Secure and Privacy-Preserving Decision Tree Classification with Lower Complexity

Liang Xue, Dongxiao Liu, Cheng Huang, Xiaodong Lin, Xuemin (Sherman) Shen

***Abstract*—As a widely-used machine-learning classifier, a decision tree model can be trained and deployed at a service provider to provide classification services for clients, e.g., remote diagnostics. To address privacy concerns regarding the sensitive information in these services (i.e., the clients' inputs, model parameters, and classification results), we propose a privacy-preserving decision tree classification scheme (PDTC) in this paper. Specifically, we first tailor an additively homomorphic encryption primitive and a secret sharing technique to design a new secure two-party comparison protocol, where the numeric inputs of each party can be privately compared as a whole instead of doing that in a bit-by-bit manner. Then, based on the comparison protocol, we exploit the structure of the decision tree to construct PDTC, where the input of a client and the model parameters of a service provider are concealed from the counterparty and the classification result is only revealed to the client. A formal simulation-based security model and the security proof demonstrate that PDTC achieves desirable security properties. In addition, performance evaluation shows that PDTC achieves a lower communication and computation overhead compared with existing schemes.**

***Keywords*—decision trees, data privacy, model privacy, secure comparison, machine-learning classification**

I. Introduction

Machine learning as a service (MLaaS) has emerged as one kind of popular remote service that leverages the extraordinary computation capability of service providers such as Google and Amazon to provide prediction and classification services for clients. Among various machine-learning models in MLaaS, a decision tree classification model is a popular and powerful model known for its interpretability and effectiveness, which can be applied in many applications, such as genome sequencing[1], spam filters[2], and online medical diagnosis[3]. Under the MLaaS service paradigm, a client can submit his or her personal data (i.e. a feature vector) to a service provider who deploys a pre-trained decision tree model, and the service provider can evaluate the decision tree model using the client's data and return the classification result back to the client.

Although the decision-tree-based MLaaS has many advantages, it still suffers from privacy threats from the perspectives of the client and the service provider. The client's data[4] may contain sensitive personal information, such as financial records and medical data in the credit assessment and remote health diagnosis applications, and the final classification results may also reveal private information of clients, such as a diagnosed disease, which should also be concealed from the service provider. At the same time, the pre-trained classification models are regarded as valuable intellectual property of service providers, which should not be disclosed to clients. Moreover, many recent researches[5-7] even indicate that the training dataset can be extracted from the model if an adversary is allowed to access the model in a white-box manner, which results in violating the European general data protection regulation (GDPR). Therefore, it becomes an urgent requirement to develop a privacy-preserving decision tree classification scheme that protects data privacy for clients and model privacy for service providers.

Generic secure multi-party computation protocols[8-10] utilize garbled circuit[11] and homomorphic encryption[12,13] to achieve privacy-preserving decision tree classification. The basic idea is to transform the decision tree classification into secure integer comparison in the ciphertext domain. LIU et al.[14] proposed a programming framework called ObliVM. The proposed framework compiles the generic programs into oblivious representations that are suitable for secure computation. Unfortunately, the size of the generated oblivious program is proportional to the size of the decision tree, which

Manuscript received Jan. 07, 2020; revised Feb. 10, 2020; accepted Feb. 19, 2020. The associate editor coordinating the review of this paper and approving it for publication was X. Cheng.

L. Xue, D. X. Liu, C. Huang, X. M. (Sherman) Shen. Department of Electrical and Computer Engineering, University of Waterloo, Waterloo N2L 3G1, Canada (e-mail: l34xue@uwaterloo.ca; dongxiao.liu@uwaterloo.ca; cheng.huang@uwaterloo.ca; sshen@uwaterloo.ca).

X. D. Lin. School of Computer Science, University of Guelph, Guelph N1G 2W1, Canada (e-mail: xlin08@uoguelph.ca).

may affect classification efficiency especially when the decision tree is large. Some protocols[15,16] exploited the additively homomorphic encryption to achieve private decision tree classification, where the threshold at each decision node and the input data are expressed in a binary form. A client needs to encrypt the data in a bit-by-bit manner using a homomorphic encryption scheme, and sends the ciphertexts to the service provider. The service provider performs the comparison homomorphically on the ciphertexts and returns the results to the client. Since each bit of the data is encrypted, the approaches may incur expensive cost in terms of computation and communication overhead for both the client and the service provider. Although existing schemes have achieved rich functionalities of decision tree classification, it still remains a very challenging task to design a privacy-preserving decision tree classification scheme with less communication and computation overhead.

In this paper, we aim to improve the efficiency of the decision tree classification, while preserving both the client data privacy and the provider model privacy. The main contributions of this paper can be summarized as follows.

- We design a novel secure numeric comparison protocol from a homomorphic encryption primitive and a secret sharing technique. Specifically, instead of encrypting the numbers to be compared in bits, the protocol achieves secure numeric comparison in a more efficient way in terms of the computational and communication cost.
- Based on the secure numeric comparison protocol, we construct a privacy-preserving decision tree classification scheme (PDTC) with versatile privacy preservations. Client data privacy, including the inputs and the classification results, and the model privacy are preserved in the proposed scheme.
- Under the simulation-based security model with the semi-honest client and service provider, we formally prove the security of PDTC. Moreover, the simulation results on different datasets show the efficiency of PDTC.

The remainder of this paper is organized as follows. Section II shows the related works. In section III, we describe the system model, security model, and the design goals. In section IV, we review the preliminaries pertaining to our construction. In section V, we give our construction, followed by the security analysis in section VI and performance evaluation in section VII. We conclude the paper in section VIII.

II. Related Work

In this section, we briefly review the literature related to the privacy-preserving decision tree classification and secure multi-party computation.

BRIKELL et al.[17] proposed the pioneering scheme for privacy-preserving evaluation of branching programs with the combination of garbled circuits, blinding techniques and homomorphic encryption in a novel way. The server first translates a branching program into an equivalent form, where each classification node is encrypted and remains unknown to the client unless the appropriate key is obtained. In order for the client to obtain the correct key corresponding to the classification result, oblivious attribute selection and homomorphic encryption are employed. In the final stage, a client only learns the classification label. The size of the generated program is linear to the size of the decision tree, thereby incurring high communication overhead and making it less efficient for large-scale trees. BOST et al.[18] constructed secure classification protocols for a wide range of applications, including decision trees and hyperplane decision. They treated decision trees as a multivariate polynomial, which is a combination of terms, each of which represents a path from the root to a leaf. The terms can be represented as the multiplications of the value of a leaf node and boolean values at the decision nodes on this path. The calculation of the polynomial in their scheme requires fully homomorphic encryption (FHE).

WU et al.[15] proposed a decision tree classification scheme that only utilizes additively homomorphic encryption (AHE). The authors employed randomization techniques to conceal decision tree structures. The two parities run a secure comparison protocol[19] which reveals a decision string as the classification result to the client via an oblivious transfer protocol with the server. TAI et al.[16] cleverly exploited the structure of the decision trees and marked the left edge of a decision node with the cost b and right edge of the node with the cost $1-b$, where $b \in \{0,1\}$ is the comparison result at that decision node. The edge costs along each path are summed and randomized, and the result is called the path cost. The classification value related to each leaf node is added to the corresponding path cost and the results are returned to the client. TUENO et al.[20] represented a decision tree as an array. In their scheme, only d comparisons need to be executed, where d represents the decision tree height. They introduced the concept of oblivious array indexing, which allows the parties to obliviously select the index of the next node. KISS et al.[21] proposed a private decision tree classification scheme with modular designs. They systematically reviewed the state-of-the-art protocols and identified new combinations of these protocols that can provide better runtime and communication tradeoffs. TUENO et al.[22] proposed a novel decision tree classification scheme where a client can delegate the calculation to the server to reduce the overall communication rounds. They also instantiated the main protocol using the binary representation of the input and arithmetic circuit, respectively. ZHANG et al.[23] proposed a secure framework that enables the server to outsource the decision tree model to the cloud while preserving both the client and the server privacy. They designed a secure comparison scheme for the decision nodes

using a two-server model and additive secret sharing. In this paper, we integrate the homomorphic encryption and secret sharing in a new way so that the secure integer comparison is achieved and computational costs for both parties are reduced.

III. Problem Formulation

In this section, we first define the system model and security model. Then, we identify the design goals that need to be achieved in this paper.

A. System Model

As shown in Fig. 1, three entities are involved in our system: a service provider, clients, and a trusted authority (TA).

• A service provider. A service provider (e.g, a medical center) holds a trained decision tree model and provides classification services to clients. As the model is proprietary, the service provider is not willing to leak the model parameters to clients.

• Clients. A client owns a private input which is represented as a feature vector that contains information of different attributes, such as weight, heart rate, blood pressure, etc. The client would like to leverage the model generated by the service provider to obtain the classification result of her input. Since the input contains sensitive information, the client would not offer the feature vector to the service provider in the plain text. In addition, due to the limited storage and computation resources of the client, the communication and computation overhead at the client should be small.

• TA. A TA is introduced to issue the identity certificates to both parties, which can be used for mutual authentication between the client and the service provider.

At a high level, our system works as follows. First, clients and the service provider generate their public-private keys and register with a TA, who will issue certificates for them. Then, a client encrypts his or her input vector and sends the ciphertext to the service provider. After receiving the input of the model, the service provider executes the decision tree classification on the ciphertexts and returns the protected result to the client, who can recover the classification result using his or her private key.

B. Security Model

We adopt the simulation-based model[24] in this paper, which compares the real-world protocol execution with the ideal-world function evaluation with a trusted party. In this paradigm, a protocol π securely achieves a function f if the following holds. 1) There exists an ideal-world adversary $\mathscr{A}'$ (sometimes referred to as a simulator $\mathscr{S}$) for any real-world adversary A, which can perform the equivalent attack. 2) No feasible environment can distinguish whether it is interacting with the real-world $\mathscr{A}$ and π or the ideal-world f and $\mathscr{A}'$. The

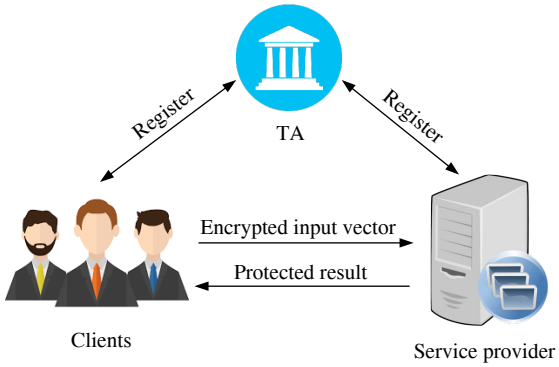


Figure 1 System model

environment, modeled as a polynomial-size circuit, takes λ as a security parameter and chooses inputs for entities. It acts as a distinguisher between the two executions.

In our protocol, we denote m as the number of internal nodes in the decision tree, and n as the feature vector dimension. m,n and the client public key are known to the public. We assume that the client and service provider in the protocol are semi-honest, which means they behave according to the protocol specification, but are curious about the counterparty's input[15,16]. In the following, we define the real-world and ideal-world executions with a formal security definition of the decision tree classification.

Real-world execution. Let π be a decision tree classification protocol. The execution of π and $\mathscr{A}$ is coordinated by the environment, denoted as $\mathscr{Z}$. First, $\mathscr{Z}$ generates a key pair and a feature vector $\boldsymbol{X} = (x_1, \cdots, x_n)$ for the client as the inputs. $\mathscr{Z}$ also chooses a decision tree model T for the service provider and $\mathscr{Z}$ reveals the input of either a corrupted client or provider to the adversary $\mathscr{A}$. During the protocol execution, the honest party behaves according to the protocol specification, and the corrupted party behaves as directed by $\mathscr{A}$. Since $\mathscr{A}$ is semi-honest, the corrupted client or provider will also follow the protocol. At the end of the protocol, $\mathscr{A}$ computes an output of an arbitrary function based on its view and sends the output to the environment $\mathscr{Z}$. The honest party also gives its output to $\mathscr{Z}$. After the protocol execution, $\mathscr{Z}$ outputs a bit b. We denote $\mathrm{REAL}_{\pi,\mathscr{A},\mathscr{Z}}(\lambda)$ as a random variable of the bit value.

Ideal-world execution. In the ideal world, there is a trusted party that evaluates the function f securely and provides outputs to the parties. At the beginning, $\mathscr{Z}$ first gives the service provider a model T and the client a vector $\boldsymbol{X}$. The honest party sends its input to the trusted party. The input of corrupted party is given to the adversary. Since the adversary $\mathscr{A}'$ is semi-honest, it instructs the corrupted party to submit the input it received to the trusted party. After receiving the inputs $\boldsymbol{X}$ from the client and T from the service provider, the trusted party calculates $T(\boldsymbol{X})$ and sends the result to the client. After the execution of the function, the honest party gives its output

to $\mathscr{Z}$. The adversary computes an output of an arbitrary function based on its view and sends it to $\mathscr{Z}$. Finally, $\mathscr{Z}$ outputs a bit b. We denote $\text{IDEAL}_{f,\mathscr{A}',\mathscr{Z}}(\lambda)$ as the random variable of the bit value.

Based on the executions in the real world and ideal world, we have the following definition of security.

Definition 1 (Security) Given a client with input $\boldsymbol{X}$ and a service provider with a model T, a protocol π is said to securely implement the decision tree classification functionality, if for any semi-honest probabilistic polynomial time (PPT) adversary $\mathscr{A}$ and every polynomial-size circuit family $\mathscr{Z}$, there exists a PPT adversary $\mathscr{A}'$ so that the following probability is negligible:

$$|\Pr[\text{REAL}_{\pi,\mathscr{A},\mathscr{Z}}(\lambda)=1]-\Pr[\text{IDEAL}_{f,\mathscr{A}',\mathscr{Z}}(\lambda)=1]|.$$

C. Design Goals

- Security: The proposed scheme should achieve data privacy and model parameter privacy based on Definition 1. To be specific, for the data privacy, the service provider can fulfill the decision tree classification functionality without knowing the client input and the classification result. On the other hand, for the model parameter privacy, the parameters of the model, as the intellectual property of the service provider, cannot be inferred by the client.
- Efficiency: Considering that the client may be resource-constrained and the delay requirement of the classification services, the communication overhead and computation overhead of the client and the service provider should be small.

IV. Preliminaries

In this section, we present the preliminaries that are used to construct our scheme, which include decision tree classification and homomorphic encryption.

A. Decision Tree Classification

Decision tree models are widely used in machine learning services. A decision tree is a binary tree, and each internal node or decision node represents a test on an attribute. The input of a decision tree is an n-dimensional feature vector $\boldsymbol{X}=\{x_1,\cdots,x_n\}$ and the decision tree has a threshold vector $\boldsymbol{Y}=\{y_1,\cdots,y_m\}$, where m denotes the number of decision nodes. Each decision node D_k corresponds to a Boolean function $f_k(\boldsymbol{X})=1\{x_{i_k}<y_k\}$, where $i_k\in[n]$ and $\{y_k\}_{k\in[m]}$ are thresholds. We say that $1\{x_{i_k}<y_k\}$ equals to 1 if and only if $x_{i_k}<y_k$. The outcome of each decision node determines the branch taken next. Each leaf node represents a classification result.

Decision tree classification means that given an input $\boldsymbol{X}$, an evaluation $f_k(\boldsymbol{X})$ at each decision node is performed and a classification result of the vector $\boldsymbol{X}$ is returned. As shown in Fig. 2, the input $\boldsymbol{X}$ is $\{x_1,x_2,x_3\}$, and the threshold vector $\boldsymbol{Y}$ is $\{y_1,y_2,y_3\}$. $z_1,\cdots,z_4$ are leaf nodes, which correspond to different classification results. If $f_k(\boldsymbol{X})=0$, the left branch of the node is taken, otherwise, the right branch is taken. Once we reach a leaf node, the decision value for that node is outputted. We denote the longest length from the root node to any leaf node as the depth of the tree. Without loss of generality, we assume the decision trees are complete binary trees.

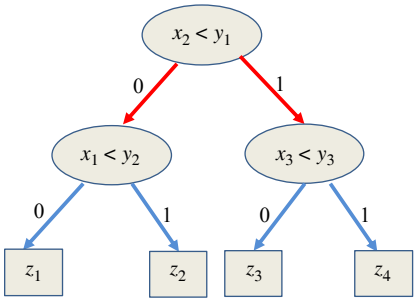


Figure 2 A decision tree

B. Homomorphic Encryption

An additively homomorphic public key encryption scheme is composed of three algorithms: KeyGen, Encryption, and Decryption. The key generation algorithm outputs a pubic key pk and a private key sk. Encryption algorithm takes as input a message τ and a random number s, and generates an ciphertext c. We denote an encryption of τ as $\text{Enc}_{\text{pk}}(\tau)$. Decryption algorithm takes as input a ciphertext c and the private key sk, and outputs a plaintext. By utilizing the additive homomorphism property, given the encryptions of two messages τ_0 and τ_1, one can output an encryption of $\tau_1+\tau_2$. In addition, given an encryption of τ and a scalar l, the encryption of $l\tau$ can be homomorphically calculated.

In our scheme, we use a variant of Paillier's encryption with fast decryption[25] as the homomorphic encryption scheme. The details of the scheme are shown in Tab. 1. In this fast variant, the trapdoorness relies on the knowledge of α instead of λ, making the decryption algorithm run in complexity $O(|n|^2|\alpha|)$ rather than $O(|n|^3)$ in the original scheme. Thus, it can reduce the computation cost of the client, who may have limited computing power compared with the service provider. In the scheme, $L(x)=(x-1)/N$, and λ here is $lcm(p-1,q-1)$. $\mathscr{B}_\alpha$ denotes the elements of order $N\alpha$, where $1\leqslant\alpha\leqslant\lambda$.

V. Privacy-Preserving Decision Tree Classification

In this section, we first present our secure comparison protocol, then show the detailed construction of our proposed PDTC.

Table 1 The variant of Paillier encryption with fast decryption

Choose two large prime numbers p and q
Public key (pk):
$N = pq$, a base $g \in \mathscr{B}$
Private key (sk):
λ
Encryption:
randomly select r
$c = \text{Enc}_{\text{pk}}(\tau) = g^{\tau} r^n \bmod N^2$
Decryption:
$\tau = Dec_{\text{sk}}(c) = \frac{L(c^{\lambda} \bmod N^2)}{L(g^{\lambda} \bmod N^2)} \bmod N$
Homomorphic:
$\text{Enc}_{\text{pk}}(\tau_1)\text{Enc}_{\text{pk}}(\tau_2) \bmod N^2 = \text{Enc}_{\text{pk}}(\tau_1 + \tau_2) \bmod N^2$
$\text{Enc}_{\text{pk}}(\tau)^l \bmod N^2 = \text{Enc}_{\text{pk}}(l\tau) \bmod N^2$

A. Secure Comparison Protocol

One of the main building blocks to construct privacy-preserving decision tree classification is a secure comparison protocol. In private comparison protocols, there are two parties, P_1 and P_2, where P_1 holds a private value v_1 and P_2 holds a private value v_2. At the end of the protocol, P_1 and P_2 can determine the comparison result $1\{x < y\}$ without knowing the other party's input.

To compare v_0 and v_1, we compute $w = v_0 - v_1$ and check whether w is positive. The difference w between the two values should be hidden, and only the comparison result can be revealed by the two parties[26,27]. The basic idea of our protocol is that the comparison is performed on a finite field of modulus N and we use homomorphic encryption to compare the values without knowing the plaintexts. Moreover, multiplicative hiding is utilized to hide the w by multiplying it with a large random number r. In the finite field, negative values can be denoted as the upper half of the interval $[0, N-1]$, which means

$$\left[-\left\lfloor \frac{N}{2} \right\rfloor, -1\right] \equiv \left[\left\lceil \frac{N}{2} \right\rceil, N-1\right].$$

We denote $||v||$ as the bit length of a value v. Given a large number N and two positive values v_0 and v_1, we can obtain that $(v_0 - v_1)r > N/2 \Leftrightarrow v_0 < v_1$ if the bit lengths of v_0, v_1, and r are less than $||N||/2 - 1$. From this property, we can compare the two values without revealing the difference w. Moreover, although w is randomized by r, one may factor the result to get the information of w. To prevent this information leakage, we add a small value r' to it. The process of the comparison protocol is shown as below.

P_1 encrypts its private value v_1 using the Paillier cryptosystem and sends the ciphertext to P_2.

P_2 randomly chooses r and r', where $||r|| \leqslant ||N||/2$, and $r' < r$. Then, P_2 homomorphically calculates the encryption of $r(v_1 - v_2) + r'$ using the public key of P_1, and returns the ciphertext back to P_1.

To learn the comparison result, P_1 decrypts the ciphertext and compares the result with $N/2$. If it is greater than $N/2$, P_1 knows that $v_1 < v_2$, otherwise, $v_1 \geqslant v_2$.

At the end of the protocol, P_1 learns the comparison result and P_2 learns nothing.

In order for P_1 and P_2 jointly determine the comparison result, which means that P_1 cannot learn the actual result without the information of P_2, we combine the secret sharing during the comparison. To be specific, after receiving the encryption of v_1, P_2 randomly chooses a bit $b \leftarrow \{0,1\}$. If $b = 0$, P_2 returns the encryption of $r(v_1 - v_2) + r'$ to P_1; otherwise, it replies with $r(v_2 - v_1) + r'$. P_1 decrypts the ciphertext, and compares the result with $N/2$. If it is greater than $N/2$, P_1 sets $b' = 1$. We can obtain that $b \bigoplus b' = 1$ if $v_1 < v_2$.

B. Detailed Construction of PDTC

In our system, a client with an input vector $\boldsymbol{X}$ interacts with a service provider who holds a private decision tree model to obtain the correct classification result without revealing the input. The client and the service provider first generate their public-private key pairs and register with a TA. The client generates its query by encrypting the features in $\boldsymbol{X}$ with the fast variant of Paillier's encryption, and sends the ciphertext of $\boldsymbol{X}$ to the service provider. Based on our secure comparison protocol, the client and service provider obliviously evaluate the decision tree and secretly share the comparison results at each decision node. Following the idea of TAI et al.[16] that assigns the edge cost for each edge in the decision tree based on the comparison result at the corresponding decision node, we represent each leaf node as the sum of the edge costs along the path and the classification value related to the leaf node. After the evaluation, the service provider returns the encrypted classification result to the client, who can recover it using its private key. Overall, our scheme consists of three phases: query vector generation, numeric comparison at decision nodes, and path evaluation. The notations used in our protocol are shown in Tab. 2.

1) Query Vector Generation: The client and the service provider generate the public-private key pair $(\text{pk}_c, \text{sk}_c)$, $(\text{pk}_s, \text{sk}_s)$ based on the KeyGen algorithm of the fast variant of Paillier encrytion. Then, they register themselves with a TA, who is responsible for issuing public certificates that can be used for authentication. Assume pk_c is (g, N) and sk_c is α. For a query vector $\boldsymbol{X} = \{x_1, x_2, \cdots, x_n\}$, where $||x_i|| \leqslant ||n||/2 - 1$, $i \in [n]$, the client encrypts each feature x_i using its public key by randomly choosing a number $s < N$ and calculating the ciphertext as $\langle x_i \rangle = g^{x_i + Ns} \bmod n^2$. Then, the client sends the generated ciphertext vector $\langle \boldsymbol{X} \rangle$ to the service provider. Here, we assume the service provider and the client can authenticate with each other by using the public key certificates.

Table 2 System parameters

Acronym	Definition
n	dimension of a feature vector
d	the depth of a tree
D_k	a decision node
m	the number of decision nodes in a tree
$\mathrm{pk}_c, \mathrm{sk}_c$	the public-private key pair of the cleint
$\mathrm{pk}_s, \mathrm{sk}_s$	the public-private key pair of the service provider
$[n]$	the set of integers $\{1, 2 \cdots, n\}$
$\langle \tau \rangle$	an encryption of τ under Paillier cryptosystem
$\|\|x\|\|$	the bit length of x
b_k	the comparison result at the decision node D_k
$E_{k,0}, E_{k,1}$	the left, right edge of decision node D_k
L_i	the ith leaf node
P_i	the path from the root node to the L_i
$T(\boldsymbol{X})$	the classification result of the input $\boldsymbol{X}$

2) Numeric Comparison at Decision Nodes: The service provider and the client cooperate to obtain the comparison result of the client's input and a threshold value at each decision node. Denote m as the number of the decision nodes in the tree and the threshold corresponding to the decision node D_k is y_k, where $k \in [m]$. Let $f_k(\boldsymbol{X}) = 1\{x_{i_k} < t_k\}$ be the Boolean function associated with D_k. For the decision node D_k, the service provider chooses a random bit b_{k_1} and two random numbers r_k and r'_k, where $||r_k|| \leqslant ||N||/2 - 1$, and $r'_k < r_k$. For the query vector $\langle \boldsymbol{X} \rangle = (\langle x_1 \rangle, \cdots \langle x_n \rangle)$ received from the client, the service provider calculates the comparison result based on $\langle x_{i_k} \rangle$ and y_k. The process for secure comparison between the client and the service provider is shown as follows.

• Service provider. For each decision node D_k, $k \in [m]$, if $b_{k_1} = 0$, the service provider computes the ciphertext $\mathrm{Enc}_{\mathrm{pk}_c}(c_k)$ as

$$\begin{aligned}\mathrm{Enc}_{\mathrm{pk}_c}(c_k) = (\langle x_i \rangle \mathrm{Enc}_{\mathrm{pk}_c}(N - y_k))^{r_k} \mathrm{Enc}_{\mathrm{pk}_c}(r'_k) = \\ (\mathrm{Enc}_{\mathrm{pk}_c}(x_{i_k}) \mathrm{Enc}_{\mathrm{pk}_c}(N - y_k))^{r_k} \mathrm{Enc}_{\mathrm{pk}_c}(r'_k) = \\ \mathrm{Enc}_{\mathrm{pk}_c}(r_k(x_{i_k} - y_k) + r'_k).\end{aligned}$$

If $b_{k_1} = 1$, the service provider calculates $\mathrm{Enc}_{\mathrm{pk}_c}(c_k)$ as

$$\begin{aligned}\mathrm{Enc}_{\mathrm{pk}_c}(c_k) = \left(\mathrm{Enc}_{\mathrm{pk}_c}(y_k) \langle x_i \rangle^{N-1}\right)^{r_k} \mathrm{Enc}_{\mathrm{pk}_c}(r'_k) = \\ \left(\mathrm{Enc}_{\mathrm{pk}_c}(y_k) \mathrm{Enc}_{\mathrm{pk}_c}(x_{i_k})^{N-1}\right)^{r_k} \mathrm{Enc}_{\mathrm{pk}_c}(r'_k) = \\ \mathrm{Enc}_{\mathrm{pk}_c}(r_k(y_k - x_{i_k}) + r'_k).\end{aligned}$$

Then, the service provider sends the ciphertexts $\{\mathrm{Enc}_{\mathrm{pk}_c}(c_k)\}_{k \in [m]}$ to the client.

• Client. For $k \in [m]$, the client decrypts the ciphertext $\mathrm{Enc}_{\mathrm{pk}_c}(c_k)$ using its secret key sk_c and obtains the c_k. If $c_k > N/2$, the client sets $b_{k_2} = 1$; otherwise, it sets $b_{k_2} = 0$. Note that b_{k_2} is actually a secret share of the comparison result at the decision node D_k. Then, the client encrypts the b_{k_2} using its public key and sends the ciphertexts $\langle b_{k_2} \rangle$, $k \in [m]$ to the service provider.

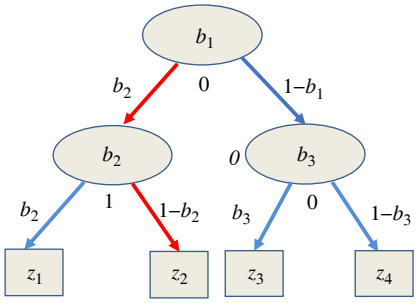


Figure 3 Edge cost of a decision tree

• Service provider. For $k \in [m]$, the service provider computes $\langle b_k \rangle = \langle b_{k_1} \oplus b_{k_2} \rangle$ homomorphically using the fact that for a bit $x \in \{0,1\}, x \oplus 0 = x$, and $x \oplus 1 = 1 - x$. The resulting b_k corresponds to the actual comparison result at the decision node D_k.

3) Path Evaluation: For each decision node D_k, $k \in [m]$, let b_k be the result of $1\{x_{i_k} < y_k\}$. If $b_k = 1$, which means $x_{i_k} < y_k$, the classification result u of the input $\boldsymbol{X}$ is in the right subtree of the node D_k; otherwise, the result is in the left subtree of D_k. We assign the edge cost of the left outgoing edge of D_k as $ec_{k,0} = b_k$, and the edge cost of the right outgoing edge of D_k as $ec_{k,1} = 1 - b_k$. Let $E_{k,0}$ ($E_{k,1}$) denote the left (right) edge of the D_k. For $i \in [m+1]$, let P_i denote the path of the leaf node z_i from the root node. The path cost pc of a leaf node can be represented as the summation of the edge cost along the path. As shown in Fig. 3, the path cost of the leaf nodes are: $pc_1 = b_1 + b_2, pc_2 = b_1 + (1 - b_2), pc_3 = (1 - b_1) + b_3$, and $p_4 = (1 - b_1) + (1 - b_3)$. If $b_1 = 0$ and $b_2 = 1$, we can see that only the path cost of the leaf node z_2 is 0. That is, if the z_k represents the final classification result, the path cost pc_k is 0. We can use this property to evaluate the decision tree so that the client can recover the classification result of its input. The process of path evaluation is shown as below.

• Service provider. For $i \in [m+1]$, the service provider calculates the path cost pc_i of each leaf node z_i as $\langle pc_i \rangle = \langle \sum_{E_{k,j} \in P_i} ec_{k,j} \rangle$, where $k \in [m]$ and $j \in \{0,1\}$. Then, the service provider chooses two random values h_i and h'_i from Z_N, and computes $\langle \overline{pc_i} \rangle = \langle h_i \cdot pc_i \rangle$. Assume the classification result corresponds to the leaf node z_i is v_i, which is an element in Z_N. The service provider computes the ciphertext of v_i as

$$\langle \overline{v_i} \rangle = \langle h'_i \cdot pc_i + v_i \rangle.$$

Finally, the service provider sends $\{\langle \overline{pc_i} \rangle, \langle \overline{v_i} \rangle\}_{i \in [m+1]}$ to the client in random order.

• Client. After receiving the ciphertexts, the client decrypts the ciphertexts $\langle \overline{pc_i} \rangle, i \in [m+1]$ and checks whether $\overline{pc_i} = 0$. If $\overline{pc_{i'}} = 0$, where $i' \in [m+1]$, the client recovers the classification result by decrypting the $\langle \overline{v_i} \rangle$ using its secret key.

VI. Security Analysis

In this section, we prove that the client data privacy and model parameter privacy are preserved in the proposed scheme by showing that the scheme is secure against a semi-honest service provider and secure against a semi-honest client.

Theorem 1 The proposed decision tree classification scheme is secure against semi-honest adversaries.

Security against a semi-honest service provider. Denote π as the proposed protocol, and f as the secure decision tree classification function. We first prove that π is secure against a semi-honest service provider, which means the privacy of the client data is preserved. Assume adversary $\mathscr{A}$ is a semi-honest service provider in the real world execution. We construct an ideal world simulator $\mathscr{S}$ and prove that $|Pr[\text{REAL}_{\pi,\mathscr{A},\mathscr{Z}}(\lambda)=1]-Pr[\text{IDEAL}_{f,\mathscr{S},\mathscr{Z}}(\lambda)=1]|$ is negligible. The ideal-world S acts as follows.

• $\mathscr{S}$ first receives the input T from the environment $\mathscr{Z}$, and sends it to the trusted party. After receiving the inputs from the client and $\mathscr{S}$, the trusted party performs f and returns the result to the client.

• $\mathscr{S}$ generates a public-private key pair (pk,sk) for the Paillier cryptosystem, and for $i\in[n]$, $\mathscr{S}$ randomly chooses n random values x_i, where $||x_i||\leqslant||N||/2-1$, and N is a component of pk. Then, $\mathscr{S}$ sends the $\text{Enc}_{\text{pk}}(x_i)$ to $\mathscr{A}$.

• After $\mathscr{A}$ responds with the ciphertexts c_k, where $k\in[m]$, $\mathscr{S}$ randomly chooses b_k from $\{0,1\}$, and sends the encryptions $\text{Enc}_{\text{pk}}(b_k)$ to $\mathscr{A}$. After receiving the response from $\mathscr{A}$, $\mathscr{S}$ outputs it to $\mathscr{Z}$.

Since the adversary is semi-honest, the client would obtain the classification result $T(\boldsymbol{X})$ from the service provider, where $\boldsymbol{X}$ is the input of the client. In the ideal world, the trusted party executes the function f and returns the result to the client. Thus, the outputs of the client in the real-world execution and ideal world execution are the same.

We then prove that the outputs of $\mathscr{A}$ and $\mathscr{S}$ to the environment $\mathscr{Z}$ have an identical distribution. In the real world, $\mathscr{A}$ receives the encryptions of $\{x_i\},i\in[n]$, and the encryptions of $b_{k_2},k\in[m]$, which are secret shares of the comparison results. When $\mathscr{S}$ interacts with $\mathscr{A}$, $\mathscr{A}$ receives n independent encryptions of random numbers, and also m encryptions of $b_k,k\in[m]$, where b_k are randomly chosen from $\{0,1\}$. Since the Paillier cryptosystem is semantically secure, from the view of $\mathscr{A}$, it is computationally indistinguishable whether it is interacting with the client in the real world or interacting with $\mathscr{S}$. Thus, the view and output distributions of $\mathscr{A}$ and $\mathscr{S}$ are computationally indistinguishable, and $\text{REAL}_{\pi,\mathscr{A},\mathscr{Z}}$ and $\text{IDEAL}_{f,\mathscr{S},\mathscr{Z}}$ have the same distribution for $\mathscr{Z}$. Therefore, the protocol is secure against a semi-honest service provider.

Security against a semi-honest client. Let $\mathscr{A}$ be a semi-honest client in the real-world execution. We prove that π is secure against semi-honest client, which means the model privacy of the service provider is preserved. We construct a simulator $\mathscr{S}$ as follows.

• First, $\mathscr{S}$ receives the input vector $\boldsymbol{X}$ and sends it to the trusted party, who executes the function f and returns the result $\hat{v}$ to $\mathscr{S}$.

• $\mathscr{S}$ starts running $\mathscr{A}$ on input $\boldsymbol{X}$. Let $\text{pk}=(g,N)$ be the public key of $\mathscr{A}$. For $k\in[m]$, $\mathscr{S}$ randomly chooses b_k from $\{0,1\}$.

• After receiving the encryptions of the feature vector $\boldsymbol{X}$ from $\mathscr{A}$, $\mathscr{S}$ responds to $\mathscr{A}$ as follows.

For $k\in[m]$, if b_k=0, $\mathscr{S}$ chooses a random value $\hat{c}_k$ from Z_N, where $\hat{c}_k<N/2$.

Otherwise, $\mathscr{S}$ chooses a random value $\hat{c}_k$ from Z_N, where $\hat{c}_k\geqslant N/2$.

$\mathscr{S}$ returns the collection of the ciphertexts $\{\text{Enc}_{\text{pk}}(\hat{c}_k)\}_{k\in[m]}$ to $\mathscr{A}$.

• After receiving the encrypted bit string from $\mathscr{A}$, $\mathscr{S}$ chooses a random index $i^*\xleftarrow{R}[m+1]$, and sets $\hat{pc}_{i^*}=0$. For $i\in[m+1]$ and $i\neq i^*$, $\mathscr{S}$ samples $\hat{pc}_i\xleftarrow{R}Z_N$. Then, $\mathscr{S}$ sets $\hat{v}_{i^*}=\hat{v}$. For $i\in[m+1]$ and $i\neq i^*$, $\mathscr{S}$ samples $\hat{v}_i\xleftarrow{R}Z_N$. Finally, $\mathscr{S}$ sends the encryptions $\{\text{Enc}_{\text{pk}}(\hat{pc}_i),\text{Enc}_{\text{pk}}(\hat{v}_i)\}_{i\in[m+1]}$ to $\mathscr{A}$.

• $\mathscr{S}$ outputs whatever $\mathscr{A}$ outputs.

Since the honest service provider has no output at the end of the protocol, we only need to show that the output of $\mathscr{S}$ to $\mathscr{Z}$ is computationally indistinguishable from the output of $\mathscr{A}$ to $\mathscr{Z}$.

In the real world, what the client receives from the service provider consists of two components: the ciphertexts of the comparison results $\{c_k\}_{k\in[m]}$ at each decision node, and the final classification result v of the input $\boldsymbol{X}$. For the distribution of $c_k,k\in[m]$, since the service provider chooses b_{k_1} uniformly and $b_{k_1}\oplus b_{k_2}=1\{x_{i_k}<y_k\}$, where $k\in[m],i_k\in[n]$, for a fix index $j\in[m]$, there is an equal probability that $c_k\leqslant[N]/2$ or $c_k>[N]/2$. We can see that $\hat{c}_k$ sampled by $\mathscr{S}$ and c_k have the same distribution. In addition, since the service provider here is honest, for the client, the received classification result v from the service provider is the same as the value $\hat{v}$ received from the trusted party. Thus, it is computationally indistinguishable whether the client is interacting with $\mathscr{S}$ or the service provider, and the view and outputs of $\mathscr{S}$ and $\mathscr{A}$ are computationally indistinguishable. Therefore, the protocol is secure against a semi-honest client.

VII. Performance Evaluation

A. Numerical Analysis

As shown in section V, our scheme consists of three phases: query vector generation, secure comparison, and path evaluation. Let n be the dimension of the feature vector, m be the

Table 3 Complexity comparison

Scheme	Complexity		
	Client	Service provider	Rounds
BOST et al.[18]	$O((n+m)t)$	$O(mt)$	$\geqslant 6$
WU et al.[15]	$O((n+m)t+d)$	$O(mt+2^d)$	6
TAI et al.[16]	$O((n+m)t)$	$O(mt)$	4
this work	$O(n+m)$	$O(m)$	4

number of the decision nodes in the tree, d be the depth of the tree, and t be the number of bits needed to represent one feature. In the phase of query vector generation, the client needs to encrypt n ciphertexts and sends them to the service provider. During the secure comparison, the service provider computes m ciphertexts and returns them to the client. For the client, it needs to decrypt m ciphertexts, then encrypts m resulting comparison bits and sends the ciphertexts to the service provider. In the phase of path evaluation, the service provider generates the ciphertexts of the path cost and classification value for each leaf nodes, which means the service provider needs to compute $2(m+1)$ ciphertexts and sends them to the client. For the client, it needs to perform at most $m+2$ decryptions to get the final classification result. We compare the computational complexity of our protocol with Refs. [18], [15] and [16], as shown in Tab. 3.

B. Simulation Results

To evaluate the performance of the proposed scheme, we conduct the simulations on Windows 10 enterprise with Intel Core i7-7500U CPU @2.9 GHz and 8 GB RAM memory. We implement the Paillier cryptosystem in Visual Studio 2012, and generate the public and private keys by calling the functions of Miracl library[28]. The prime p and q in the private key have a bit length of 512 in our experiments.

We measure the computation cost for both the client and service provider on five datasets from UCI repository[29], which are the same as the datasets for Refs. [15,16], and application domains include heart disease diagnosis and credit assessment. Tab. 4 demonstrates the computation time and the bandwidth needed for the client and the service provider to complete the decision tree classification. From Tab. 4, we can see the proposed scheme has good performance in various applications. Even for the spambase data which have high dimension input vectors and the tree model has large depth, the time cost required for the client is less than 2 s and the bandwidth needed is about 0.08 MB.

Note that in our protocol, the client and the service provider do not need to encrypt the values to be compared in bits, which greatly reduces the computation and communication overhead for both two parties. Moreover, for the service provider, there is no need to transform a non-complete tree into a complete binary tree. Thus, our protocol is also highly efficient for sparse

Table 4 Performance on UCI datasets

Dataset	n	d	m	Computation (s)		Bandwidth (KB)	
				Client	Server	Upload	Download
breast-cancer	9	8	12	0.297	0.114	5.376	9.728
heart-disease	13	3	5	0.162	0.051	4.608	4.352
housing	13	13	92	1.989	0.834	26.88	71.168
credit-screening	15	4	5	0.168	0.051	5.12	4.352
spambase	57	17	58	1.989	0.528	29.44	45.056

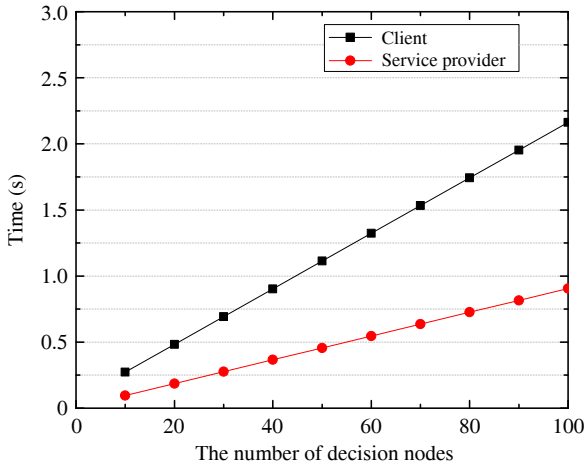


Figure 4 Computation cost vs. the number of decision nodes

trees. When the number of the decision nodes goes from 10 to 100, the computation time and the bandwidth required for the client and the service provider are shown in Fig. 4 and Fig. 5, where the dimension of feature vectors is set to 15. Since the client needs to perform $n+m$ encryption operations and $2m+2$ decryption operations, when n is fixed as 15, the computation cost for the client increases linearly with the number of decision nodes. For the service provider, it needs to execute $3m+2$ encryption operations, the computation cost for the service provider also grows linearly with m. For the bandwidth required, the client needs to upload $n+m$ ciphertexts and download $3m+2$ ciphertexts, when the dimension of feature vectors is fixed, the communication overhead also grows linearly with the number of decision nodes.

VIII. Conclusion

In this paper, we have proposed a secure and privacy-preserving decision tree classification scheme, where the inputs and the classification results of the client are concealed from the service provider, and privacy of the model parameters is also preserved from the client. With the design of an efficient secure comparison protocol, the proposed decision tree classification scheme has achieved a lower computation and communication overhead for both the client and

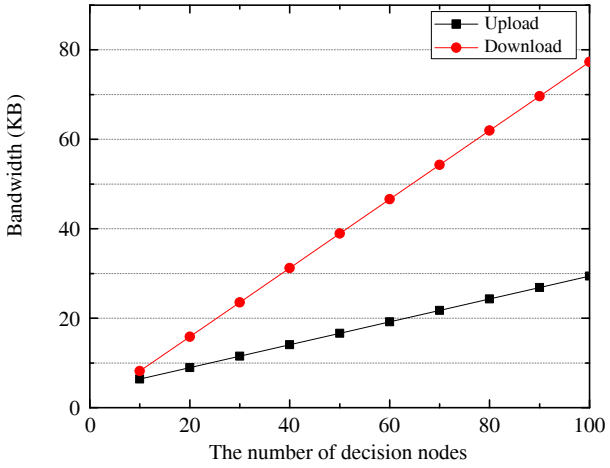


Figure 5 Bandwidth vs. the number of decision nodes

the service provider. Moreover, the formal security proof has demonstrated that the proposed scheme achieves the desired properties under the semi-honest model. In future work, we will investigate the security and privacy requirements of other machine-learning classification models as well as their design and implementation challenges for real-world applications.

REFERENCES

[1] WAN N, WEINBERG D, LIU T Y, et al. Machine learning enables detection of early-stage colorectal cancer by whole-genome sequencing of plasma cell-free DNA[J]. BMC Cancer, 2019, 19(1): 832.

[2] KUMAR S, GAO X, WELCH I, et al. A machine learning based web spam filtering approach[C]//2016 IEEE 30th International Conference on Advanced Information Networking and Applications (AINA). Piscataway: IEEE Press, 2016: 973-980.

[3] LIANG J, QIN Z, XIAO S, et al. Efficient and secure decision tree classification for cloud-assisted online diagnosis services[J]. IEEE Transactions on Dependable and Secure Computing, 2019.

[4] HUANG C, LU R, CHOO K K R. Secure and flexible cloud-assisted association rule mining over horizontally partitioned databases[J]. Journal of Computer and System Sciences, 2017, 89: 51-63.

[5] FREDRIKSON M, JHA S, RISTENPART T. Model inversion attacks that exploit confidence information and basic countermeasures[C]//Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security. New York: ACM, 2015: 1322-1333.

[6] TRAMER F, ZHANG F, JUELS A, et al. Stealing machine learning models via prediction APIs[C]//25th USENIX Security Symposium. Berkeley: USENIX Association, 2016: 601-618.

[7] HUANG C, LU R, LIN X, et al. Secure automated valet parking: A privacy-preserving reservation scheme for autonomous vehicles[J]. IEEE Transactions on Vehicular Technology, 2018, 67(11): 11169-11180.

[8] KELLER M, ORSINI E, SCHOLL P. MASCOT: Faster malicious arithmetic secure computation with oblivious transfer[C]//Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security. New York: ACM, 2016: 830-842.

[9] DAMGARD I, PASTRO V, SMART N, et al. Multiparty computation from somewhat homomorphic encryption[C]//Annual Cryptology Conference. Berlin, Heidelberg: Springer-Verlag, 2012: 643-662.

[10] LIU D, ALAHMADI A, NI J, et al. Anonymous reputation system for IIoT-enabled retail marketing atop PoS blockchain[J]. IEEE Transactions on Industrial Informatics, 2019, 15(6): 3527-3537.

[11] FRANZ M, HOLZER A, KATZENBEISSER S, et al. CBMC-GC: An ANSI C compiler for secure two-party computations[C]//International Conference on Compiler Construction. Berlin, Heidelberg: Springer-Verlag, 2014: 244-249.

[12] GENTRY C. Fully homomorphic encryption using ideal lattices[C]//Proceedings of the 41st Annual ACM Symposium on Theory of Computing. New York: ACM, 2009: 169-178.

[13] NI J, ZHANG K, XIA Q, et al. Enabling strong privacy preservation and accurate task allocation for mobile crowdsensing[J]. IEEE Transactions on Mobile Computing, 2019.

[14] LIU C, WANG X S, NAYAK K, et al. Oblivm: A programming framework for secure computation[C]//2015 IEEE Symposium on Security and Privacy. Piscataway: IEEE Press, 2015: 359-376.

[15] WU D J, Feng T, NAEHRIG M, et al. Privately evaluating decision trees and random forests[J]. Proceedings on Privacy Enhancing Technologies, 2016, 2016(4): 335-355.

[16] TAI R. K. H, MA J. P. K, Zhao Y, et al. Privacy-preserving decision trees evaluation via linear functions[C]//European Symposium on Research in Computer Security. Cham, Switzerland: Springer-Verlag, 2017: 494-512.

[17] BRICKELL J, PORTER D E, SHMATIKOV V, et al. Privacy-preserving remote diagnostics[C]//Proceedings of the 14th ACM conference on Computer and communications security. New York: ACM, 2007: 498-507.

[18] BOST R, POPA R A, TU S, et al. Machine learning classification over encrypted data[C]//NDSS. Reston: Internet Society, 2015, 4324-4325.

[19] DAMGÅRD I, GEISLER M, KRØIGAARD M. Efficient and secure comparison for on-line auctions[C]//Australasian Conference on Information Security and Privacy. Berlin, Heidelberg: Springer-Verlag, 2007: 416-430.

[20] TUENO A, KERSCHBAUM F, KATZENBEISSER S. Private evaluation of decision trees using sublinear cost[J]. Proceedings on Privacy Enhancing Technologies, 2019, 2019(1): 266-286.

[21] KISS A, NADERPOUR M. Naderpour, LIU J, et al. SoK: Modular and efficient private decision tree evaluation[J]. Proceedings on Privacy Enhancing Technologies, 2019, 2019(2): 187-208.

[22] TUENO A, BOEV Y, KERSCHBAUM F. Non-interactive private decision tree evaluation[J]. arXiv:1909.08362, 2019.

[23] ZHENG Y, DUAN H, WANG C. Towards secure and efficient outsourcing of machine learning classification[C]//European Symposium on Research in Computer Security. Cham, Switzerland: Springer-Verlag, 2019: 22-40.

[24] KATZ J. On achieving the “best of both worlds” in secure multiparty computation[C]//Proceedings of the 39th Annual ACM Symposium on Theory of Computing. New York: ACM, 2007: 11-20.

[25] PAILLIER P. Public-key cryptosystems based on composite degree residuosity classes[C]//International Conference on the Theory and Applications of Cryptographic Techniques. 1999: 223-238.

[26] KERSCHBAUM F, BISWAS D, HOOGH S D. Performance comparison of secure comparison protocols[C]//20th International Workshop on Database and Expert Systems Application. Piscataway: IEEE Press, 2009: 133-136.

[27] LIU L, SU J, CHEN R, et al. Secure and fast decision tree evaluation on outsourced cloud data[C]//International Conference on Machine Learning for Cyber Security. Cham, Switzerland: Springer-Verlag, 2019: 361-377.

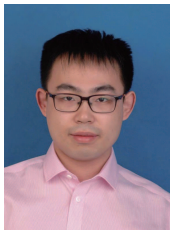
[28] XUE L, LIU D, NI J, et al. Balancing privacy and accountability for industrial mortgage management[J]. IEEE Transactions on Industrial

Informatics, 2019.
[29] BACHE K, LICHMAN M. UCI machine learning repository[EB].

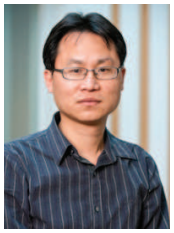
About the Authors

Liang Xue received her B.S. and M.S. degrees in the School of Computer Science and Engineering, University of Electronic Science and Technology of China (UESTC), Chengdu, China in 2015 and 2018, respectively. Currently, She is pursuing her Ph.D. degree at the Department of Electrical and Computer Engineering, University of Waterloo, Canada. Her research interests include applied cryptography, cloud computing, and blockchain technologies.

Dongxiao Liu received his B.S. and M.S. degrees in the School of Computer Science and Engineering, University of Electronic Science and Technology of China (UESTC), Chengdu, China in 2013 and 2016, respectively. Currently, he is pursuing his Ph.D. degree at the Department of Electrical and Computer Engineering, University of Waterloo, Canada. His research interests include applied cryptography and privacy enhancing technologies for blockchain.



Cheng Huang received his B.Eng. and M.Eng. degrees from Xidian University, Xi'an, China, in 2013 and 2016 respectively, and was a project officer with the INFINITUS laboratory at the School of Electrical and Electronic Engineering, Nanyang Technological University till July 2016. Currently, he is working towards his Ph.D. degree with the Department of Electrical and Computer Engineering, University of Waterloo, ON, Canada. His research interests are in the areas of applied cryptography, cyber security and privacy in the mobile network.



Xiaodong Lin [corresponding author] received his Ph.D. degree in information engineering from Beijing University of Posts and Telecommunications, Beijing, China, and his Ph.D. degree (with Outstanding Achievement in Graduate Studies Award) in electrical and computer engineering from University of Waterloo, Canada. He is currently an associate professor in the School of Computer Science at University of Guelph, Canada. His research interests include computer and network security, privacy protection, applied cryptography, computer forensics, and software security. He is a fellow of the IEEE.

Xuemin (Sherman) Shen received his Ph.D. degree in electrical engineering from Rutgers University, New Brunswick, NJ, USA, in 1990. He is currently a university professor with the Department of Electrical and Computer Engineering, University of Waterloo, Canada. His research focuses on network resource management, wireless network security, Internet of things, 5G and beyond, and vehicular ad hoc and sensor networks. He is a registered professional engineer of Ontario, Canada, an Engineering Institute of Canada fellow, a Canadian Academy of Engineering fellow, a Royal Society of Canada fellow, a Chinese Academy of Engineering Foreign fellow, and a distinguished lecturer of the IEEE Vehicular Technology Society and Communications Society.

Dr. Shen received the R.A. Fessenden Award in 2019 from IEEE, Canada, James Evans Avant Garde Award in 2018 from the IEEE Vehicular Technology Society, Joseph LoCicero Award in 2015 and Education Award in 2017 from the IEEE Communications Society. He has also received the Excellent Graduate Supervision Award in 2006 and Outstanding Performance Award 5 times from University of Waterloo and the Premier's Research Excellence Award (PREA) in 2003 from the Province of Ontario, Canada. He served as the technical program committee chair/co-chair for the IEEE Globecom'16, the IEEE Infocom'14, the IEEE VTC'10 Fall, the IEEE Globecom'07, the symposia chair for the IEEE ICC'10, and the chair for the IEEE Communications Society Technical Committee on Wireless Communications. He was the editor-in-chief of the IEEE Internet of Things Journal and IEEE Network, and the vice president on publications of the IEEE Communications Society.