

Received December 7, 2019, accepted December 23, 2019, date of publication December 27, 2019, date of current version January 6, 2020.

Digital Object Identifier 10.1109/ACCESS.2019.2962700

Privacy-Preserving Locally Weighted Linear Regression Over Encrypted Millions of Data

XIAOXIA DONG^{1,3}, JIE CHEN^{2,3}, KAI ZHANG⁴, AND HAIFENG QIAN^{2,3}

¹Department of Computer Science and Technology, East China Normal University, Shanghai 200062, China

²School of Software Engineering, East China Normal University, Shanghai 200062, China

³Shanghai Institute of Intelligent Science and Technology, Tongji University, Shanghai 201804, China

⁴School of Computer Science and Technology, Shanghai University of Electric Power, Shanghai 201306, China

Corresponding authors: Jie Chen (s080001@e.ntu.edu.sg) and Kai Zhang (kzhang@shiep.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61972156, Grant 61802248, Grant 61571191, Grant 61632012, Grant U1705264, and Grant 61961146004, in part by the Young Elite Scientists Sponsorship Program by China Association for Science and Technology under Grant 2017QNRC001, in part by the Chenguang Program through the Shanghai Education Development Foundation and Shanghai Municipal Education Commission under Grant 18CG62, and in part by the Fundamental Research Funds for the Central Universities.

ABSTRACT The emerging development of cloud computing makes a trend that the cloud becomes a outsourced agglomeration for storing big data that generally contains numerous information. To mine the rich value involved in big data, the machine learning methodology is widespread employed due to its ability to adapt to data changes. However, the data mining process may involve the privacy issues of the users, hence they are reluctant to share their information. This is the reason why the outsourced data need to be dealt with securely, where data encryption is considered to be the most straightforward method to keep the privacy of data, but machine learning on the data in ciphertext domain is more complicated than the plaintext, since the relationship structure between data is no longer maintained, in such a way that we focus on the machine learning over encrypted big data. In this work, we study locally weighted linear regression (LWLR), a widely used classic machine learning algorithm in real-world, such as predict and find the best-fit curve through numerous data points. To tackle the privacy concerns in utilizing the LWLR algorithm, we present a system for privacy-preserving locally weighted linear regression, where the system not only protects the privacy of users but also encrypts the best-fit curve. Therefore, we use Paillier homomorphic encryption as the building modular to encrypt data and then apply the stochastic gradient descent in encrypted domain. After given a security analysis, we study how to let Paillier encryption deal with real numbers and implement the system in Python language with a couple of experiments on real-world data sets to evaluate the effectiveness, and show that it outperforms the state-of-the-art and occurs negligible errors compared with performing locally weighted linear regression in the clear.


INDEX TERMS Locally weighted linear regression, privacy-preserving, paillier homomorphic encryption, stochastic gradient descent.

I. INTRODUCTION

The rapid development of cloud computing technology makes an appealing trend that data owners are more and more willing to outsource their numerous big data to cloud-based servers [1], [2], many applications collect large amounts of data from different data owners. In such a way that, users are released from complicated data managements. To mine the rich value contained in the outsourcing stored big data in

cloud [3], the machine learning methodology is widely used due to its typically improvement on efficiency and accuracy over time thanks to the ever-increasing amounts of processed data. Nevertheless, some intermediate calculating results of a machine learning training process may be leaked to the cloud, hence the privacy concerns are raised when to use machine learning into the field of big data processing.

For example, the machine learning algorithms are widely used in E-health application. The patients, as data owners, prefer to upload personal medical record data to the E-health cloud and the health-care service provider (e.g., hospital) may

The associate editor coordinating the review of this manuscript and approving it for publication was Junggab Son .

use some machine learning methods aiming to make some predictions for medical diagnosis systems. Although this brings usage advances for the health-care service provider, the privacy concern immediately arises, since the patients' personal information usually contains a number of personal privacy information, such as age, gender, family address, social security number and so on. There are also other examples of machine learning that reveal user privacy, such as common advertising attacks, user phone number leaks, and so on [4], [5]. This motivates the occurrence of privacy-preserving machine learning.

In order to prevent the emergence of the privacy problem, it is natural for users to encrypt their data before sending them to the evaluator to preserve confidentiality [6]. However, encryption makes the evaluator side data mining and machine learning be very thorny. Moreover, the machine learning progress on the evaluator side may meet privacy threats as the involved private data belong to different users, and they don't trust each other. Even so, users are reluctant to give up the benefits and convenience of outsourcing data. So data are both outsourced and encrypted in the cloud, machine learning algorithms become challenging [2].

So it is important to design and develop the privacy-preserving machine learning protocol. To preserve the data privacy in the machine learning process, some cryptographic technologies are employed. One approach is utilizing secure multi-party computation (SMC) [7] technology, where data is divided among multiple servers using secret sharing. However, the SMC protocol requires users to stay online to participate in the subsequent computations, which brings about heavy computation and communication costs to the user side. Another usually used method is based on Yao's garbled circuit [8] along with homomorphic encryption [9], which can finally get the learning model by encrypting, designing gates and calculating obfuscation values without exposing any user's data. Nikolaenko et al. [10] still adopts Yao's garbled circuit to solve the computational problems in the learning model, but the computational efficiency is extremely low since complex circuits should be designed well that needs expensive communication costs and a large amount of time consumption.

However, using garbled circuits [10] to implement the calculation and processing between encrypted data is very inefficient, and designing garbled circuits also requires many circuit gates. In contrast, we only use stochastic gradient descent to solve the encryption learning model, and the communication overhead is greatly reduced. A table of theoretical efficiency comparisons is in the first question. And compared with the fully homomorphic encryption scheme [9], the Paillier encryption scheme is additive homomorphism, in comparison, the execution efficiency is much faster. From the efficiency comparison table of the first question, it is obvious that our homomorphic scheme is more effective than the garbled circuit scheme, and our scheme is more secure than the differential privacy scheme.

Generally, the system model we mentioned above is an outsourced model as shown in Figure 1. Users send their encrypted private data to a semi-honest evaluator, and don't want to reveal any plaintext of their private data to the evaluator or CSP [3]. To this end, we need a secure machine learning algorithm to enable the evaluator to perform privacy-preserving operations over these encrypted data, which is owned by different users [11]. We try to design machine learning algorithms under this outsourced model. Because locally weighted linear regression is widely used algorithm. We implement LWLR for the outsourced model.

A. OUR CONTRIBUTIONS

In this paper, we propose a new efficient privacy-preserving locally weighted linear regression scheme, in which we run the stochastic gradient descent method over encrypted data by Paillier encryption in order to get learning model without leaking data privacy. Specifically, the contributions of this work can be summarized as follows:

- 1) **Privacy Preserving.** All data uploaded by users, even the intermediate results calculated using these data, are encrypted, and no user privacy is leaked to the cloud. Even the learning models calculated at the end are encrypted, and users need to decrypt them before they can be used. Fundamentally address user data privacy concerns.
- 2) **Functionality Maintenance.** We provide a security analysis and conduct a couple of experiments to illustrate practicality, and hence make evaluations on the system under real UCI datasets. The experimental results show that we not only provide the privacy guarantee for the data, but also realize the optimal curve by using the encrypted data to perform stochastic gradient descent, which still only incurs negligible error rates compared with running locally weighted linear regression in the clear.
- 3) **Security Ensurance.**
 - i) User-Evaluator Security: the evaluator is unable to learn any plaintext of any user's data in our protocol.
 - ii) Evaluator-CSP Security: CSP is unable to learn any plaintext of any user's data in our protocol.
 - iii) User-User Security: No user is able to learn any privacy information of other user's data.
- 4) **Practical Running Efficiency.** A theoretical comparison is taken between our protocol and Nikolaenko et al.'s [6], which consists of three parts: CSP computation, evaluator computation and communication cost. The results show that our solution reduces the computation time from both the CSP side and the evaluator side, meanwhile, no need to design complex garbled circuit, hence it quite reduces the time required for computing gates of the circuit. Therefore, the introduced protocol enjoys better efficiency in the real environment via a theoretical analysis.

II. PROBLEM STATEMENT

A. SYSTEM MODEL

The studied system consists of three entities as shown in Figure. 1: data users, an evaluator and a cloud service provider (CSP). Data users provide data information (\vec{x}_i, y_i) , where \vec{x}_i is a d -dimensional vector that come from real numbers (i.e., $\vec{x}_i \in \mathbb{R}^d$), which contain the privacy information of user, such as age, family address, social security number and so on; while y_i is the information to be predicted by the learning model (e.g. the probability of suffering from A disease), which is also composed of real numbers (i.e., $y_i \in \mathbb{R}$). CSP is a crypto service provider that generates public/private keys and sends the public key to users, and initialize the system by setting up machine learning parameters. Evaluator is a service center who acts as a cloud-based data service role and runs calculate machine learning algorithm with provided data by users.

B. DESIGN GOALS

Our designed system aim to achieve the following four goals:

- 1) The system allows users to get rid of complex computing processes and to always stay offline during regression process.
- 2) The system allows users to efficiently store intermediate calculation results, users no longer need to consider how to get a learning model when to outsource data.
- 3) The system is able to ensure the privacy of users without revealing any users' private information or intermediate results.
- 4) The system considers the private key leakage problem as the public and private keys are generated by reputable CSP.

C. THREAT MODEL

The designed system aims to compute $[\theta]$ in a private-preserving manner, that is keeping any user's information and intermediate results private to evaluator and CSP.

In the system, both the evaluator and the CSP are assumed to be honest-but-curious and does not collude each other, that is honestly running the defined protocols to run the locally weighted linear regression algorithm but may infer information of each user. We remark that in the actual deployment of our system, there is only one evaluator or CSP is actually honest-but-curious as same as the existing work [12]. In this way, the user's privacy information cannot be inferred which is the inherent requirement of the system security. Moreover, the evaluator is not necessarily assumed to be a malicious party, since its actual goal is to get the computed value θ and privacy information of user rather than corrupting the computation to produce an incorrect result.

As for communication channel, we assume that the communication channel is open. Users only communicate with evaluator for sending encrypted data, users are able to capture or analyse the data transmitted in the channels, but data are encrypted. Users are also honest but curious, but they

just provide their encrypted privacy information to evaluator, there is no collaboration between users. However, they also attempt to get other users' privacy information.

III. BACKGROUND KNOWLEDGE

A. LOCALLY WEIGHTED LINEAR REGRESSION

Locally weighted linear regression algorithm is a classic algorithm that used in the field of statistical and machine learning [13], whose expected goal is to find the learning model θ . Specifically, its loss function $F(\theta)$ is based on linear regression and with a weight as:

$$F(\theta) = \sum_{i=1}^n w_i (y_i - \theta^T x_i)^2, \quad (1)$$

where W is the weight and let the weight be a decreasing function about the distance between the predicted points X . Hence, the Gaussian kernel function:

$$W(i) = \exp\left(-\frac{(x_i - x)^2}{2k^2}\right), \quad (2)$$

is employed to achieve locality for w , where x is the query point, x_i is the i -th training data point and k is a parameter that to be adjusted. It controls the attenuation rate of the distance between the training point and the query point.

Since the exponential functions cannot be operated in homomorphic cryptosystem, this work uses Taylor's formula to expand the exponential functions into polynomial inspired by [14], [15].

B. STOCHASTIC GRADIENT DESCENT (SGD)

For the loss function of locally weighted linear regression, the value of the function is expected to be minimized. Thus, stochastic gradient descent (SGD), a widely used approach to train learning model in machine learning aggregation, is considered to be an effective approximation algorithm to solve the local minimum of a function.

In SGD algorithm, θ is an initialized vector which can be all random numbers or all 0 value, where it is updated in each iteration as:

$$\theta_j := \theta_j - \alpha \frac{\partial L(\theta)}{\partial \theta},$$

where $L(\cdot)$ is the loss function of a machine learning algorithm in each iteration, where the samples (x_i, y_i) are used to update the randomly selected coefficient θ_j , and α is the learning rate that determines the step length to move down in each iteration, it is easy to miss the lowest point for α when it increases quickly, while the moving speed may be slow when α is quite small, hence it takes multiple iterations to reach the lowest point.

Based on the definition of SGD, the stochastic gradient descent formula of locally weighted linear regression behaves:

$$\theta_j := \theta_j - \alpha \sum_{i=1}^n (y_i - (x_i \theta_j)) w_i x_i, \quad (3)$$

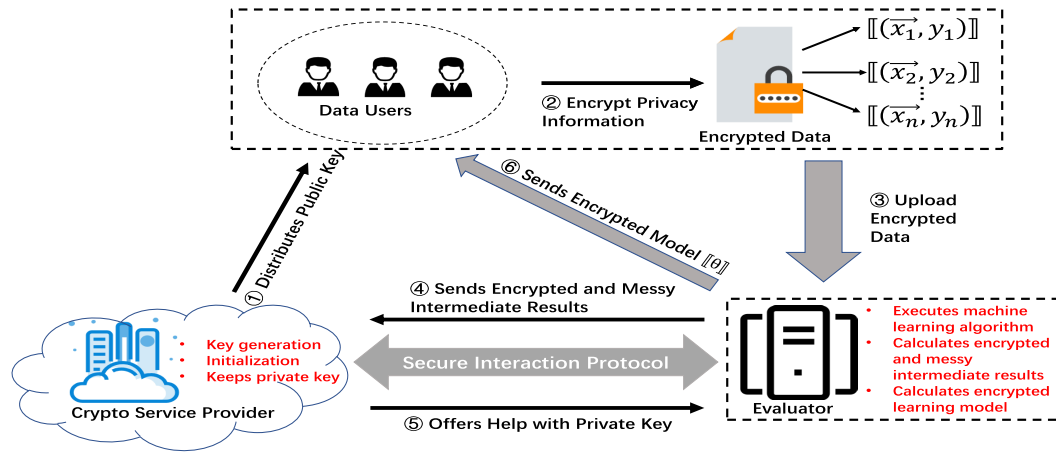


FIGURE 1. System model.

where computing the predicted output $y_i^* = x_i \cdot \theta_i$ is considered as forward propagation, while computing the change $\alpha \cdot w_i \cdot (y_i^* - y_i)x_i$ is regarded as backward propagation.

C. PAILLIER HOMOMORPHIC ENCRYPTION

Paillier homomorphic encryption system is a probabilistic public key encryption notion. The Paillier encryption consists the following three algorithms:

— **Key Generation.** Randomly select two independent large prime numbers p and q and a random integer $g \leftarrow \mathbb{Z}_{N^2}^*$. Compute $N = pq$ and sets (N, g) as the public key pk ; compute $\lambda = lcm(p-1)(q-1)$ and $\mu = (L(g^\lambda \bmod N^2)) - 1$ where $L(x) = \frac{x-1}{N}$ and sets (λ, μ) as the secret key sk .

— **Encryption.** The encryption algorithm E input a message $m \in \mathbb{Z}_N$ and picks a random number $r \in \mathbb{Z}_{N^2}^*$, generates the ciphertext $c = E(m, r) \rightarrow g^m \cdot r^N \bmod N^2$.

— **Decryption.** The decryption algorithm D decrypts the ciphertext c as $m \leftarrow D(c) = L(c^\lambda \bmod N^2) \cdot \mu \bmod N$.

We note that the Paillier encryption cryptosystem supports the following homomorphic operations:

1) *Homomorphic Addition.*

$$D(E(m_1) \cdot E(m_2) \bmod N^2) = m_1 + m_2.$$

2) *Homomorphic Multiplication.*

$$D(E(m_1)^{m_2} \bmod N^2) = m_1 \cdot m_2.$$

IV. SUB-ROUTINES FOR OUR PROTOCOLS

Before presenting our protocol in Section V, in this section, we introduce two generic protocols that are be used as sub-routines for our privacy-preserving locally weighted linear regression scheme. Since the Paillier homomorphic encryption can only operate on positive integers, while the processed data involved in processing the regression algorithm are real numbers, hence we have to extend the original Paillier encryption to deal with negatives and decimals. There are the issues that must be addressed in the implementation process.

A. ENCRYPTED MULTIPLICATION PROTOCOL

An objective between the evaluator who has a private input $(E(m_1), E(m_2))$ and the CSP who hold the secret key is to efficiently compute $E(m_1 \cdot m_2)$ while reveal nothing about m_1 and m_2 to evaluator or CSP. The Basic Secure Multiplication Protocol (BSMP) [16] makes this possible, whose basic idea follows the following observation:

$$m_1 \cdot m_2 = (m_1 + r_1) \cdot (m_2 + r_2) - m_1 \cdot r_2 - m_2 \cdot r_1 - r_1 \cdot r_2.$$

The step-by-step details for Encrypted Multiplication Protocol are shown in Algorithm 1. Firstly, the evaluator generates two random numbers r_1, r_2 and calculates $c_1 = E(m_1 + r_1) = E(m_1) \cdot E(r_1)$, and the same as m_2 to get c_2 . Then CSP decrypts c_1 and c_2 to calculate $h = (m_1 + r_1) \cdot (m_2 + r_2)$ and sends $E(h)$ to the evaluator. As the fixed-point form is used to process all the regression data that may cause some fractional parts to be discarded and thus cause some errors, which we will formally analyze the accuracy of fixed number in the experiment section.

B. EUCLIDEAN DISTANCE

Under the two encrypted vectors $(E(X'), E(X))$ held by the evaluator, the computed Euclidean Distance $E(|X' - X|^2)$ between X' and X , where X' and X are both d -dimensional vectors, reveals nothing about X and X' to the evaluator or the CSP. The basic idea of Euclidean Distance follows the following observation:

$$|X' - X|^2 = \sum_{i=1}^d (x'_i - x_i)^2.$$

The overall routines in Euclidean Distance are shown in Algorithm 2. Firstly, the Evaluator calculates $E(X'_i - X_i)$ for $1 \leq i \leq d$. Then the evaluator and the CSP use Encrypted Multiplication protocol to calculate $E(X' - X)$ in together.

C. TAYLOR POLYNOMIAL

The Taylor polynomial [17] is a formula that uses a function to describe the value of a point at a certain point. Note that

Algorithm 1 EM($E(m_1), E(m_2)$) $\rightarrow E(m_1 \cdot m_2)$

Require: The evaluator has a private input $E(m_1)$ and $E(m_2)$ and the CSP has sk .

Ensure: The evaluator obtains the value of $E(m_1 \cdot m_2)$.

— (i) **Evaluator:**

- 1: Generates two random numbers $r_1, r_2 \in \mathbb{Z}_N$
- 2: Computes $c_1 = E(m_1 + r_1) = E(m_1) \cdot E(r_1)$
- 3: Computes $c_2 = E(m_2 + r_2) = E(m_2) \cdot E(r_2)$
- 4: Sends c_1 and c_2 to the CSP

— (ii) **CSP:**

- 5: Receives c_1 and c_2 from evaluator
- 6: Computes $h_1 = (m_1 + r_1) = D(c_1)$
- 7: Computes $h_2 = (m_2 + r_2) = D(c_2)$
- 8: Computes $h = (m_1 + r_1) \cdot (m_2 + r_2) \bmod N = h_1 \cdot h_2 \bmod N$
- 9: Encrypts h to $h' \leftarrow E(h)$ and sends h' to the evaluator

— (iii) **Evaluator:**

- 10: Receives h' from the CSP
- 11: Computes $s \leftarrow h' \cdot E(m_1)^{N-r_2}$
- 12: Computes $s' \leftarrow s \cdot E(m_2)^{N-r_1}$
- 13: Computes $E(m_1 \cdot m_2) \leftarrow s' \cdot E(r_1 \cdot r_2)^{N-1}$

Algorithm 2 ED($E(X'), E(X)$) $\rightarrow E(|X' - X|^2)$

Require: Evaluator has $E(X')$ and $E(X)$, CSP has sk .

Ensure: Evaluator should get $E(|X' - X|^2)$ with the help of CSP.

— (i) **Evaluator:**

- 1: **for** $1 \leq i \leq d$ **do**
- 2: $E(x'_i - x_i) = E(x'_i) \cdot E(x_i)^{N-1}$
- 3: **end for**

— (ii) **Evaluator and CSP:**

- 4: **for** $1 \leq i \leq d$ **do**
- 5: Computes $E((x'_i - x_i)^2)$ using EM protocol
- 6: **end for**

— (iii) **Evaluator:**

- 7: Computes $E(|X' - X|^2) = \prod_{i=1}^d E((x'_i - x_i)^2)$

if the function is sufficiently smooth, the Taylor polynomial can use these derivative values as coefficients to construct a polynomial to approximate the value of the function in the neighborhood of this point. Since the homomorphic encryption system cannot perform the encryption operation on the exponential function, the weighted formula Eq.(2) is expanded by the Taylor polynomial to change into a polynomial operation, so that the exponential function can be operated by the homomorphic encryption system. It is mainly developed according to Eq.(4):

$$e^x = 1 + \frac{1}{1!}x + \frac{1}{2!}x^2 + \frac{1}{3!}x^3 + \frac{1}{4!}x^4 + \frac{1}{5!}x^5 + o(x^5). \quad (4)$$

The relationship between the number of polynomials and the accuracy is studied in Section VI. The overall routine in Taylor polynomial are shown in Algorithm 4, where m is the number of Taylor polynomial in terms of the exponential

function. As we use Taylor formula to expand the exponential function into polynomial operation that may cause an error, which we will analyze the accuracy of Taylor formula in the experiment section.

Algorithm 3 Taylor polynomial($E(x) \rightarrow E(e^x)$)

Require: Evaluator has $E(x)$, CSP has sk .

Ensure: Evaluator should get approximate value of $E(e^x)$ with the help of CSP.

— (i) **Evaluator:**

- 1: **for** $1 \leq i \leq m$ **do**
- 2: Generates m Taylor coefficients: $r_i = \frac{1}{i!}$
- 3: Encrypts all Taylor coefficients: $E(r_i)$
- 4: **end for**

— (ii) **Evaluator and CSP:**

- 5: Computes $u_1 = E(r_1) \cdot E(x)$ using EM protocol
- 6: Computes $a_2 = E(x) \cdot E(x)$ using EM protocol
- 7: **for** $2 \leq i \leq m$ **do**
- 8: $u_i = E(r_i) \cdot a_i$ using EM protocol
- 9: $a_{i+1} = E(x) \cdot a_i$ using EM protocol
- 10: **end for**
- 11: Computes $E(e^x) = \prod_{i=1}^m u_i$

V. OUR PROPOSED PROTOCOLS

This section presents a privacy-preserving locally weighted linear regression system, where we use Paillier encryption scheme to tackle the privacy concern since it is an additively homomorphic encryption with more highly efficiency than somewhat homomorphic encryption [18]. In the system, users sends the encrypted data to the evaluator, the evaluator calculates learning model in together with CSP. Assume the number of users in the system is n , the $E(\cdot)$ is the encryption fuction of Paillier encryption, xtr_i and ytr_i are the attributes of training set, xte and yte are the attributes of testing set. The λ is a coefficient of calculating weights, which corresponds to the Eq. (2) and θ is the learning model that we need to get.

Firstly, data owners generate and send the encrypted data to evaluator, the evaluator calculates the corresponding weight w according to Eq. (2) using all the data of the training set for each piece of user data in the test set. Secondly, based on the weight calculated, the loss model is minimized by the stochastic gradient descent according to Eq. (1), in this way, the evaluator and CSP can collaboratively run locally weighted linear regression algorithm through a secure interaction protocol. Finally, under getting the learning model θ , the evaluator sends back the encrypted learning model to data owners.

A. OBTAINING WEIGHTS

The main details in the stage of obtaining weights are given in Algorithm 4. Generally speaking, users initially encrypt their private information xtr_i , ytr_i , xte and yte then send them to the evaluator, where xtr_i is the x attribute of the training set,

ytr_i is the y attribute of the training set, xte is the x attribute of the testing set and yte is the y attribute of the testing set. Upon receiving $E(xtr_i)$, $E(ytr_i)$, $E(xte)$ and $E(yte)$ from users, the evaluator takes private input as $(E(xtr_i), E(xte))$ along with the secret key sk of CSP to jointly run the Euclidean Distance protocol for $1 \leq i \leq n$, which is used to calculate d_i that is the squared Euclidean distance between $E(xtr_i)$ and $E(xte)$. Then evaluator and CSP involve in the Encrypted Multiplication protocol together to obtain u_i , which corresponds to the Eq. (2). And $\lambda = -1/2k^2$ is a coefficient where k is a parameter that needs to be adjusted, after obtain u_i , they finally jointly involve in the Taylor polynomial protocol to get $E(w_i)$.

Throughout the process of obtaining weight, our system does not disclose any private information of users to the evaluator and CSP. Moreover, the intermediate results of calculating weights are also not leaked since they are still dealt with in encryption.

Algorithm 4 $OW(E(Xtr), E(Ytr), E(xte), E(yte) \rightarrow E(W))$

Require: Evaluator has $E(Xtr)$, $E(Ytr)$, $E(xte)$ and $E(yte)$, CSP has sk .

Ensure: Evaluator should get $E(W)$ with the help of CSP.

— (i) Users:

- 1: **for** $1 \leq i \leq n$ **do**
- 2: Encrypts xtr_i and ytr_i to get $E(xtr_i)$ and $E(ytr_i)$
- 3: Sends $E(xtr_i)$ and $E(ytr_i)$ to evaluator
- 4: **end for**
- 5: Sends $E(xte)$ and $E(yte)$ to evaluator

— (ii) Evaluator:

- 6: Computes $\lambda = -\frac{1}{2k^2}$
- 7: Encrypts λ to get $E(\lambda)$

— (iii) Evaluator and CSP:

- 8: **for** $1 \leq i \leq n$ **do**
 - 9: Computes $d_i = \text{ED protocol}(E(xtr_i), E(xte))$
 - 10: Computes $u_i = \text{EM protocol}(E(\lambda), d_i)$
 - 11: $E(w_i) = \text{Taylor polynomial}(u_i)$
 - 12: **end for**
-

B. CALCULATING LEARNING MODEL

The described obtaining weights protocol calculates the weights between all trained data and test data, but we still need to calculate the learning model θ with weights. The main details involved in the calculating learning model protocol is introduced in Algorithm 5.

In general, CSP firstly initializes θ_0 as 0s and then encrypts θ_0 and α , where α is the learning rate that determines the step size to move down in each iteration. And the CSP sends the encrypted $E(\theta_0)$ and $E(\alpha)$ to the evaluator. Secondly, the evaluator and CSP jointly calculate the product of $E(\theta_0(i))$ and $E(xtr_i)$ as h_i , for $1 \leq i \leq n$. Then the evaluator computes the difference between the true value of the i -th data and the predicted value, as $k_i = ytr_i - xtr_i \cdot \theta_0(i)$. Since it is a locally weighted algorithm where the difference k_i is also multiplied by the weight w_i as u_i , and s_i is the

backward propagation of Eq. (3). We add all the s_i together by a homomorphic multiplication, and the sum of all the s_i is m_j as $\frac{\partial L(\theta)}{\partial \theta}$. For $0 \leq j \leq \text{Num}$ where Num is the number of stochastic gradient descent iterations, f_j means the product of m_j and α , and it is also used in the learning rate to multiply backward propagation. Finally, we compute $E(\theta_{j+1}) = E(\theta_j) \cdot E(\alpha \cdot \frac{\partial L(\theta)}{\partial \theta})^{N-1}$ to implement Eq. (3).

Algorithm 5 $CLM(E(Xtr), E(Ytr), E(W) \rightarrow E(\theta))$

Require: Evaluator has $E(Xtr)$, $E(Ytr)$ and $E(W)$, CSP has sk .

Ensure: Evaluator should get $E(\theta)$ with the help of CSP.

— (i) CSP:

- 1: Sets θ_0 to 0s
- 2: Encrypts θ_0 and α , sends $E(\theta_0)$ and $E(\alpha)$ to evaluator

— (ii) Evaluator and CSP:

- 3: **for** $0 \leq j \leq \text{Num}$ **do**
 - 4: **for** $1 \leq i \leq n$ **do**
 - 5: $h_i = \text{EM Protocol}(E(\theta_0(i)), E(xtr_i))$
 - 6: $k_i = E(ytr_i) \cdot h_i^{N-1} \text{ mod } N$
 - 7: $u_i = \text{EM Protocol}(k_i, E(w_i))$
 - 8: $s_i = \text{EM Protocol}(u_i, E(xtr_i))$
 - 9: **end for**
 - 10: $m_j = \prod_{i=1}^n s_i$
 - 11: $f_j = \text{Encrypted Multiplication Protocol}(m_j, E(\alpha))$
 - 12: $E(\theta_{j+1}) = E(\theta_j) \cdot f_j^{N-1}$
 - 13: **end for**
-

C. SECURITY ANALYSIS

In this section, we give a security analysis on the proposed protocol. As can be concluded from the process of calculating the learning model, no sensitive information of any user are revealed to the evaluator or CSP, as well as the intermediate process part. As no collusion are assumed between the evaluator and CSP, the security analysis are taken from both the evaluator side and the CSP side. We assume that adopted homomorphic encrypted schemes are secure.

1) USER-EVALUATOR SECURITY

Lemma 1: The evaluator is unable to learn any plaintext of any user's data in our protocol.

Proof 1: To prove this lemma, we first show that user-evaluator security holds for the third phase of our model. In uploading encrypted data phase, the evaluator's responsibility is data storage. Users use the semantically secure Paillier cryptosystem to encrypt the data and then send to the evaluator, therefore the evaluator is unable to obtain any sensitive information of the user from the ciphertext.

In privacy-preserving LWLR operation, the evaluator plays a role in receiving, transferring, and computing intermediate variables in our protocol. Obviously, the ciphertext is applied into the locally weighted linear regression algorithm for a limited number of operations, moreover, only uses the secure Encrypted Multiplication Protocol or Euclidean Distance

protocol along with CSP to calculate the encrypted learning model θ . Note that, any intermediate results are encrypted throughout process, hence no any private information are leaked, as well as no information are leaked from calculating the learning model θ by the evaluator since they are still encrypted.

2) EVALUATOR-CSP SECURITY

Lemma 2: CSP is unable to learn any plaintext of any user's data in our protocol.

Proof 2: To prove this lemma, we first show that evaluator-CSP security holds for the secure interaction protocol in our model. Although it generates public and private keys but it cannot directly access the encrypted sensitive information submitted by the users. Furthermore it only helps the evaluator to run the secure Encrypted Multiplication protocol and Euclidean Distance protocol, at the same time, the received data from the evaluator are also encrypted data by involving some random numbers to blind intermediate results, hence even CSP successfully decrypts the data, the intermediate result information are still not obtained by it.

3) USER-USER SECURITY

Lemma 3: No user is able to learn any privacy information of other user's data.

Proof 3: Since users only communicate with evaluator, they won't directly communicate with each other. So user's privacy information may only be leaked through the communication channel during the third phase in our protocol. Because the channel is open, and users can capture and analyze data through it. However, users only send encrypted data through communication channel to evaluator. So other users are unable to learn any plaintext of transmission data as encryption scheme is secure. Even if the user captures the data, he can't determine which user's data is.

Putting the above three lemmas together, we have the following theorem.

Theorem 1: Privacy-preserving LWLR satisfies three security properties: user-evaluator security, evaluator-CSP security and user-user security.

VI. EXPERIMENTS

Experiment setup and Libraries. In this section, we conduct a couple of experiments on real datasets from the UCI repository [19] to illustrate the practical efficiency, where the protocol is compiled with Python 3.7 and implemented on a PC with an Intel(R) Core(TM) i7-4510U CPU @2GHz processor and 8GB RAM running Windows 10. And we use Paillier encryption to encrypt data with a 1024-bit modulus version, and use the math library and the random library that come from with python to write the code of Paillier cryptosystem. The pandas, sklearn and matplotlib libraries are used to write the codes for the locally weighted linear regression algorithm.

A. REPRESENTING REAL NUMBERS

As both the Paillier Cryptosystem and BSMP protocol can only operate over positive integers, nevertheless, the designed locally weighted linear regression algorithm need to deal with real numbers, which are typically rescaled the same domain (i.e., between -1 and 1). Hence, we should introduced a new modular to connect negative numbers and real numbers.

1) DECIMALS

Firstly, there are two approaches for representing real numbers: floating point and fixed point. In our system, we use fixed points to represent real numbers, since the element operations over floating point representation are difficult to implement in a data-agnostic way. And the fixed point representation is proceeded as follow: $[a] = \lfloor a \cdot 2^p \rfloor$, where p is a pre-defined system parameter and the number of bits p of the fractional part can be picked. When the parameters of different p are selected, the accuracy of the real number represented by the fixed point is also different where $\lfloor \cdot \rfloor$ is a round-down function. Here are listed some basic arithmetic operations for fixed-point numbers:

- 1) Addition/Subtraction: $[a \pm b] = [a] \pm [b]$;
- 2) Multiplication: $[a \cdot b] = [a] \cdot [b] / 2^p$.

For the multiplication of fixed-point numbers, the fixed-point representation of $[a \cdot b]$ is extended to 2^{2p} . However in the implementation, the expansion factor of the fixed-point representation should be independent of any operation, thus the expansion factor should be reduced to 2^p after each multiplication of the fixed-point representation. The overall steps in FNMT (Fixed Number Multiplication Truncation) are shown in Algorithm 6. That is, the evaluator selects a random number $r \in \mathbb{Z}_N$ with calculating $E([a] \cdot [b] + r)$ and sends it to the CSP. Under receiving it, the CSP decrypts it and calculates $\lfloor \frac{[a] \cdot [b] + r}{2^p} \rfloor$, also encrypts the result and sends it to the evaluator. Finally, the evaluator gets the result of $E([a \cdot b])$ by calculating $E(\lfloor \frac{[a] \cdot [b] + r}{2^p} \rfloor - \lfloor \frac{r}{2^p} \rfloor)$.

Algorithm 6 FNMT ($E([a] \cdot [b]) \rightarrow E([a \cdot b])$)

Require: Evaluator has $E([a] \cdot [b])$, CSP has sk .

Ensure: Evaluator should get $E([a \cdot b])$ with help of CSP.

— (i) **Evaluator:**

- 1: Picks random number $r \in \mathbb{Z}_N$
- 2: Calculates $U = E([a] \cdot [b] + r) = E([a] \cdot [b]) \cdot E(r)$, sends U to CSP

— (ii) **CSP:**

- 3: Receives U from evaluator
- 4: Decrypts U , $u = D(U)$
- 5: $h' = \lfloor \frac{[a] \cdot [b] + r}{2^p} \rfloor = \lfloor \frac{u}{2^p} \rfloor$
- 6: Encrypts h' , sends $E(h')$ to evaluator

— (iii) **Evaluator:**

- 7: Receives $E(h')$ from CSP
 - 8: Calculates $\lfloor \frac{r}{2^p} \rfloor$ and encrypts $s = E(\lfloor \frac{r}{2^p} \rfloor)$
 - 9: $E([a \cdot b]) = E(h' - \lfloor \frac{r}{2^p} \rfloor) = E(h') \cdot s^{N-1}$
-

2) NEGATIVES

For the negative number, in our system, the negative number is described as a fixed point, and then use the standard two's complement to represent the negative number. For a plaintext b , we denote it as a fixed-point form, i.e., $[b] = [b \cdot 2^p]$, while the fixed-point data type will be represented by σ bits, so the binary complement of plaintext b is represented as $\bar{b} = 2^\sigma + [b] \bmod 2^\sigma$. And subsequent operations are performed on \bar{b} . When to calculate the final result, the evaluator calculates the true representation of b by $\bmod 2^\sigma$. That is: $b(true) = \bar{b} \bmod 2^\sigma$.

B. FIXED NUMBER ACCURACY

As the decimal and negative numbers cannot be dealt with by Paillier encryption system, hence we use the fixed-point form to process all the regression data, which may cause some of the fractional part to be discarded and some errors. And more, we use θ^* to represent the learning model of locally weighted linear regression obtained in plaintext, and θ to represent the learning model obtained through our privacy-preserving scheme. Thus, we define the error rate of our protocol as:

$$Err_{\theta^*} = \left| \frac{F(\theta) - F(\theta^*)}{F(\theta^*)} \right|, \tag{5}$$

where the Eq. (5) can well reflect the loss of accuracy of our scheme, among them, we use matlab on a 64-bit commodity server in plaintext with locally weighted linear regression algorithm. Note that the experiments evaluate the average error rate by selecting different parameter values.

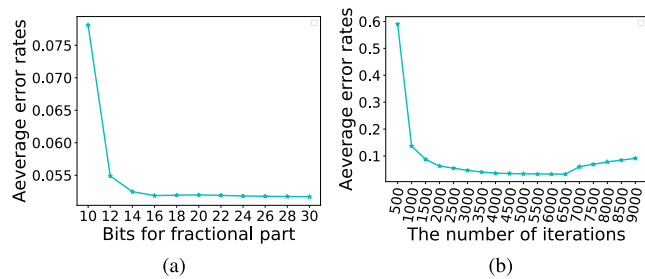


FIGURE 2. (a) Average error rate as a function of number of bits used for the fractional part of number representation. (b) Error rate as a function of the performed iterations in stochastic gradient descent.

Through a broad experiments, Figure 2a illustrates the relationship between bits for fractional part and average error rates. To verify this, we choose 10 to 30 for bits of fractional part, although the error is relatively large when to use 10 bits to represent the fractional part: as the number of fractional parts increases, the average error rate drops rapidly. When to use 14 bits to represent the fractional part, the average error rate reaches the smallest average error rate very quickly. And Figure 2b illustrates the relationship between the number of iterations of stochastic gradient descent and average error rates. To verify this, we choose 500 to 9000 for the number of iterations, although the average error rate is relatively large when the number of iterations is less than 1000, and when the number of iterations is greater than 1000, the average error

rate decreases as the number of iterations increases rapidly. When the number of iterations exceeds 6500, the average error rate increases since the number of iterations of the stochastic gradient descent is too large, resulting in over-fitting of the learning model and an increase in error rate.

C. TAYLOR FORMULA ACCURACY

Since locally weighted linear regression algorithm needs to calculate the encrypted exponential function e^x when calculating the weight, the Paillier encryption system cannot deal with the exponential function. As a result, we use Taylor formula to expand the exponential function into polynomial operation, which may cause an error incurred. Thus, we use e^x to denote the normal exponential function and $Taylor(x)$ to denote the Taylor expansion polynomial of e^x and define the error rate of our protocol as:

$$Err_{e^x} = \left| \frac{Taylor(x) - e^x}{e^x} \right|, \tag{6}$$

where the Eq. (6) calculates the error rate between the Taylor expansion and the real exponential function e^x .

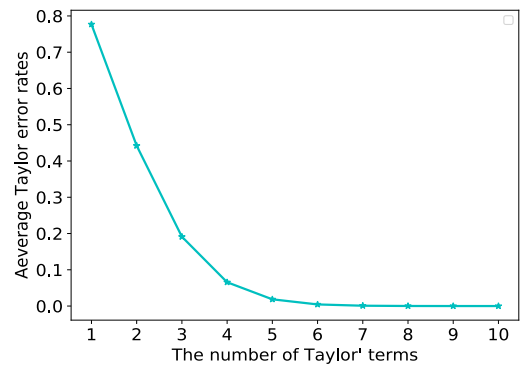


FIGURE 3. The average Taylor error rates as a function of the number of Taylor's terms.

Deriving Fig.3 by calculating Eq. (6) in matlab, it illustrates the relationship between the number of Taylor's terms and Taylor error rates, to verify this, we choose 1 to 10 for Taylor's number of terms. Although the error of Taylor expansion is larger when the number of Taylor expansion terms is smaller, but as the number of items expand, the Taylor error rate drops rapidly and steadily. As can be concluded: when expanding the exponential function to item 6, the Taylor error rate reaches the almost smallest average error rate very quickly; when expanding the exponential function to item 8, the average Taylor error rate decreases to 10^{-4} .

D. EFFICIENCY

The protocol is conducted on real datasets retrieved from UCI, where the UCI database is a database for machine learning proposed by the University of California Irvine who currently has 335 data sets and commonly used as a standard test databas. In TABLE 1, a theoretical comparison is taken between our protocol and Nikolaenko et al.'s, which consists of three parts: CSP computation, evaluator computation and

TABLE 1. The differences of theoretical analysis between our protocol and Nikolaenko et al.’s [10].

Comparison	Nikolaenko	Our protocol
CSP computation	$O(\zeta_{CSP} + O(d^2))$	$O(d^2)$
Evaluator computation	$O(\zeta_{CSP} + O(nd^2))$	$O(nd^2)$
Communication cost	$O(\zeta_{CSP} + O(d^2))$	$O(d^2)$

¹ ζ_{CSP} denotes that CSP solves linear problem by garbled circuit. $|\zeta_{CSP}|$ denotes the numbers of gates in the garbled circuit by CSP.

communication cost. CSP communication refers to the time efficiency of the CSP during the entire calculation process; Evaluator communication is the time efficiency of the evaluator throughout the calculation process; communication cost refers to the time efficiency consumed by the interaction between CSP and evaluator to complete machine learning algorithm. Among them, ζ_{CSP} denotes the designed garbled circuit by CSP and the $|\zeta_{CSP}|$ denotes the numbers of gates used in the garbled circuit by CSP. The results show that our solution reduces the computation time from both the CSP side and the evaluator side, meanwhile, no need to design complex garbled circuit, hence it quite reduces the time required for computing gates of the circuit. Therefore, the introduce protocol enjoys better efficiency in the real environment via a theoretical analysis.

Before the application or company outsources the data, they have to run the locally weighted linear regression algorithm with the user’s plaintext data, the efficiency is $O(nd^2)$, but they use the user’s plaintext data to train the learning model, which requires them to consume a lot of calculation Storage space for intermediate data. In order to improve efficiency and save space, the company will encrypt the user’s data and upload it to the cloud server. In this case, the company only needs to encrypt the user’s private information. The efficiency becomes $O(nd)$, which greatly reduces the company’s computing efficiency. The work of training the learning model only needs to be completed on the cloud server. At this time, the computing efficiency of the cloud server is $O(nd^2)$.

TABLE 2. Experimental results using UCI datasets.

Name	n	d	Bits	EnryT (min)	Time (h)
Automobile	205	26	24	12.02	19.71
Autompg	398	9	21	14.23	11.31
Challenger	23	5	13	0.55	0.22
Communities	1994	20	21	69.14	130.03
Computerhardware	209	9	19	11.26	11.44
Concreteslumpstest	103	10	19	5.41	4.93
ConcreteStrength	1030	9	19	50.12	51.90

TABLE 2 lists the results of our protocol running with actual data sets, in which there are 7 real-world datasets that are used for evaluating our protocol. Assume summarizing the number of entries to be n , the number of features to be d and the number of bits to represent the fixed numbers. The purpose of this table is aimed to provide the overall time evaluation on the evaluator and the CSP, which starts from the initial stage of the computation by CSP and final stage of the execution by the evaluator. At the same time, TABLE 2 also

provides the encryption costs of all users, where “EnryT” means the encryption time-consuming of all users and “Time” means the overall time between CSP and evaluator.

Since locally weighted linear regression employes all training sets to calculate weights for each set of test data, even if the algorithm used to run the machine learning in plaintext takes longer than the normal regression algorithm. Among them, the smallest data set “challenger” has only five features, thus the encryption time required for it achieves as short as 0.55 minutes, the overall time costs of the evaluator and CSP are almost 0.2 hours. The slowest overall execution time between CSP and evaluator is the “communities” data set that requires about 130 hours, among them, the encryption time take 69 minutes for 1994 entries and 20 features are involved. However, we can see the efficiency of the protocol is quite acceptable for most actual users.

VII. RELATED WORK

Earlier work on privacy-preserving machine learning mainly focused on k-means [20], [21], KNN, SVM classification [22], [23] and decision trees [24], [25]. As linear regression becomes more and more widespread in practical applications, the privacy-preserving problem of linear regression has received enthusiastic attention, but the former employed traditional methods were difficult to divide the databases horizontally or vertically.

On one hand, the used method in horizontally partitioning databases is secret sharing methodology [26], where users share their sensitive information into multiple servers. The multiple servers follow a distributed protocol to cooperatively run machine learning algorithms together, i.e., BGW [27]. On the other hand, the common method for vertically partitioning of databases is homomorphic encryption. Nikolaenko et al., realized the privacy-preserving linear regression by vertically dividing the databases and combining LHE and garbled circuits, and Gascon et al., [28] extends its conclusions to horizontally divided databases. As a result, both of them use a garbled circuit and take resources to design the gate of the circuit. Moreover, Hu et al., [29] gives a solution on the privacy-preserving linear regression problem via Gaussian elimination and Jacobi iteration and then transform the linear regression problem into solving linear equations.

There are also many privacy-preserving machine learning protocols that use hybrid approaches, such as combining secret sharing with garbled circuits to learn the decision tree model [25]. Meanwhile, the use of differential privacy [30], [31] to implement privacy-preserving machine learning is also widely studied [22], [32]. Nevertheless, the principle of differential privacy needs to add noises to the data or to update function (e.g., [14]), in such a way that, the server can fully access to the user’s plaintext data that is not expected due to privacy concerns.

Compared with the garbled circuit and the fully homomorphic protocol, our protocol is significantly more efficient and requires less communication overhead. The theoretical analysis can be seen in SECTION VI, SUBSECTION E

Efficiency. Compared with the differential privacy scheme, our scheme is more secure. For specific proof, see SECTION V, SUBSECTION C Security Analysis.

However, the previous privacy-preserving linear regression protocols only addresses the problems of linear system, on the contrary, our protocol uses the stochastic gradient descent method to concern about the privacy-preserving problem of machine learning. We note that this method can also be extended to non-linear systems, such as logistic regression [33] and neural networks.

VIII. CONCLUSION

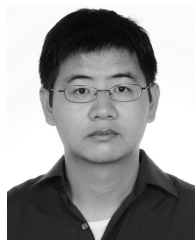
In this paper, we proposed a novel privacy-preserving locally weighted linear regression scheme without leaking anything else about the users' data. By using Taylor formula to extend exponential function based on Paillier encryption methodology, we employed stochastic gradient descent to compute locally weighted linear regression over encrypted data and gave a formal security analysis. Finally, we implemented the scheme through extensive experiments, where the experimental results indicate that our protocol outperforms the state-of-the-art protocol in both computation efficiency and communication cost. Most importantly, our protocol can be used as a modular to address the privacy concerns of data mining.

REFERENCES

- [1] F. Liu, W. K. Ng, and W. Zhang, "Encrypted SVM for outsourced data mining," in *Proc. IEEE 8th Int. Conf. Cloud Comput.*, Jun. 2015.
- [2] L. Fang, W. K. Ng, and W. Zhang, "Encrypted scalar product protocol for outsourced data mining," in *Proc. IEEE 7th Int. Conf. Cloud Comput.*, Jun. 2014.
- [3] F. Liu, W. K. Ng, W. Zhang, D. H. Giang, and S. Han, "Encrypted set intersection protocol for outsourced datasets," in *Proc. IEEE Int. Conf. Cloud Eng.*, Mar. 2014.
- [4] S. Hu, M. Li, Q. Wang, S. S. M. Chow, and M. Du, "Outsourced biometric identification with privacy," *IEEE Trans. Inf. Forensics Security*, vol. 13, no. 10, pp. 2448–2463, Oct. 2018.
- [5] S. Hu, C. Cai, Q. Wang, C. Wang, X. Luo, and K. Ren, "Searching an encrypted cloud meets blockchain: A decentralized, reliable and fair realization," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Apr. 2018.
- [6] F. Liu, W. K. Ng, and W. Zhang, "Encrypted association rule mining for outsourced data mining," in *Proc. IEEE 29th Int. Conf. Adv. Inf. Netw. Appl.*, Mar. 2015.
- [7] A. P. Sanil, A. F. Karr, X. Lin, and J. P. Reiter, "Privacy preserving regression modelling via distributed computation," in *Proc. 10th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2004.
- [8] A. C.-C. Yao, "How to generate and exchange secrets," in *Proc. 27th Annu. Symp. Found. Comput. Sci. (SFCS)*, Oct. 1986, pp. 162–167.
- [9] C. Gentry, "Fully homomorphic encryption using ideal lattices," in *Proc. 41st Annu. ACM Symp. Symp. Theory Comput. (STOC)*, vol. 9, 2009, pp. 169–178.
- [10] V. Nikolaenko, U. Weinsberg, S. Ioannidis, M. Joye, D. Boneh, and N. Taft, "Privacy-preserving ridge regression on hundreds of millions of records," in *Proc. IEEE Symp. Secur. Privacy*, May 2013, pp. 334–348.
- [11] F. Liu, W. K. Ng, and W. Zhang, "Encrypted gradient descent protocol for outsourced data mining," in *Proc. IEEE 29th Int. Conf. Adv. Inf. Netw. Appl.*, Mar. 2015.
- [12] S. Kim, J. Kim, D. Koo, Y. Kim, H. Yoon, and J. Shin, "Efficient privacy-preserving matrix factorization via fully homomorphic encryption," in *Proc. 11th ACM Asia Conf. Comput. Commun. Secur. (ASIA CCS)*, 2016, pp. 617–628.
- [13] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction* (Springer Series in Statistics). New York, NY, USA: Springer, 2009.
- [14] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur. (CCS)*, 2016, pp. 308–318.
- [15] Y. Lindell and B. Pinkas, "A proof of security of Yao's protocol for two-party computation," *J. Cryptol.*, vol. 22, no. 2, pp. 161–188, Apr. 2009.
- [16] Y. Elmehdwi, B. K. Samanthula, and W. Jiang, "Secure k-nearest neighbor query over encrypted data in outsourced environments," in *Proc. IEEE 30th Int. Conf. Data Eng.*, Mar. 2014, pp. 664–675.
- [17] T. M. Apostol, *Mathematical Analysis: A Modern Approach to Advanced Calculus*. Reading, MA, USA: Addison-Wesley, 1974.
- [18] J. Fan and F. Vercauteren, "Somewhat practical fully homomorphic encryption," IACR Cryptol. ePrint Arch., Tech. Rep. 2012/144, 2012, p. 144.
- [19] *UCI Machine Learning Repository*. Accessed: Sep. 1, 2018. [Online]. Available: <http://archive.ics.uci.edu/ml/datasets.html>
- [20] Y. Ishai, J. Kilian, K. Nissim, and E. Petrank, "Extending oblivious transfers efficiently," in *Advances in Cryptology—CRYPTO*. Berlin, Germany: Springer, 2003.
- [21] M. Bellare, V. T. Hoang, and P. Rogaway, "Foundations of garbled circuits," in *Proc. ACM Conf. Comput. Commun. Secur. (CCS)*, 2012.
- [22] S. Shuang, K. Chaudhuri, and A. D. Sarwate, "Stochastic gradient descent with differentially private updates," in *Proc. Global Conf. Signal Inf. Process.*, Dec. 2013, pp. 245–248.
- [23] A. C. Yao, "Protocols for secure computations," in *Proc. Symp. Found. Comput. Sci.*, 2008, pp. 160–164.
- [24] Y. Qi and M. J. Atallah, "Efficient privacy-preserving k-nearest neighbor search," in *Proc. 28th Int. Conf. Distrib. Comput. Syst.*, Jun. 2008, pp. 311–319.
- [25] R. Agrawal and R. Srikant, "Privacy-preserving data mining," *ACM SIGMOD Rec.*, vol. 29, no. 2, 2000, pp. 439–450.
- [26] A. F. Karr, W. J. Fulp, F. Vera, S. S. Young, X. Lin, and J. P. Reiter, "Secure, privacy-preserving analysis of distributed databases," *Technometrics*, vol. 49, no. 3, pp. 335–345, Aug. 2007.
- [27] M. Ben-Or, S. Goldwasser, and A. Wigderson, "Completeness theorems for non-cryptographic fault-tolerant distributed computation," in *Proc. 20th Annu. ACM Symp. Theory Comput.*, 1988, pp. 1–10.
- [28] A. Gascón, P. Schoppmann, B. Balle, M. Raykova, J. Doerner, S. Zahur, and D. Evans, "Secure linear regression on vertically partitioned datasets," IACR Cryptol. ePrint Arch., Tech. Rep., 2016, p. 892.
- [29] S. Hu, Q. Wang, J. Wang, S. S. Chow, and Q. Zou, "Securing fast learning! ridge regression over encrypted big data," in *Proc. IEEE TrustCom/BigDataSE/ISPA*, Aug. 2016, pp. 19–26.
- [30] C. Dwork, "Differential privacy," in *Encyclopedia of Cryptography and Security*. 2011, pp. 338–340.
- [31] C. Dwork and J. Lei, "Differential privacy and robust statistics," in *Proc. 41st Annu. ACM Symp. Symp. Theory Comput. (STOC)*, vol. 9, 2009, pp. 371–380.
- [32] K. Chaudhuri and C. Monteleoni, "Privacy-preserving logistic regression," in *Proc. Adv. Neural Inf. Process. Syst.*, 2009, pp. 289–296.
- [33] A. B. Slavkovic, Y. Nardi, and M. M. Tibbitts, "Secure logistic regression of horizontally and vertically partitioned distributed databases," in *Proc. 7th IEEE Int. Conf. Data Mining Workshops (ICDMW)*, 2007, pp. 723–728.



XIAOXIA DONG is currently pursuing the M.S. degree with the Department of Computer Science and Technology, East China Normal University, Shanghai, China. Her research interests include applied cryptography, information security, and machine learning.



JIE CHEN received the B.S. degree in mathematics from Soochow University, China, in 2008, and the Ph.D. degree in mathematics from Nanyang Technological University, Singapore, in 2013. He was a Researcher with ENS de Lyon, France, in 2016. He is currently a Professor with East China Normal University, China. His research interests include public-key cryptography and information security.



KAI ZHANG received the bachelor's degree with the School of Information Science and Engineering, Shandong Normal University, China, in 2012, and the Ph.D. degree with the Department of Computer Science and Technology, East China Normal University, China, in 2017. He is currently an Assistant Professor with the Shanghai University of Electric Power, China. His research interests include applied cryptography and information security.



HAIFENG QIAN received the B.S. and master's degrees in algebraic geometry from the Department of Mathematics, East China Normal University, Shanghai, China, in 2000 and 2003, respectively, and the Ph.D. degree from the Department of Computer Science and Engineering, Shanghai Jiao Tong University, in 2006. He is currently a Professor with the School of Software Engineering, East China Normal University. His main research interests include network security, cryptography, and algebraic geometry. He is also serving as a Reviewer of multiple international journals and academic conferences.

...