k-size Bloom filter B by using h hash functions to hash his or her data string v. Next, each bit $b_i \in B$ is released after six rounds of perturbation determined by flipping elaborated coins. We describe the probability of each coin in Table 1 and illustrate the details of each design as follows. The first and second rounds, which are called PRR, are obtained from the perturbation of each bit b_i in B. The result of PRR, b_i', is generated by operating both an unfair coin 1 that comes up as heads with probability $1-f$ and a fair coin 2. Specifically, if the result of the coin 1 flip is tails, then the result of PRR is determined by the coin 2 with the probability $\frac{1}{2}$. Otherwise, we just set b_i' to the actual value of b_i. The underlying result of PRR can be used to prevent robust statistical analysis of longitudinal data from attackers [7]. The third and fourth

TABLE 1. Probabilities of coin flips, in which $f \in [0, 1)$, $p \in (0, 1)$, $q \in (0, 1)$, and $p \neq q$.

Bit String		Head	Tail
Bloom filter bit (b_i)	-	-	-
PRR (b_i')	Coin 1	$1-f$	f
	Coin 2	$\frac{1}{2}$	$\frac{1}{2}$
COP (b_i^*)	Coin 3	$\frac{f}{2}$	$1-\frac{f}{2}$
	Coin 4	$1-\frac{f}{2}$	$\frac{f}{2}$
IRR (s_i)	Coin 5	$q^{b_i^*} \cdot (1-p)^{1-b_i^*}$	$(1-q)^{b_i^*} \cdot p^{1-b_i^*}$
COI (s_i')	Coin 6	$z_1^{b_i^* s_i} \cdot (1-z_2)^{(1-b_i^*)s_i} \cdot (1-z_3)^{b_i^*(1-s_i)} \cdot z_4^{(1-b_i^*)(1-s_i)}$	$(1-z_1)^{b_i^* s_i} \cdot z_2^{(1-b_i^*)s_i} \cdot z_3^{b_i^*(1-s_i)} \cdot (1-z_4)^{(1-b_i^*)(1-s_i)}$

rounds, which are called COP, consist of b_i and b'_i to keep b_i^* and b_i as identical as possible. The result of COP, b_i^*, is generated by operating both an unfair coin 3 that comes up as heads with probability $\frac{f}{2}$ and an unfair coin 4 that comes up as heads with probability $1-\frac{f}{2}$. On the basis of b'_i and b_i, b_i^* can be formatted as follows:

$$\forall i \in k, \quad Pr(b_i^* = 1) = (\frac{f}{2})^{b_i} + (1-\frac{f}{2})^{b'_i},$$

where probability $Pr(b_i^* = 1)$ can be dynamically adjusted with the parameter f to reserve more features that can be analyzed in our machine learning model. The intuition of tossing coins 3 and 4 is to design the probabilities of these coins through a synthetic consideration of b_i and b'_i. The more frequently 1 occurs in b_i and b'_i, the more probable it is that the coin will be heads.

The fifth round, which is called IRR, is devised to provide protection against possible tracking attacks [28] and is defined as follows:

$$\forall i \in k, \quad Pr(s_i = 1) = \begin{cases} q, & if\ b_i^* = 1 \\ p, & if\ b_i^* = 0 \end{cases},$$

where b_i^* affects the probability of coin 5 for response s_i. If b_i^* is equal to 1, then the probability of heads is q; otherwise, the probability of heads is p.

Eventually, the final round, which is called COI, is established in accordance with COP and IRR and is defined as follows:

$$\forall i \in k, \quad Pr(s'_i = 1) = \begin{cases} z_1, & if\ b_i^* = 1\ and\ s_i = 1 \\ z_2, & if\ b_i^* = 0\ and\ s_i = 1 \\ z_3, & if\ b_i^* = 1\ and\ s_i = 0 \\ z_4, & if\ b_i^* = 0\ and\ s_i = 0. \end{cases}$$

The intuition of COI is to retain numerous features in b_i^* as reserved in s'_i while maintaining the randomness of the results. We design the probability of this coin through a synthetic consideration of b_i^* and s_i. For $s_i = 1$, if b_i^* is equal to 1, then the probability of heads is z_1; otherwise, if b_i^* is equal to 0, then the probability of heads is z_2. For $s_i = 0$, if b_i^* is equal to 1, then the probability of heads is z_3; otherwise, if b_i^* is equal to 0, then the probability of heads is z_4. Therefore, we can employ s'_i to reformulate client-side strings effectively.

We quantitatively interpret coin flips from the view of conditional probability; that is, each round of coin flipping operates under the conditions of $b_i = 1$ and $b_i = 0$ and is formulated according to the following lemmas:

Lemma 1: If the bit b_i in Bloom filter B is set to 1, then the probability of generating 1 in the response b_i^* is given by

$$\begin{aligned} c^* &= Pr(b_i^* = 1|b_i = 1) \\ &= \frac{f}{2}Pr(b_i = 1|b_i = 1) + \left(1-\frac{f}{2}\right)Pr(b'_i = 1|b_i = 1) \\ &= \frac{f}{2} + \left(1-\frac{f}{2}\right)\left(1-\frac{f}{2}\right) \\ &= 1 - \frac{f}{2} + \frac{f^2}{4}. \end{aligned}$$

Lemma 2: If the bit b_i in Bloom filter B is set to 0, then the probability of generating 1 in the response b_i^* is given by

$$\begin{aligned} &Pr(b_i^* = 1|b_i = 0) \\ &= \frac{f}{2}Pr(b_i = 1|b_i = 0) + \left(1-\frac{f}{2}\right)Pr(b'_i = 1|b_i = 0) \\ &= \left(1-\frac{f}{2}\right)\cdot\frac{f}{2} \\ &= \frac{f}{2} - \frac{f^2}{4} \\ &= 1 - c^*. \end{aligned}$$

Lemma 3: If the bit b_i in Bloom filter B is set to 1, then the probability of generating 1 in the response s'_i is given by

$$\begin{aligned} z^* &= Pr(s'_i = 1|b_i = 1) \\ &= z_1 Pr(s_i = 1|b_i^* = 1, b_i = 1) \\ &\quad + z_2 Pr(s_i = 1|b_i^* = 0, b_i = 1) \\ &\quad + z_3 Pr(s_i = 0|b_i^* = 1, b_i = 1) \\ &\quad + z_4 Pr(s_i = 0|b_i^* = 0, b_i = 1) \\ &= z_1 q_1^* + z_2 p_1^* + z_3 q'_1 + z_4 p'_1. \end{aligned}$$

Lemma 4: If the bit b_i in Bloom filter B is set to 0, then the probability of generating 1 in the response s'_i is given by

$$\begin{aligned} z' &= Pr(s'_i = 1|b_i = 0) \\ &= z_1 Pr(s_i = 1|b_i^* = 1, b_i = 0) \\ &\quad + z_2 Pr(s_i = 1|b_i^* = 0, b_i = 0) \\ &\quad + z_3 Pr(s_i = 0|b_i^* = 1, b_i = 0) \\ &\quad + z_4 Pr(s_i = 0|b_i^* = 0, b_i = 0) \\ &= z_1 q_2^* + z_2 p_2^* + z_3 q'_2 + z_4 p'_2, \end{aligned}$$

where

$$\begin{aligned} q_1^* &= Pr(s_i = 1|b_i^* = 1, b_i = 1) = qc^* \\ q'_1 &= Pr(s_i = 1|b_i^* = 1, b_i = 0) = (1-q)c^* \\ q_2^* &= Pr(s_i = 1|b_i^* = 1, b_i = 0) = q(1-c^*) \\ q'_2 &= Pr(s_i = 0|b_i^* = 1, b_i = 0) = (1-q)(1-c^*) \\ p_1^* &= Pr(s_i = 1|b_i^* = 0, b_i = 1) = p(1-c^*) \\ p'_1 &= Pr(s_i = 0|b_i^* = 0, b_i = 1) = (1-p)(1-c^*) \\ p_2^* &= Pr(s_i = 1|b_i^* = 0, b_i = 0) = pc^* \\ p'_2 &= Pr(s_i = 0|b_i^* = 0, b_i = 0) = (1-p)c^*. \end{aligned}$$

B. PREDICTION OF RANDOMIZED DATA USING MACHINE LEARNING MODEL

Because the prediction of a client-side strings is a nonlinear multiclassification problem, the prediction network model is implemented in accordance with a multilayer perceptron (MLP) [11]. An MLP is a feed-forward neural network including three network layer structures, namely an input layer, a hidden layer, and an output layer, in which each layer is connected to the next layer. We can understand the stacked layers of an MLP as a multilayer stack or a multiperceptron stack, similar to multiple regression methods or linear

classifier layer stacking. Moreover, the hidden layer is used to capture data characteristics and increase or decrease data dimensions.

The learning model of an MLP is a supervised learning scheme. Moreover, the training network is known as error-backward propagation for the correction of learning errors. Our training and prediction processes follow two steps:

Step 1 (Feed-Forward Propagation): Because our model is basically a supervised learning scheme, it is necessary to artificially supplement the process with a known training data set. The input and output data sets are both known; the shared weight between two neurons is trained. The details of settings are illustrated in Section VI.

In a feed-forward network, a certain neuron is equal to the sum of all the neurons in the previous layer multiplied by the shared weight, and the predicted result can be obtained through an activation function: $v^{pred} = \sigma(\cdot)$. The activation function must be a nonlinear differentiable function. If it is linear, then the results of multilayer and single-layer systems are the same. If it is nondifferentiable, then the shared weight cannot be adjusted. In fact, this activation function enables error-backward propagation to handle nonlinear classification and train the shared weights.

Because the predicted probability of data $S' = \{s'_i\}_{i=1}^{k}$ in our system is a multiclassification prediction, we use softmax as the activation function $\sigma(\cdot)$ for our learning model and formulate it as follows:

$$v^{pred} = \sigma(S')_i = \frac{e^{s'_i}}{\sum_{j=1}^{k} e^{s'_j}}, \quad i = 1, 2, \ldots, k,$$

where $s'_i = w_i^T x + \beta$ given a sample vector x, a shared weight vector w, and bias β.

Step 2 (Error-Backward Propagation): When the system is trained with data for the first time, the model may misclassify the target value, and thus subsequent training data may be classified in the wrong direction. This problem can be rectified with a known actual value. When the prediction result is different from the actual result, the error can be calculated. From the output layer, the error is passed back to the input layer, and the shared weights are readjusted during the transfer process.

In addition, we use a categorical cross-entropy function as a loss function $L(\cdot, \cdot)$ to evaluate the quality of prediction when training the network for the predicted string vector $v^{pred} = [b_1^{pred}, b_2^{pred}, \ldots, b_k^{pred}]$ and the actual string vector $v^{act} = [b_1, b_2, \ldots, b_k]$:

$$L(v^{pred}, v^{act}) = -\sum_{i=1}^{k} b_i log(b_i^{pred}).$$

If $L(v^{pred}, v^{act})$ is less than a given threshold, then v^{pred} is returned as output; otherwise, the learning model must be trained again.

V. ε-DIFFERENTIAL PRIVACY ANALYSIS

The differential privacy guarantee is used to assess the degree of privacy protection of the data. We prove that COP and COI satisfy ε-differential privacy in this section.

A. DIFFERENTIAL PRIVACY GUARANTEE OF COP

Theorem 1: COP satisfies ε_p-differential privacy, where

$$\varepsilon_p = 2hln\left(\frac{c^*}{1-c^*}\right).$$

Proof: Let $V = \{v_1, v_2, \ldots, v_n\}$ be the original strings collected from clients; let $B = \{b_1, b_2, \ldots, b_k\}$ be the Bloom filter converted from a single string of V; let $B' = \{b'_1, b'_2, \ldots, b'_k\}$ be the noisy version of B; let $B^* = \{b_1^*, b_2^*, \ldots, b_k^*\}$ be the probable synthetic version of B and B'; let $S = \{s_1, s_2, \ldots, s_k\}$ be the noisy version of B^*; and let $S' = \{s'_1, s'_2, \ldots, s'_k\}$ be the final complementarily encoded reports. Then, the probability of S' given V is:

$$\begin{aligned}
&Pr(S' = s'|V = v)\\
&\quad = Pr(S' = s'|S = s, B^* = b^*, B' = b', v)\\
&\qquad \cdot Pr(S = s|B^* = b^*, B' = b', B = b, v)\\
&\qquad \cdot Pr(B^* = b^*|B' = b', B = b, v)\\
&\qquad \cdot Pr(B' = b'|B = b, v) \cdot Pr(B = b|v)\\
&\quad = Pr(S' = s'|S = s, B^* = b^*)\\
&\qquad \cdot Pr(B^* = b^*|B' = b', B = b)\\
&\qquad \cdot Pr(B = b|v)\\
&\quad = Pr(S' = s'|S = s, B^* = b^*)\\
&\qquad \cdot Pr(B^* = b^*|B' = b', B = b),
\end{aligned}$$

where S' is conditionally independent of B.

Moreover, we assume that v_a and v_b are two distinct strings, and their bits in B are set as follows:

$$\begin{aligned}
B_a &= \{b_1 = 1, \ldots, b_h = 1, b_{h+1} = 0, \ldots, b_k = 0\},\\
B_b &= \{b_1 = 0, \ldots, b_h = 0, b_{h+1} = 1, \ldots, b_{2h} = 1,\\
&\quad b_{2h+1} = 0, \ldots, b_k = 0\},
\end{aligned}$$

where h is the number of hash functions and k is the number of bits in B. The probability mass functions under different conditions are as follows:

$$Pr(b_i^*|b_i = 1) = (c^*)^{b_i^*}(1-c^*)^{1-b_i^*} = \begin{cases} c^*, & if\ b_i^* = 1 \\ 1-c^*, & if\ b_i^* = 0, \end{cases}$$

and

$$Pr(b_i^*|b_i = 0) = (1-c^*)^{b_i^*}(c^*)^{1-b_i^*} = \begin{cases} 1-c^*, & if\ b_i^* = 1 \\ c^*, & if\ b_i^* = 0. \end{cases}$$

Then,

$$\begin{aligned}
P(B^* = B_a^*|B = B_a) = &\prod_{i=1}^{h}(c^*)^{b_i^*}(1-c^*)^{1-b_i^*}\\
&\cdot \prod_{i=h+1}^{k}(1-c^*)^{b_i^*}(c^*)^{1-b_i^*},
\end{aligned}$$

and

$$P(B^* = B_b^*|B = B_b) = \prod_{i=1}^{h}(1-c^*)^{b_i^*}(c^*)^{1-b_i^*} \cdot \prod_{i=h+1}^{2h}(c^*)^{b_i^*}(1-c^*)^{1-b_i^*} \cdot \prod_{i=2h+1}^{k}(1-c^*)^{b_i^*}(c^*)^{1-b_i^*}.$$

Let $\mathcal{RP}_{\mathcal{B}^*}$ be the ratio of two conditional probabilities and let R_{B^*} be all possible outputs of B^*. According to Observation 1 in [28], we can formulate

$$\begin{aligned}\mathcal{RP}_{\mathcal{B}^*} &= \frac{P(B^* \in R_{B^*}|B = B_a)}{P(B^* \in R_{B^*}|B = B_b)} \\ &= \frac{\sum_{B_i^* \in R_{B^*}} Pr(B^* = B_i^*|B = B_a)}{\sum_{B_i^* \in R_{B^*}} Pr(B^* = B_i^*|B = B_b)} \\ &\leq \max_{B_i^* \in R_{B^*}} \frac{Pr(B^* = B_i^*|B = B_a)}{Pr(B^* = B_i^*|B = B_b)} \\ &= \max_{B_i^* \in R_{B^*}} \left\{(c^*)^{2(b_1^*+\ldots+b_h^*-b_{h+1}^*-\ldots-b_{2h}^*)} \cdot \left[(1-c^*)\right]^{2(-b_1^*-\ldots-b_h^*+b_{h+1}^*+\ldots+b_{2h}^*)}\right\} \\ &= \left(\frac{c^*}{1-c^*}\right)^{2h},\end{aligned}$$

where $b_1^* = \ldots = b_h^* = 0$ and $b_{h+1}^* = \ldots = b_{2h}^* = 1$ cause sensitivity to be maximized.

To guarantee the protection of differential privacy, $\mathcal{RP}_{\mathcal{B}^*}$ must be bounded by e^{ε_p}. As a result, the privacy budget ε_p can be calculated as: $\varepsilon_p = 2hln\left(\frac{c^*}{1-c^*}\right)$. □

B. DIFFERENTIAL PRIVACY GUARANTEE OF COI

Theorem 2: COI satisfies ε_I-differential privacy, where

$$\varepsilon_I = hln\left[\frac{z^*(1-z')}{z'(1-z^*)}\right].$$

Proof: We know that s_i' is a random variable with a Bernoulli distribution in accordance with Lemmas 3 and 4, and the probability mass functions under different conditions are as follows:

$$Pr(s_i'|b_i = 1) = (z^*)^{s_i'}(1-z^*)^{1-s_i'} = \begin{cases} z^*, & if\ s_i' = 1 \\ 1-z^*, & if\ s_i' = 0, \end{cases}$$

and

$$Pr(s_i'|b_i = 0) = (z')^{s_i'}(1-z')^{1-s_i'} = \begin{cases} z', & if\ s_i' = 1 \\ 1-z', & if\ s_i' = 0. \end{cases}$$

Let $RP_{S'}$ be the ratio of two conditional probabilities and let $R_{S'}$ be all possible outputs of S'.

$$\begin{aligned}\mathcal{RP}_{\mathcal{S}'} &= \frac{P(S' \in R_{S^*}|B = B_a)}{P(S' \in R_{S'}|B = B_b)} \\ &= \frac{\sum_{S_i' \in R_{S'}} Pr(S' = S_i'|B = B_a)}{\sum_{S_i' \in R_{S'}} Pr(S' = S_i'|B = B_b)} \\ &\leq \max_{S_i' \in R_{S'}} \frac{Pr(S' = S_i'|B = B_a)}{Pr(S' = S_i'|B = B_b)} \\ &= \max_{S_i' \in R_{S'}} \left\{[z^*(1-z')]^{s_1'+\ldots+s_h'-s_{h+1}'-\ldots-s_{2h}'} \cdot [z'(1-z^*)]^{2(-s_1'-\ldots-s_h'+s_{h+1}'+\ldots+s_{2h}')}\right\} \\ &= \left[\frac{z^*(1-z')}{z'(1-z^*)}\right]^h.\end{aligned}$$

To guarantee the protection of differential privacy, $RP_{S'}$ needs to be bounded by e^{ε_I}. As a result, the privacy budget ε_I can be calculated as: $\varepsilon_I = hln\left[\frac{z^*(1-z')}{z'(1-z^*)}\right]$. □

VI. LEARNING MODEL CONSTRUCTION

Before detailing the construction of our learning model, we describe the preparation of the input of our learning model and review the cross-validation procedures. A k-bit Bloom filter is used to represent a true client-side string, in which h hash functions are used to encode the client-side string individually and obtain h array positions. Subsequently, the bits at these particular positions in the Bloom filter are set to 1. These 1s are represented as the features of this string. After operating the C-RR, a noisy version of the client-side string is generated. During the training of our learning network, each noisy string is labeled with its true string locally. In cross-validation procedures, we used 70% of the targeted data set, such as Kosarak and MHEALTH, as the training data set and used 30% of the data set as the validation data set. After the training and validation processes following feed-forward propagation and error-backward propagation, our learning model can be completed. The implementation of our learning model is described in detail as follows.

For constructing our learning model, we tested the number of hidden layers to determine the number of hidden layers that would be optimal for our model; 32-bit Bloom filters were input as validated targets and the probabilities of 100 categories were returned as predicted results. Fig. 3 illustrates cross-validation results under different numbers of hidden layers with randomized data as input. By observing the experimental results, we found that systems with two hidden layers demonstrated superior convergence. As the iterations proceeded, the output approached a specific value. For a system with two hidden layers, the number of nodes in each layer is listed in Table 2.

TABLE 2. The number of nodes for each layer.

	input layer	hidden layer 1	hidden layer 2	output layer
# of nodes	32	1024	256	100

We applied general optimization algorithms (*i.e.*, RMSprop, AdaMax, SGD, Nadam, Adam, and Adagrad) [23] for testing the optimization of our prediction model. Fig. 4 demonstrates the test results. As indicated in Fig. 4, RMSprop and AdaMax could not converge to a specific value in our model for cross validation. Nadam was unstable in cross

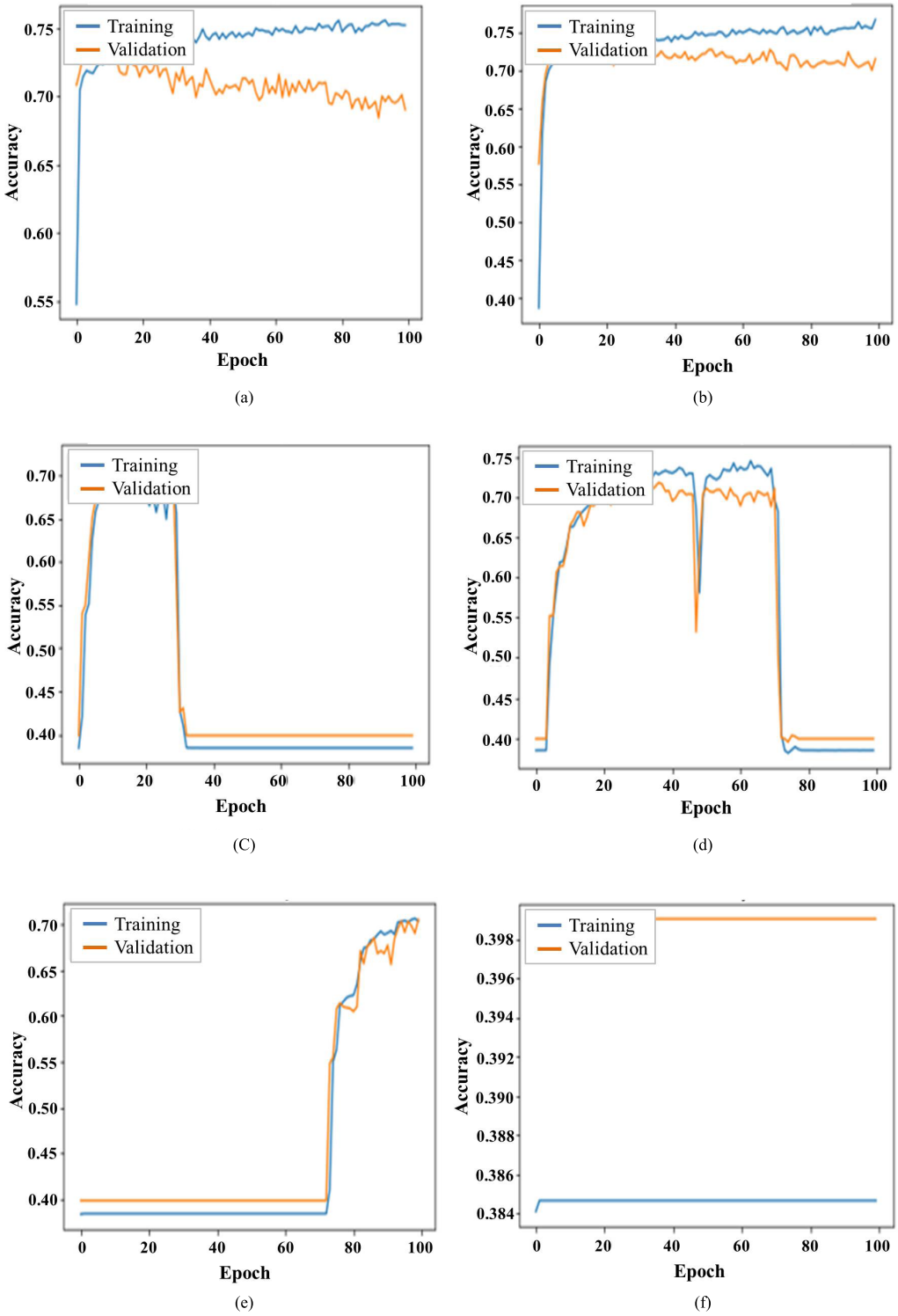

FIGURE 3. Cross-validation results under different numbers of hidden layers. (a) One layer. (b) Two layers. (c) Three layers. (d) Four layers. (e) Five layers. (f) Six layers.

validation, whereas Adagrad had an overfitting problem. By contrast, SGD demonstrated superior convergence; SGD outperformed Adam. Therefore, we used SGD as the optimization algorithm in our model. Because our prediction model was designed to address a multiclassification problem, we used the categorical cross-entropy function as a loss function along with the softmax algorithm as our activation function in the output layer, as described in Section IV-B.

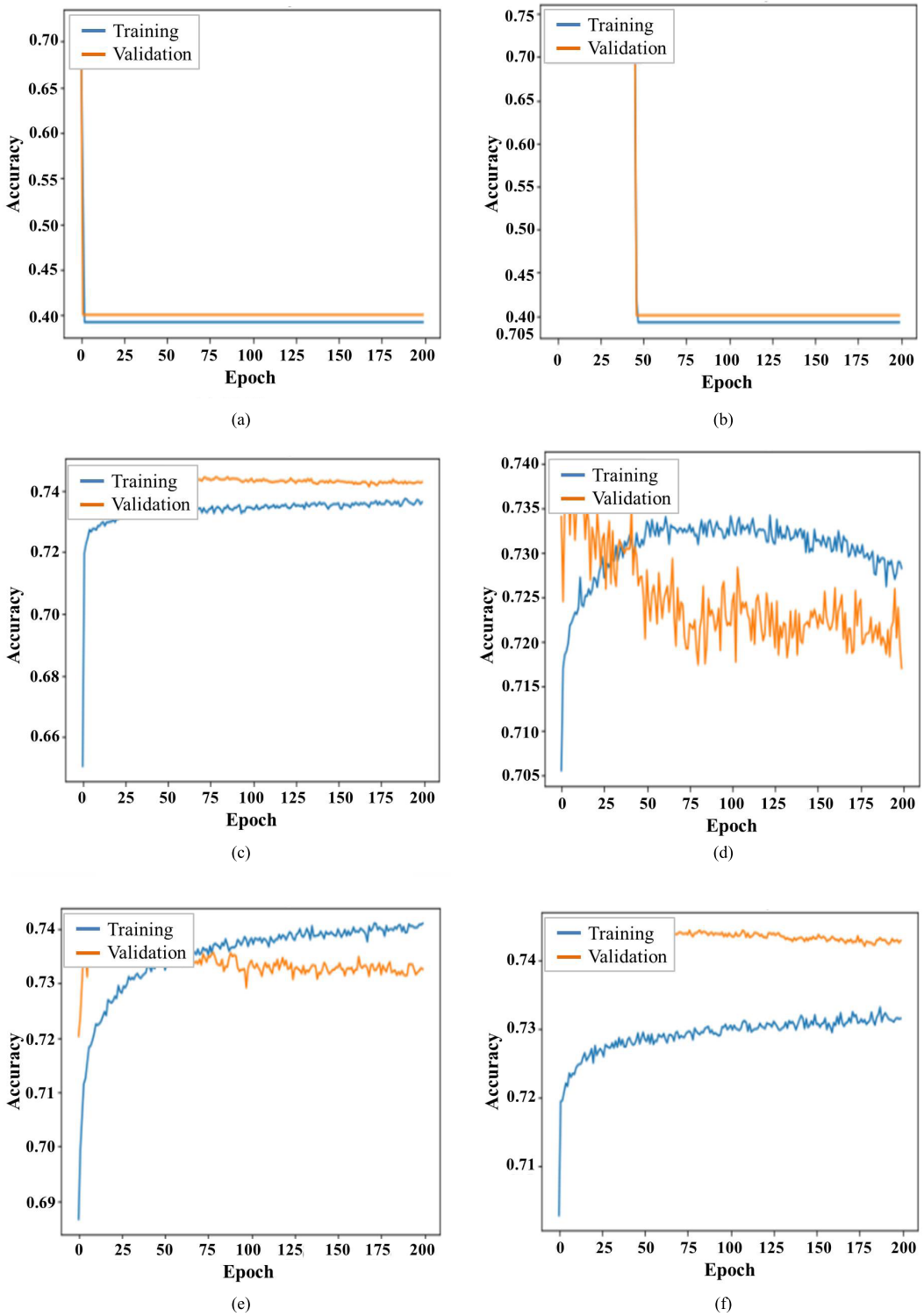

FIGURE 4. Optimization algorithms testing for our prediction model. (a) RMSprop. (b) AdaMax. (c) SGD. (d) Nadam. (e) Adam. (f) Adagrad.

After constructing our data prediction model, we performed a series of extensive experiments that compared PPDCA with state-of-the-art methods, as described in the next section.

VII. EXPERIMENTAL EVALUATION

Fanti *et al.* [14] estimated client-side strings without explicit dictionary knowledge of RAPPOR. The aim of other methods, such as [3], [26], [30], and [32], is to reduce transmission

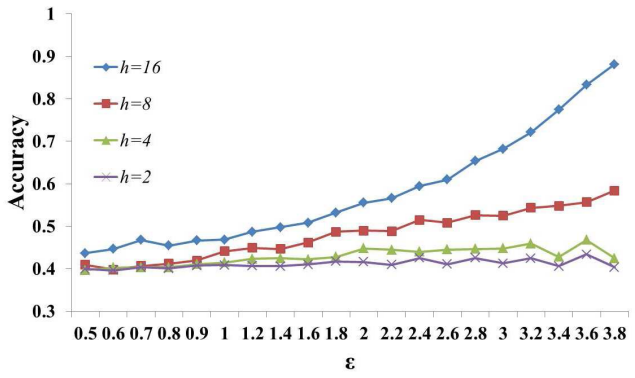

FIGURE 5. Accuracy varied with the parameters h and ε at $f = 0.5$, $q = 0.75$, $p = 0.5$, $z_1 = 0.8$, $z_2 = 0.2$, $z_3 = 0.8$, and $z_4 = 0.2$.

cost or restrict the accuracy of their estimates depending on the variable sizes of users' strings. However, the estimation accuracy of these methods is similar to or less than that of RAPPOR. Therefore, we present a detailed comparison of RAPPOR and PPDCA for heavy-hitters discovery in this section. In addition, the learning model of PPDCA is similar to that of MLDP. Hence, we also present a comparison of PPDCA and MLDP in terms of the accuracy of prediction for various values of ε.

We used Kosarak [1] and MHEALTH [19] as our test data for multiclassification prediction. Comparing PPDCA with RAPPOR, we conducted our experiments on the real-world data set Kosarak. For a fair comparison of PPDCA and MLDP, our experiments were conducted on the real-world data set MHEALTH. Kosarak, which was provided by Ferenc Bodon [1], records information from approximately 990, 000 clients; the information involves more than 3, 200, 000 click actions within 41, 270 different pages. Various web masters may seek to discover the popularity of each page through the estimation of clicks. MHEALTH, which is a mobile health data set provided by Oresti Banos [19], contains more than one million records, each comprising the data from 24 different sensor signals. Because each signal is at the same scale, we randomly chose one type of signal for evaluation.

In the following sections, we first demonstrate the effects of the parameters h, z_1, z_2, z_3, and z_4 on the accuracy of predicting original strings in our method. Subsequently, we compare the accuracy of PPDCA with that of RAPPOR for varying f, p, and q values and demonstrate the population of client-side strings reconstructed by PPDCA and RAPPOR. Finally, PPDCA is compared with MLDP for various values of ε for the prediction accuracy based on the machine learning model.

A. EFFECT OF SYSTEM PARAMETERS

The number of hash functions h and probability parameters z_1, z_2, z_3, and z_4 significantly affected the degree of differential privacy guaranteed by our system.

First, we determined that the accuracy varied with h at $f = 0.5$, $q = 0.75$, $p = 0.5$, $z_1 = 0.8$, $z_2 = 0.2$, $z_3 = 0.8$, and $z_4 = 0.2$, as presented in Fig. 5. Fig. 5 reveals that if a system has a large number of hash functions, calculations

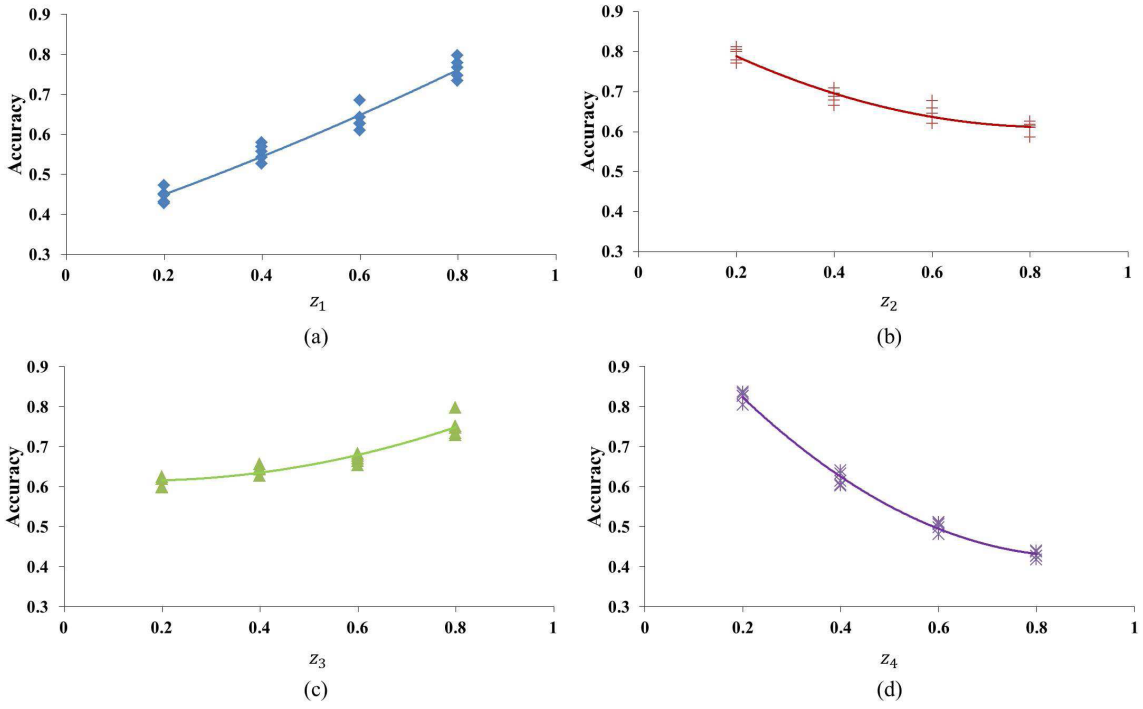

FIGURE 6. Effects of z_1, z_2, z_3, and z_4 varying from 0.2 to 0.8 on accuracy at $f = 0.5$, $q = 0.75$, $p = 0.5$, and $h = [2, 4, 8, 16]$.

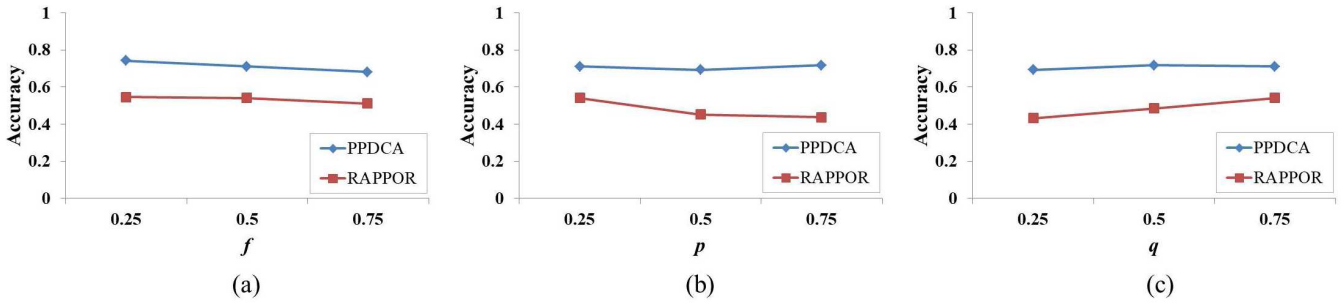

FIGURE 7. Comparison of the accuracy of PPDCA with that of RAPPOR at $h = 2$, $z_1 = 0.2$, $z_2 = 0.8$, $z_3 = 0.2$, and $z_4 = 0.8$, where (a) $q = 0.75$ and $p = 0.25$; (b) $f = 0.5$ and $q = 0.75$; and (c) $f = 0.5$ and $p = 0.25$.

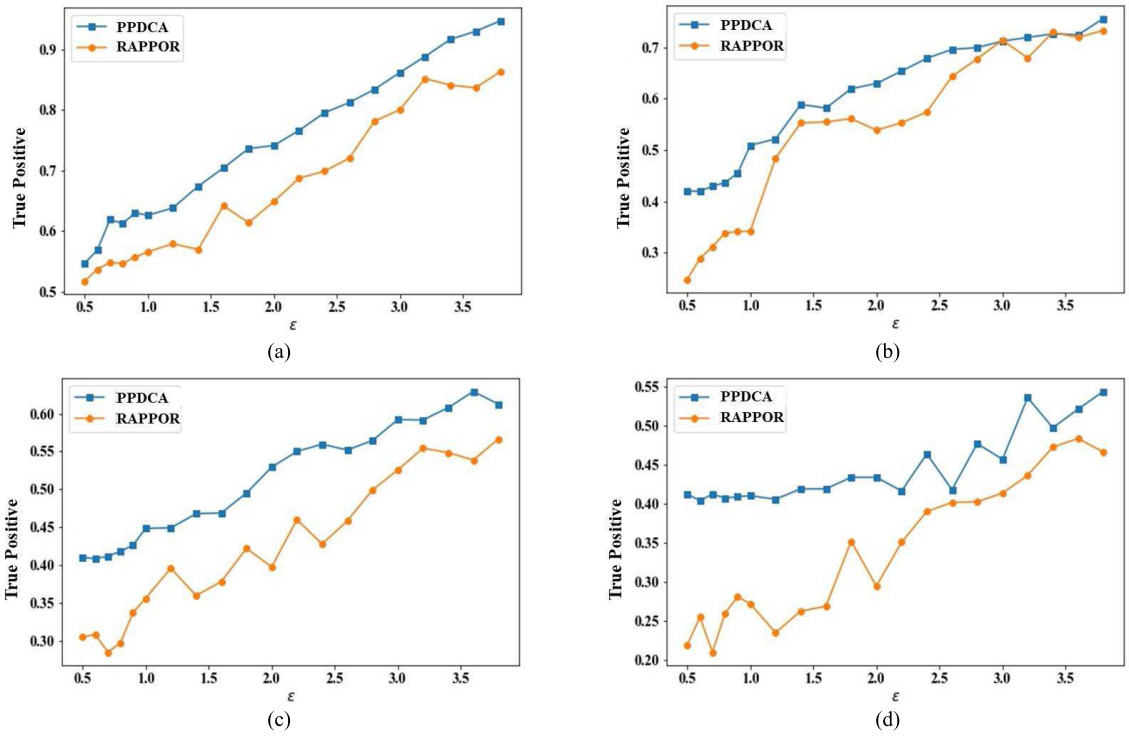

FIGURE 8. Comparison of the true positive rate of population detection for RAPPOR and PPDCA. (a) $h = 2$, (b) $h = 4$, (c) $h = 8$, (d) $h = 16$ in varying ε from 0.5 to 3.8.

could produce numerous 1 values in a typical bit string, which would produce numerous features from the original data; thus, we can predict the corresponding original string with high accuracy. For each h, we varied ε from 0.5 to 3.8. Notably, the number of hash functions affected the value of ε, according to Theorem 2; that is, the higher the value of h was, the higher the value of ε was. In terms of the differential privacy definition, the higher the value of ε was, the lower the degree of guaranteed privacy was. Consequently, in the selection of the number of hash functions, which directly affects the setting of the value of ε, one faces a puzzle between data utility and data privacy.

Apart from the parameter h, the parameters z_1, z_2, z_3, and z_4 affected the accuracy of data prediction. Fig. 6 demonstrates the effect of z_1, z_2, z_3, and z_4 on accuracy at $f = 0.5$, $q = 0.75$, and $p = 0.5$, where each z_i ($i \in [1, 2, 3, 4]$) for each probability (*i.e.*, 0.2, 0.4, 0.6, and 0.8) was executed four times by setting $h = [2, 4, 8, 16]$. The parameters z_1, z_2, z_3, and z_4 can be used to control how many features retained in S' are reserved from B. If the settings of z_1, z_2, z_3, and z_4 increase the similarity of randomized data S' to the Bloom filter B, the predictability of S' can be increased.

The results of the aforementioned experiments for our system privacy parameters can guide appropriate selections of system parameters to provide the required degree of analytical power and to guarantee the required degree of privacy.

B. COMPARISON

The accuracy of data string prediction observed for both PPDCA and RAPPOR is shown in Fig. 7 at $h = 2$, $z_1 = 0.2$,

$z_2 = 0.8$, $z_3 = 0.2$, and $z_4 = 0.8$. RAPPOR can reserve a few features from b_i, but C-RR is applied in PPDCA to reserve even more features from b_i; thus, our prediction model can rebuild original strings with high accuracy even if the parameters f, p, and q are varied. Compared with RAPPOR, for example, the accuracy could be improved by up to 20%, 30%, and 30% by varying f, q, and p, respectively.

Fig. 8 shows the true positive rate of population detection at $h = [2, 4, 8, 16]$ and $\varepsilon \in [0.5, 3.8]$, indicating the superiority of PPDCA over RAPPOR in detecting all possible distinct strings. Because the C-RR encoding method and the learning network model are used in PPDCA, the population of prediction results in PPDCA can accurately reconstruct correct strings. Moreover, the true positive rate depends on the overall population distribution. Therefore, Fig. 9 shows that PPDCA outperforms RAPPOR in population detection.

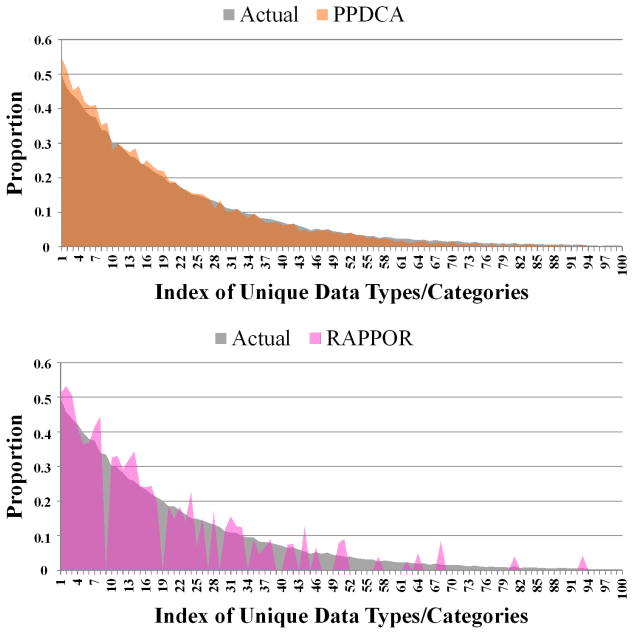

FIGURE 9. Population of client-side strings reconstructed by PPDCA and RAPPOR.

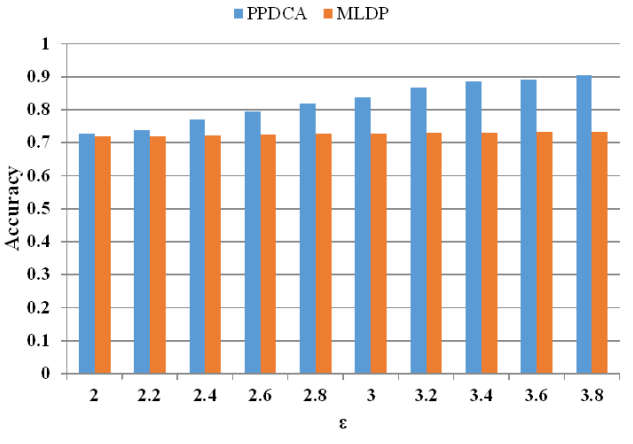

FIGURE 10. Comparison of the accuracy of PPDCA with that of MLDP at various ε.

Fig. 10 shows the accuracy of prediction results at various ε for PPDCA and MLDP. Under the same privacy level ε as used by MLDP, PPDCA uses the C-RR data encoding method to reserve more features from the original data string for the machine learning model. Thus, when ε increases, the accuracy of PPDCA exceeds that of MLDP.

VIII. CONCLUDING REMARKS

We propose a randomized response-based data protection mechanism for crowdsourcing data collection and analysis, namely PPDCA. It provides a mathematically rigorous data privacy guarantee and preserves high-utility data prediction using C-RR and our learning network model. C-RR applies six coin flips to preserve crowdsourcing data privacy. Moreover, we train the parameters in the neural network to enable high-utility data prediction through a TensorFlow learning model, which enables crowdsourced prediction on collections from individuals. Through a sequence of experiments in a real-world environment, we verified PPDCA to outperform the well-known methods RAPPOR and MLDP under different conditions.

ACKNOWLEDGMENT

The authors acknowledge Wallace Academic Editing for editing this manuscript.

REFERENCES

[1] *Kosarak*. Accessed: Aug. 2017. [Online]. Available: http://fimi.ua.ac.be/data/
[2] (2015). *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. [Online]. Available: https://www.tensorflow.org/
[3] Differential Privacy Team, Apple, "Learning with privacy at scale," *Mach. Learing J.*, vol. 1, no. 8, pp. 1–25, 2017.
[4] B.-C. Lin, S.-H. Wu, Y.-T. Tsou, and Y. Huang, "PPDCA: Privacy-preserving crowdsensing data collection and analysis with randomized response," in *Proc. IEEE Wireless Commun. Netw. Conf.*, Apr. 2018, pp. 1–6.
[5] B. Ding, J. Kulkarni, and S. Yekhanin, "Collecting telemetry data privately," in *Proc. 31st Conf. Neural Inf. Process. Syst.*, 2017, pp. 3571–3580.
[6] B. Hitaj, G. Ateniese, and F. Perez-Cruz. (2017). "Deep models under the GAN: Information leakage from collaborative deep learning." [Online]. Available: https://arxiv.org/abs/1702.07464
[7] B. Edwards, S. Hofmeyr, S. Forrest, and M. van Eeten, "Analyzing and modeling longitudinal security data: Promise and pitfalls," in *Proc. 31st Annu. Comput. Secur. Appl. Conf.*, 2015, pp. 391–400.
[8] B. H. Bloom, "Space/time trade-offs in hash coding with allowable errors," *Commun. ACM*, vol. 13, no. 7, pp. 422–426, Jul. 1970.
[9] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Proc. 3rd Theory Cryptogr. Conf.*, 2006, pp. 265–284.
[10] C. Xu, J. Ren, Y. Zhang, Z. Qin, and K. Ren, "DPPro: Differentially private high-dimensional data release via random projection," *IEEE Trans. Inf. Forensics Security*, vol. 12, no. 12, pp. 3081–3093, Dec. 2017.
[11] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," in *Parallel Distributed Processing—Explorations in the Microstructure of Cognition*, vol. 1, D. E. Rumelhart, Ed. Menlo Park, CA, USA: AAAI, 1986, ch. 8, pp. 318–362.
[12] E. Stefanov, C. Papamanthou, and E. Shi, "Practical dynamic searchable encryption with small leakage," in *Proc. Netw. Distrib. System Secur. Symp.*, 2014, pp. 832–848.
[13] E. Zdravevski *et al.*, "Improving activity recognition accuracy in ambient-assisted living systems by automated feature engineering," *IEEE Access*, vol. 5, pp. 5262–5280, 2017.

[14] G. Fanti, V. Pihur, and Ú. Erlingsson, "Building a RAPPOR with the unknown: Privacy-preserving learning of associations and data dictionaries," *Privacy Enhancing Technol.*, vol. 16, no. 3, pp. 41–61, 2016.
[15] G. Shrivastava and V. Bhatnagar, "Secure association rule mining for distributed level hierarchy in Web," *Int. J. Comput. Sci. Eng.*, vol. 3, no. 6, pp. 2240–2244, 2011.
[16] K. Sharma and G. Shrivastava, "Public key infrastructure and trust of Web based knowledge discovery," *Int. J. Eng., Sci. Manage.*, vol. 4, no. 1, pp. 56–60, 2014.
[17] M. Yang, T. Zhu, B. Liu, Y. Xiang, and W. Zhou, "Machine learning differential privacy with multifunctional aggregation in a fog computing architecture," *IEEE Access*, vol. 6, pp. 17119–17129, 2017.
[18] P. Kumar, M. Quadri, K. Sharma, N. N. Gia, and P. Ranjan, "Persistent cellular telephony: Enhanced secure GSM architecture," *Recent Patents Eng.*, vol. 12, no. 1, pp. 23–29, 2018.
[19] P. Leandro, C. Liming, N. Chris, and B. Jose, "MHEALTH dataset," in *Ambient Assisted Living and Daily Activities*, 6th ed. New York, NY, USA: Springer, 2014.
[20] R. Chen, A. Reznichenko, P. Francis, and J. Gehrke, "Towards statistical queries over distributed private user data," in *Proc. 9th USENIX Conf. Netw. Syst. Design Implement.*, 2012, pp. 169–182.
[21] R. Kumari and K. Sharma, "Cross-layer based intrusion detection and prevention for network," in *Handbook of Research on Network Forensics and Analysis Techniques*. IGI Global, 2018, pp. 38–56.
[22] R. Li, A. X. Liu, S. Xiao, H. Xu, B. Bezawada, and A. L. Wang, "Privacy and integrity preserving top-k query processing for two-tiered sensor networks," *IEEE/ACM Trans. Netw.*, vol. 25, no. 4, pp. 2334–2346, Aug. 2017.
[23] S. Ruder. (2016). "An overview of gradient descent optimization algorithms." [Online]. Available: https://arxiv.org/abs/1609.04747
[24] S. L. Warner, "Randomized response: A survey technique for eliminating evasive answer bias," *J. Amer. Statist. Assoc.*, vol. 60, no. 309, pp. 63–69, 1965.
[25] T.-H. H. Chan, E. Shi, and D. Song, "Optimal lower bound for differentially private multi-party aggregation," in *Proc. 20th Annu. Eur. Symp. Conf. Algorithms*, 2012, pp. 277–288.
[26] T. Wang, J. Blocki, N. Li, and S. Jha, "Optimizing locally differentially private protocols," in *Proc. 26th USENIX Security Symp.*, 2017, pp. 1–17.
[27] T. T. Nguyên, X. Xiao, Y. Yang, S. C. Hui, H. Shin, and J. Shin. (2016). "Collecting and analyzing data from smart device users with local differential privacy." [Online]. Available: https://arxiv.org/abs/1606.05053
[28] U. Erlingsson, V. Pihur, and A. Korolova, "RAPPOR: Randomized aggregatable privacy-preserving ordinal response," in *Proc. ACM Conf. Comput. Commun. Secur.*, 2014, pp. 1054–1067.
[29] X. Ren *et al.*, "LoPub: High-dimensional crowdsourced data publication with local differential privacy," *IEEE Trans. Inf. Forensics Security*, vol. 13, no. 9, pp. 2151–2166, Sep. 2018.
[30] Y. Sei and A. Ohsuga, "Differential private data collection and analysis based on randomized multiple dummies for untrusted mobile crowdsensing," *IEEE Trans. Inf. Forensics Security*, vol. 12, no. 4, pp. 926–939, Apr. 2017.
[31] Y.-T. Tsou, C.-S. Lu, and S.-Y. Kuo, "SER: Secure and efficient retrieval for anonymous range query in wireless sensor networks," *Comput. Commun.*, vol. 108, pp. 1–16, Aug. 2017.
[32] Z. Qin, Y. Yang, T. Yu, I. Khalil, X. Xiao, and K. Ren, "Heavy hitter estimation over set-valued data with local differential privacy," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2016, pp. 192–203.

YAO-TUNG TSOU received the Ph.D. degree in electrical engineering from National Taiwan University, Taipei, Taiwan. He was a Research Assistant with the Institute of Information Science, Academia Sinica, Taipei, from 2009 to 2012. He joined the Research Center for Information Technology Innovation (CITI), Academia Sinica, as a Research Assistant in 2013 and a Post-Doctoral Research Fellow in 2015, where he is currently a Visiting Scholar. He is also an Assistant Professor with the Department of Communications Engineering, Feng Chia University, Taichung, Taiwan. His research interests include sensor network security, data privacy and security, PUF, embedded system, and STT-MRAM applications.

BO-CHENG LIN received the B.S. and M.S. degrees from the Department of Communications Engineering, Feng Chia University, in 2014 and 2018, respectively. His research interests include data privacy and machine learning methodologies.