Secure and Efficient *k*NN Classification for Industrial Internet of Things

Haomiao Yang, *Member, IEEE*, Shaopeng Liang, Jianbing Ni, *Member, IEEE*, Hongwei Li, *Senior Member, IEEE*, and Xuemin (Sherman) Shen, *Fellow, IEEE*

***Abstract*—The *k*-nearest neighbors (*k*NN) classification has been widely used for defective product identification and anomaly detection in the Industrial Internet of Things (IIoT). In this article, we propose a secure and efficient distributed *k*NN classification algorithm (SEED-*k*NN) to prevent information and control flow exposure while supporting large-scale data classification on distributed servers. Specifically, we first design a secure and efficient vector homomorphic encryption (VHE) scheme by constructing a key-switching matrix and a noise matrix for data encryption. Based on the designed VHE, SEED-*k*NN is proposed to efficiently achieve the confidentiality of data flow, *k*NN query, and class label, while enabling homomorphic operations on the encrypted data. Moreover, by leveraging the Map/Reduce architecture, SEED-*k*NN enables the *k*NN classification over the large-scale encrypted data on distributed servers for industrial control systems. Finally, we demonstrate that SEED-*k*NN achieves semantic security and high classification accuracy, and is applicable in IIoT due to its high efficiency.**

***Index Terms*—Big data, Industrial Internet of Things (IIoT), intelligent systems, machine learning (ML), security and privacy.**

I. Introduction

INDUSTRIAL Internet of Things (IIoT) uses interconnected sensors, actuators, and other equipment to promote the manufacturing and industrial process. The IIoT, also known as Industry 4.0, is an evolution of industrial control systems that shifts from traditional supervisory control and data acquisition (SCADA) technologies to cloud-based systems, potentially improving productivity and efficiency. Especially, due to the great advances of artificial intelligence technology, intelligent control, in its attempt to mimic a human operator, has brought new avenues to make industrial control systems smart, automatic, and economical. The intelligent industrial control [1], [2] is able to monitor and control the industrial processes and extract the most valuable information from data for planning and decision making. Intelligence has flourished in various domains, such as intelligent logistics, smart robotics, and smart automotive manufacturing. By applying intelligence to industrial control systems, numerous machine learning (ML) algorithms, such as support vector machine, decision tree, and neural network, have been used for the information and control flow classification from complex systems and environment [3], [4]. For example, in industrial manufacturing, the ML algorithms are utilized to train the model of fault detection from the historical data for the rapid detection of future abnormal behavior, and the intelligence of traffic light control is built based on ML algorithms to mitigate traffic congestion and improve travel efficiency for drivers [5]–[7].

As one of the popular ML algorithms, the *k*-nearest neighbors (*k*NN) classify an object to the class most common among its *k*-nearest neighbors based on a plurality vote of its neighbors. The *k*NN algorithm can acquire high accuracy in defective product identification [8] and anomaly detection [9] for IIoT. However, the *k*NN algorithm has high computational complexity and it consumes large amounts of computing resources if there is massive data with a large number of features to be analyzed. To address this issue, the powerful cloud server is employed to perform the expensive *k*NN algorithm over the collected data. By migrating the heavy computational overhead to the cloud, the bottleneck of the *k*NN algorithm can be addressed, making it suitable to be used in IIoT.

The knowledge used to form the training data set in *k*NN is extracted from data flows, system states, or measurements of various sensors. Generally, these data contain plenty of important information that manufacturers or corporations are unwilling to expose to the public, such as industrial process flows, product information, business secrecy, and traceable data of individuals [10]. To preserve the confidentiality of these data, a widely used approach is to encrypt data before uploading them to the centralized server [11]–[13]. Nevertheless, traditional ciphers may affect data availability, which means that it is difficult to conduct data analytics directly over ciphertexts. To address this problem, secure *k*NN

Manuscript received April 1, 2020; accepted April 27, 2020. Date of publication May 6, 2020; date of current version November 12, 2020. This work was supported in part by the National Key Research and Development Program of China under Grant 2017YFB0802300, Grant 2017YFB0802000, and Grant 2017YFB0802003; in part by the National Natural Science Foundation of China under Grant U1633114; in part by the Sichuan Science and Technology Program under Grant 2018GZ0202; in part by the State Key Laboratory of Cryptography under Grant MMKFKT201912; and in part by the Research Grant of the Natural Sciences and Engineering Research Council of Canada. *(Corresponding author: Hongwei Li.)*

Haomiao Yang is with the School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China, and also with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON N2L 3G1, Canada (e-mail: haomyang@uestc.edu.cn).

Shaopeng Liang and Hongwei Li are with the School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China (e-mail: polarisliang@qq.com; hongweili@uestc.edu.cn).

Jianbing Ni is with the Department of Electrical and Computer Engineering, Queen's University, Kingston, ON K7L 3N6, Canada (e-mail: jianbing.ni@queensu.ca).

Xuemin (Sherman) Shen is with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON N2L 3G1, Canada (e-mail: sshen@uwaterloo.ca).

Digital Object Identifier 10.1109/JIOT.2020.2992349

classification algorithms [14], [15] have been proposed to achieve secure computation over encrypted data, by exploiting homomorphic encryption (HE) or secure multiparty computation. However, the HE, e.g., the Paillier encryption, has to encrypt each feature in data object one by one, which may bring heavy computational overhead to devices. Meanwhile, the multiparty computation needs multiple rounds of interactions between devices for computing the similarity of vectors in the multidimensional feature space such that the communication burden is heavy. Besides, some secure *k*NN classifiers have been proposed based on data disturbance techniques, e.g., *k*-anonymity [16] or differential privacy [17]. Nevertheless, data disturbance may reduce classification accuracy due to the blending of statistical noise. Especially, it cannot yet truly preserve privacy because of reidentification attacks [18]. Therefore, how to achieve efficient and secure *k*NN classification with high accuracy guarantee is still challenging. Furthermore, the existing schemes assume that the data are maintained on a single powerful server, while the industrial control system is usually distributed, which means that many distributed servers are interconnected to form a scalable control platform for the industrial control system. For example, a control server is deployed to control the operations of devices in a car manufacturing machine shop, and these distributed servers are controlled by the control center.

In this article, we propose a secure and efficient distributed *k*NN classification algorithm (SEED-*k*NN) for intelligent industrial control systems. Specifically, we first design a secure and efficient VHE scheme (SE-VHE) that can achieve semantic security and efficient data encryption, and then propose the SEED-*k*NN based on the designed SE-VHE to support efficient *k*NN classification over the encrypted training samples with high classification accuracy. Moreover, by considering the fact that the data are separately maintained on multiple servers, the Map/Reduce architecture is leveraged to achieve the parallel and distributed data classification. Specifically, the main contributions of this article are summarized as threefold.

1) The SE-VHE is designed by constructing the new key-switching matrix based on the invertible matrix in key generation and inserting the noise vector in data encryption. SE-VHE is proved to achieve semantic security. SE-VHE has the distinct features of short public-key size and low time cost on data encryption while supporting multiple homomorphic operations such as linear transformation.
2) By leveraging the designed SE-VHE, SEED-*k*NN is designed to achieve secure and accurate *k*NN classification for the intelligent systems. Integrating the Map/Reduce architecture, the encrypted query is split and mapped to all the distributed servers maintaining the training samples, and then each server computes the similarity scores, i.e., the inner products between the encrypted query and the training vectors, for majority voting. During these procedures, both the *k*NN query vectors and training samples are well protected based on the SE-VHE.
3) The security of the SE-VHE and the SEED-*k*NN is discussed to demonstrate that the former realizes semantic security and the latter can well protect the confidentiality of training data, *k*NN queries, and class labels based on the security of SE-VHE. The performance evaluation is also conducted to show that the SE-VHE has the shorter public-key size and higher efficiency on data encryption than the existing vector HE (VHE) [19], and the SEED-*k*NN achieves high classification accuracy with low computational and communication overhead.

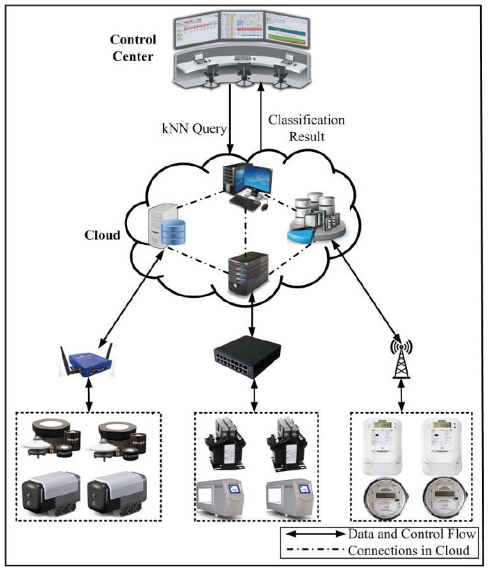


Fig. 1. System model.

The remainder of this article is organized as follows. We formalize the system model and design goals in Section II and propose the SE-VHE in Section III. Then, we propose SEED-*k*NN and discuss its security in Section IV, which followed by the performance evaluation in Section V. Finally, we review the related work in Section VI and draw the conclusion in Section VII.

II. Problem Statement

In this section, we formally define the system and threat models, and identify our design goals.

A. System and Threat Models

The intelligent industrial control systems monitor and control the manufacturing processes and system operations based on the collected data on distributed servers using artificial intelligence technologies. As shown in Fig. 1, the system model consists of three entities, namely, the control center, the cloud, and the devices.

Control Center: The control center learns the current states of the industrial control system or subsystem, monitors the manufacturing processes, and makes decisions for the system events. For example, the control center in the smart grid collects the power consumption of the customers in a particular area and adjusts the power generation to maintain the balance of power demand and supply. The control center not only manages, commands, directs, or regulates the behavior of devices using control loops but also uses ML algorithms, such as *k*NN algorithm, to achieve false detection, automatic control, or process optimization based on the information and control flow, and measurements of devices.

Cloud: The cloud is composed of many connected servers that might be located in different positions, such as machine shops or data centers. It provides a scalable platform for significant data storage and resource sharing. It can be the private cloud built by the intelligent industrial control systems or the public cloud provided by the cloud storage and computing service provider. These distributed servers are responsible for managing their individual manufacturing tasks and sharing their idle resources for supporting the desirable functionalities of intelligent industrial control systems.

Devices: The devices are the deployed sensors, measurers, operating arms, and local servers monitored by the control center in intelligent industrial control systems. These devices collect the information about the system states, update the state information, and perform the control commands. They are capable of interacting with the cloud for data submission and receiving control flows from the control center through wired or wireless communications.

The control center performs the ML algorithms, such as the *k*NN algorithm, on the data set in the cloud to discover the added values for automatic control and industrial process monitoring. The data set $\boldsymbol{D}$, generated by devices, is preprocessed to generate the training samples and maintained on the distributed servers in the cloud. The control center issues the *k*NN query $\boldsymbol{q}$ to the cloud in an attempt to acquire the corresponding classification $f(\boldsymbol{q})$ based on the similarity measurement function f. This *k*NN query $\boldsymbol{q}$ will be partitioned and distributed to the multiple servers in the cloud that may maintain different parts of training samples. The classification results from servers are gathered and the final class label $f(\boldsymbol{q})$ is returned to the control center.

However, the industrial control system is vulnerable to be interfered or corrupted, as its components, including computers, networks, applications, and devices may bring serious security weaknesses to the whole system. The standard ISA/IEC-62443 has defined procedures for implementing electronically secure industrial automation and control systems, but the cyber security accidents still frequently happened, such as power grid blackouts caused by cyberattacks in Ukraine in 2015 and 2017. The data exchanged between devices and servers might be eavesdropped or captured by the hackers, and the cloud is suspicious to expose the maintained data intentionally or unintentionally. Also, the software bugs, hardware failures, bugs in the network path, and motivated hackers may bring serious threats toward data confidentiality. The control center is supposed to be fully trusted. Strong defense mechanisms would be deployed to resist potential cyberattacks for the control center, the "brain" of intelligent industrial control systems.

B. Design Goals

To enable secure and efficient distributed *k*NN classification under the aforementioned system model and against security threats, the SEED-*k*NN should achieve the following design goals.

1) *Data Confidentiality:* The training samples in the cloud should be well protected to prevent possible data leakage. Also, the *k*NN query and the returned class label are needed to be preserved to prevent the control information exposure.
2) *Classification Accuracy:* The accuracy of *k*NN classification on the encrypted training samples should be sufficiently high to support correct automatic control and decision making for the control center. The classification accuracy of encrypted data samples should be almost the same as that of the clear data.
3) *Distributed Query:* Distributed computing should be supported such that the *k*NN query could be performed on multiple servers in the cloud in parallel.
4) *High Efficiency:* The *k*NN classification should be efficiently executed on massive data on each server, and the communication and computational overhead brought by the SEED-*k*NN should be low.

III. Secure and Efficient VHE

The original VHE [19] is not secure enough to resist the existing attacks, i.e., an attacker can recover the secret key or the plain vector from the key-switching matrix or the ciphertext, respectively [20]. Besides, in the original VHE, the elements in each vector and matrix are represented as binary strings. The operations of high-dimensional vectors are performed bit by bit, which need the large size of public keys to encrypt and result in heavy computational costs. Due to the binary representation of the vector or the matrix in the original VHE, the operations to high-dimensional vectors lead to large computational and communication costs. For example, to encrypt a 500-D vector, the size of the public key should be 120 MB and the encryption time is up to 27 s. Hence, the original VHE cannot be used in IIoT. In this article, we will design a new VHE scheme for better security and higher performance (SE-VHE), which is the cryptographic primitive of SEED-*k*NN. The SE-VHE is derived from the original VHE. Specifically, a new key-switching matrix is designed to prevent the secret key recovery based on an invertible matrix in key generation and a noise vector is used to obfuscate the plain vector in data encryption. In this way, the plain vector and the secret key cannot be recovered from the ciphertext and the key-switching matrix, respectively, and thus SE-VHE can achieve IND-CPA security, i.e., the indistinguishability under chosen-plaintext attacks. Furthermore, SE-VHE has the advantages of the small public-key size and low encryption time cost, since it does not use the binary representation of vectors or matrices.

Notations: Scalars are denoted by lowercase letters, vectors are denoted by lowercase bold letters, and matrices are denoted by uppercase bold letters, respectively (e.g., x is a scalar, $\boldsymbol{x}$ is a column vector, and $\boldsymbol{X}$ is a matrix). In addition, $|\boldsymbol{x}|$ denotes the maximum entry and $\lceil \boldsymbol{x} \rfloor_q$ denotes the nearest integer modulo q of each entry x_i for the vector $\boldsymbol{x} \in \mathbb{R}^n$.

Algorithm 1 *GenInv*

```
1: Input: $n$
2: Output: $I_1, I_2$
3: Generate two identity matrices $I_1, I_2 \in \mathbb{Z}_q^{n\times n}$
4: for $i = 1$ to $k$ do
5:     Generate randomly a vector: $[s, d, o], s, d \leftarrow \chi, s \neq d, o \leftarrow \{-1, 0, 1\}$
6:     if $o = 0$ then
7:         Exchange columns $I_1[d]$ and $I_1[s]$
8:         Exchange rows $I_2[s]$ and $I_2[d]$
9:     else if $o \neq 0$ then
10:        $I_1[s] \leftarrow I_1[s] - o \cdot I_1[d]$
11:        $I_2[d] \leftarrow I_2[d] + o \cdot I_2[s]$
12:    end if
13: end for
```

Algorithm 2 *Trans*

```
1: Input: $S_{old}$
2: Output: $S, M$
3: Generate two matrices $P_s, P_m \leftarrow GenInv(n')$
4: Generate two matrices, $T \leftarrow \chi^{m\times(n'-m)}$ and $A \leftarrow \chi^{(n'-m)\times n}$
5: Construct two matrices, $S_t = [I, T] \in \mathbb{Z}^{m\times n'}$, where $I$ is an identity matrix of $m$, and $M_t = \begin{bmatrix} S_{old} - TA \\ A \end{bmatrix} \in \mathbb{Z}_q^{n'\times n}$
6: Calculate $S = S_t P_s$ and $M = P_m M_t$
```

A. Scheme Description

We first present two important algorithms, *GenInv* and *Trans*, which are used to generate the parameters in SE-VHE.

Algorithm 1 generates two matrices I_1 and I_2 with $I_1 I_2 = I$, where n is determined by the key size. Note that the larger the value of k is, the better randomness I_1 and I_2 has.

We accordingly design a new key-switching operation *Trans*, where n and n' denote the dimensions of the old key matrix $S_{\text{old}} \in \mathbb{Z}_q^{m\times n}$ and the transformed key matrix $S \in \mathbb{Z}_q^{m\times n'}$, respectively. The modified key-switching operation is depicted in Algorithm 2.

In Algorithm 2, given a ciphertext c_{old} and its secret key S_{old}, S and M are generated. Thus, the new ciphertext can be calculated by $c = Mc_{\text{old}}$ and its correctness is guaranteed by the following equation:

$$\begin{aligned} Sc &= SMc_{\text{old}} \\ &= S_t P_s P_m M_t c_{\text{old}} \\ &= [I, T]\begin{bmatrix} S_{\text{old}} - TA \\ A \end{bmatrix} c_{\text{old}} \\ &= S_{\text{old}} c_{\text{old}}. \end{aligned}$$

Now, we present the SE-VHE that consists of four probabilistic polynomial-time algorithms: $VHE = (VHE.Setup, VHE.KG, VHE.Enc, VHE.Dec)$.

1) *VHE.Setup(λ):* Given a security parameter λ, $m, n, p, q, w \in \mathbb{Z}$ are randomly chosen with $m < n$ and $q \gg p$, and a discrete Gaussian noise distribution χ on $\mathbb{Z}_q$ with standard deviation δ is picked. Note that the modulus q is chosen based on λ to guarantee the security. The system parameter Param $= (m, n, p, q, w, \chi)$ is published to the public.
2) *VHE.KG(Param)*: Algorithm 2 is executed to generate $(S, M) \leftarrow Trans(wI)$, where $I \in \mathbb{Z}^{m\times m}$ is an identity matrix. The secret key $S \in \mathbb{Z}_q^{m\times n}$ is kept private and the encryption key $M \in \mathbb{Z}_q^{n\times m}$ is released. Note that according to Algorithm 2, we have $SM = wI$.
3) *VHE.Enc(x, M):* A small noise vector $e \leftarrow \chi^n$ is randomly chosen in $\mathbb{Z}_q^n$ and the plaintext x is encrypted as $c = Mx + e$, where $x \in \mathbb{Z}_p^m$ is a plaintext vector, $c \in \mathbb{Z}_q^n$ is a ciphertext vector, and $M \in \mathbb{Z}_q^{n\times m}$ is the encryption key. Note that to guarantee the security of plaintext, the noise e is added to generate the ciphertext.
4) *VHE.Dec(c, S):* The plaintext $x \in \mathbb{Z}_p^m$ is recovered by calculating $x = \lceil(Sc/w)\rfloor_q$, where $c \in \mathbb{Z}_q^n$ is the corresponding ciphertext and $S \in \mathbb{Z}_q^{m\times n}$ is the secret key. Note that the correctness of decryption is guaranteed by the following equation:

$$x = \left\lceil \frac{Sc}{w} \right\rfloor_q = \left\lceil \frac{S(Mx+e)}{w} \right\rfloor_q = \left\lceil x + \frac{Se}{w} \right\rfloor_q.$$

It is easy to see that to recover x from c with the secret key S correctly, this condition $|Se| < (w/2)$ should be satisfied. To be specific, we have $n|S||e| < (w/2)$. Therefore, let E denote the upper bound of $|e|$, then we have

$$E < \frac{w}{2n|S|}.$$

Compared to the original VHE of $E < w/2$ [19], the upper bound of $|e|$ in our scheme has somewhat decreased. Nevertheless, if E is properly set, it does not affect the correctness when being applied to SEED-*k*NN.

In addition, SE-VHE supports all homomorphic operations of the original VHE, including addition, linear transformation, and weighted inner product [19]. Nevertheless, the conditions for guaranteeing homomorphism would accordingly change.

B. Security

The SE-VHE security follows from the learning with error (LWE) problem [21], as shown in Theorem 1. Here, we give a sketchy proof framework.

Theorem 1: Our SE-VHE scheme can achieve semantic security assuming that the LWE problem is intractable.

Proof: First, we consider that the proposed SE-VHE scheme achieves one-way security. In the SE-VHE, the plaintext x is encrypted as $c = Mx + e$. Here, to break one-way security, given the ciphertext c, the adversary is required to answer the plaintext x. This is equivalent to solve the LWE problem: to obtain x from $c \approx Mx$ if M is a random matrix. As is known, the LWE is a well-known hard problem [21]. Therefore, as long as M is a random matrix, the one-way security of the SE-VHE can be reduced to the LWE problem. Thus, we subsequently analyze the randomness of M. As stated above, we have $M = P_m M_t$. To ensure the randomness, M_t is required to have a large size, which depends on the size of T and A. It also requires P_m to be random. To be specific, according to Algorithm 1, each row of P_m should be independent of the other $n - 1$ rows. To clarify this point, we first set

P_1 as the probability that a row in $\boldsymbol{P}_m$ has no dependence on other rows. Thus, we have

$$P_1 = \left(\frac{\binom{n-1}{2}}{\binom{n}{2}} \right)^k = \left(1 - \frac{2}{n}\right)^k.$$

■

Then, we set P_2 as the probability that a row in $\boldsymbol{P}_m$ is made only one exchange operation. Thus, we have

$$P_2 = \frac{1}{3}\left(\frac{\binom{n-1}{2}}{\binom{n}{2}} \right)^{k-1} \frac{\binom{n-1}{1}}{\binom{n}{2}} = \frac{2}{3}\left(1 - \frac{2}{n}\right)^{k-1}.$$

It is easy to see that P_1 and P_2 could be negligible as long as the loop value k in Algorithm 1 is significantly large with respect to the ciphertext size $n(n > 2)$. Therefore, our SE-VHE can have $\boldsymbol{P}_m$ and $\boldsymbol{M}_t$ with good randomness. In summary, if the loop value k is large enough, the SE-VHE achieves one-way security based on the hard problem of LWE.

Subsequently, on the basis of one-way security, we further discuss IND-CPA security (indistinguishability under the chosen-plaintext attack). In IND-CPA, the adversary has the capability of querying the encryption oracle for polynomial times, in which the oracle would return the ciphertext given any adaptively chosen plaintext. Given two plaintexts $\boldsymbol{a}$ and $\boldsymbol{b}$, and the challenging ciphertext $\boldsymbol{c}$, which is the encryption of $\boldsymbol{a}$ or $\boldsymbol{b}$, the adversary is asked to decide which plaintext is corresponding to $\boldsymbol{c}$. First, IND-CPA security cannot be achieved if the adversary can break one-way security by recovering $\boldsymbol{x}$ from $\boldsymbol{c} = \boldsymbol{Mx}+\boldsymbol{e}$. Fortunately, as discussed above, the SE-VHE can guarantee one-way security. Second, for the challenge of distinguishing $\boldsymbol{a}$ and $\boldsymbol{b}$ with respect to $\boldsymbol{c}$, it is equivalent to distinguish between $\boldsymbol{a} - \boldsymbol{a}$ and $\boldsymbol{a} - \boldsymbol{b}$ based on the homomorphic property. To be specific, it is equivalent to distinguish between the zero vector $\boldsymbol{0}$ and the nonzero vector $\boldsymbol{x}$. Let $\boldsymbol{c}_0 = \boldsymbol{M0} + \boldsymbol{e}_0$ and $\boldsymbol{c}_1 = \boldsymbol{Mx} + \boldsymbol{e}_1$, where $\boldsymbol{x} \neq \boldsymbol{0}$. Since $\boldsymbol{e}_0, \boldsymbol{e}_1 \leftarrow \chi^n$, it is indistinguishable between $\boldsymbol{c}_0$ and $\boldsymbol{c}_1$. Therefore, the SE-VHE satisfies IND-CPA. Namely, the SE-VHE achieves semantic security.

C. Performance Comparison

We conduct simulation experiments to evaluate the SE-VHE performance and compare the performance between the SE-VHE and the original VHE [19]. In order to guarantee the decryption correctness, we set $w = 2^{30}$ and $E = 200$. The experiments are carried on a laptop with i3-4130 CPU and 8-GB RAM, running Windows 10 operating system and Python 3.5 programming language. The comparison results are depicted in Fig. 2(a) and (b). It can be seen that the longer the plaintext vector $\boldsymbol{x}$ is, the more storage space the SE-VHE can save for the public-key storage and the more encryption time the SE-VHE can reduce compared with the original VHE. Below, we briefly analyze the comparison results.

In the original VHE, the key-switching matrix is constructed as $\boldsymbol{M}_o = \begin{bmatrix} (w\boldsymbol{I})^* - \boldsymbol{TA} + \boldsymbol{E} \\ \boldsymbol{A} \end{bmatrix} \in \mathbb{Z}_q^{n'l \times m}$, where $(w\boldsymbol{I})^*$ is the binary representation of $w\boldsymbol{I}$ and l is the binary parameter. In SE-VHE, a new key-switching matrix is constructed as $\boldsymbol{M}_t = \begin{bmatrix} w\boldsymbol{I} - \boldsymbol{TA} \\ \boldsymbol{A} \end{bmatrix} \in \mathbb{Z}_q^{n' \times m}$ (step 5 of Algorithm 2). The difference is that the key matrix $w\boldsymbol{I}$ is represented to $\boldsymbol{M}_o$ in binary and the dimension is $n'l \times m$, while $w\boldsymbol{I}$ is represented to $\boldsymbol{M}_t$ directly in the unit of each element and the dimension is $n' \times m$. In the original VHE, the public key $\boldsymbol{M}$ is exactly $\boldsymbol{M}_o$ and the encryption of a plain vector $\boldsymbol{x}$ under the public key $\boldsymbol{M}$ can be performed as $\boldsymbol{c} = \boldsymbol{Mx}$. In SE-VHE, the public key is further calculated as $\boldsymbol{M} = \boldsymbol{P}_m\boldsymbol{M}_t$ (step 6 of Algorithm 2) and the encryption can be calculated as $\boldsymbol{c} = \boldsymbol{Mx} + \boldsymbol{e}$. Since SE-VHE does not process the vectors or matrices in binary, the public-key size in SE-VHE is much shorter than that in the original VHE and the encryption time is significantly reduced. Therefore, SE-VHE has the advantages of the small public-key size and low encryption time cost.

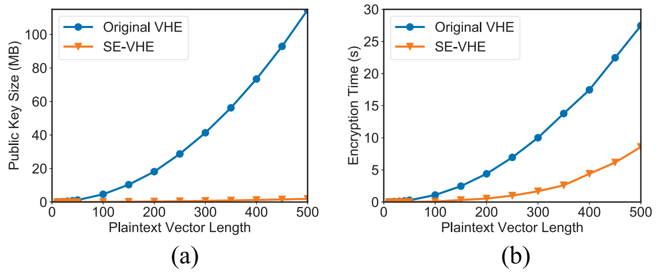


Fig. 2. VHE performance comparison. (a) Public Key Size. (b) Encryption Time.

IV. Proposed SEED-kNN

In this section, we propose SEED-kNN to achieve the kNN classification over encrypted training samples based on the SE-VHE.

A. Secure Computation of Similarity Scores

The devices upload the training data set $\boldsymbol{D}$ to the distributed servers in the cloud. $\boldsymbol{D}$ could be represented as a table comprising n data records $\boldsymbol{t}_1, \boldsymbol{t}_2, \ldots, \boldsymbol{t}_n$. In $\boldsymbol{D}$, each record has $\boldsymbol{t}_i = [t_{i1}, \ldots, t_{im}, c_i]$ with $m + 1$ attributes, $i = 1, 2, \ldots, n$. In particular, the $(m + 1)$th attribute c_i is the class label. Then, the control center has a query set, including N queries $\boldsymbol{q}_1, \boldsymbol{q}_2, \ldots, \boldsymbol{q}_N$, in which each query is $\boldsymbol{q}_j = [q_{j1}, \ldots, q_{jm}]$, $j = 1, 2, \ldots, N$.

To protect the data set, the ciphertexts $\boldsymbol{D}'$ of $\boldsymbol{D} = \{\boldsymbol{t}_1, \boldsymbol{t}_2, \ldots, \boldsymbol{t}_n\}$ are stored in a server (or multiple servers for storage replication) in the cloud. The control center performs kNN queries on the data set and acquires classification results from the cloud for industrial control systems. The N queries $\boldsymbol{q}_1, \boldsymbol{q}_2, \ldots, \boldsymbol{q}_N$ sent by the control center are also protected. Upon receiving the encrypted N queries, the master server in cloud splits the queries and maps them to the servers maintaining the data. Each distributed server calculates similarity scores between the stored data records and each query $\boldsymbol{q}_j$, to find the most similar k records of $\boldsymbol{q}_j$. Here, the similarities are evaluated by means of the inner product $\boldsymbol{q}_j{}^T\boldsymbol{t}_i$ in the kNN classification. The similarity scores computed by each server are returned to the master server and further forwarded to the control center. However, in this procedure, the main challenge is how to compute ${\boldsymbol{q}_j}^T\boldsymbol{t}_i$ from the encrypted $\boldsymbol{t}_i$ and $\boldsymbol{q}_j$. To address this issue, we use the linear-transformation operation on the encrypted $\boldsymbol{t}_i$ with the transformation matrix $\boldsymbol{G} = {\boldsymbol{q}_j}^T$ in SEED-*k*NN.

B. SEED-kNN

SEED-*k*NN is composed of two phases: 1) *initialization* and 2) *classification*.

Initialization is responsible for initializing the whole system and preparing the training data set for the *k*NN classification.

1) *Setup:* The control center invokes *Setup* to generate the system parameter *Param*, the encryption key $\boldsymbol{M}_0$, and the secret key $\boldsymbol{S}_0$. Then, the control center publishes *Param* and $\boldsymbol{M}_0$ to the public, and keeps $\boldsymbol{S}_0$ in private.
2) *DataUpload:* The device recordwise encrypts the training data set $\boldsymbol{D} = \{\boldsymbol{t}_1, \boldsymbol{t}_2, \ldots, \boldsymbol{t}_n\}$. Specifically, $\boldsymbol{t}_i[1:m]$ and $\boldsymbol{t}_i[m+1]$ (also known as the class label c_i) in the data record $\boldsymbol{t}_i$ are encrypted, respectively. That is, the device first calculates $\boldsymbol{t}'_i[1:m] \leftarrow \mathbf{VHE.Enc}(\boldsymbol{t}_i[1:m], \boldsymbol{M}_0)$ and $c'_i \leftarrow \mathbf{VHE.Enc}(c_i, \boldsymbol{M}_0)$, in which $\boldsymbol{t}'_i[1:m]$ are the ciphertexts of the first m attributes of $\boldsymbol{t}_i$, and c'_i is the ciphertext of the corresponding class label c_i. Then, the device uploads $\boldsymbol{D}' = \{(\boldsymbol{t}'_i[1:m], c'_i), 1 \le i \le n\}$ to the server in the cloud. Besides, to improve computational overhead, the device also can encrypt $\boldsymbol{t}_i$ as a whole: $\boldsymbol{t}'_i \leftarrow \mathbf{VHE.Enc}(\boldsymbol{t}_i, \boldsymbol{M}_0)$. In this way, an encryption operation can be saved for the device.

Classification utilizes the linear transformation operation to evaluate the similarity scores based on the inner product using the encrypted $\boldsymbol{t}'_i$.

1) *QueryGen:* For a single query vector $\boldsymbol{q} = [q_1, q_2, \ldots, q_m]$, there is a corresponding transformation matrix as

$$\boldsymbol{G} = \begin{bmatrix} q_1 & q_2 & \cdots & q_m & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{1}$$

and the control center needs to calculate the similarity score between the query and the data record. If there are N queries $\boldsymbol{q}_1, \boldsymbol{q}_2, \ldots, \boldsymbol{q}_N$, the control center has to calculate the similarity scores for these queries one by one. To reduce the operation times, these N queries can be handled at the same time by using the batch technique. Specifically, the transformation matrix corresponding to N queries can be constructed as

$$\boldsymbol{G} = \begin{bmatrix} q_{11} & \cdots & q_{1m} & 0 \\ \vdots & \ddots & \vdots & \vdots \\ q_{N1} & \cdots & q_{Nm} & 0 \\ 0 & \cdots & 0 & 1 \end{bmatrix}. \tag{2}$$

In this way, the control center can perform N encrypted queries to obtain the similarity scores simultaneously.
2) *ScoreCalculate:* Upon receiving the encrypted query, i.e., the key-switching matrix $\boldsymbol{M}$, from the control center, the master server in the cloud distributes $\boldsymbol{M}$ to all servers maintaining the data set. After receiving $\boldsymbol{M}$, each server calculates the similarity score $s'_i \leftarrow \boldsymbol{M}\boldsymbol{t}'_i$ for each $\boldsymbol{t}'_i$ in $\boldsymbol{D}'$ for $1 \le i \le n$, and sends $\boldsymbol{N}' = \{s'_1, \ldots, s'_n\}$ to the master server. As $\boldsymbol{t}'_1, \boldsymbol{t}'_2, \ldots, \boldsymbol{t}'_n$ are independent, multiple servers can calculate the encrypted inner products s'_i in parallel. Here, the similarity scores are the inner products of the query vector and the data set, and each s'_i is the ciphertext of a similarity score. The master server finally collects all the encrypted similarity scores returned from the distributed servers and forward them to the control center. In order to process massive training data in practice, we leverage the parallel framework of Map/Reduce [22] to calculate the matrix multiplication as $\boldsymbol{N}' \leftarrow \boldsymbol{M} \times \boldsymbol{D}'$, where $\boldsymbol{M}$ is the key-switching matrix corresponding to the user's query, $\boldsymbol{D}' = \{\boldsymbol{t}'_1, \boldsymbol{t}'_2, \ldots, \boldsymbol{t}'_n\}$ is the encrypted training data set, and $\boldsymbol{N}' = \{s'_1, \ldots, s'_n\}$ is the encrypted similarity scores. For the matrix multiplication upon Map/Reduce, we may consider the basic technique of element by element, in which the mapper independently calculates the multiplication for each pair of elements, while the reducer combines the results for each output element. However, this element-by-element technique incurs heavy computational and communication overhead. Since $\boldsymbol{t}'_1, \boldsymbol{t}'_2, \ldots, \boldsymbol{t}'_n$ are independent and the distributed servers use the same key-switching matrix $\boldsymbol{M}$ to calculate the encrypted similarity scores, we use the blocking technique of row by column to improve the efficiency of the matrix multiplication, in which the first matrix $\boldsymbol{M}$ can be decomposed into row vectors and the second matrix $\boldsymbol{D}'$ can be decomposed into column vectors.
3) *LabelDec:* The control center decrypts $\boldsymbol{N}'$ to $\boldsymbol{N} = \{\boldsymbol{G}\boldsymbol{t}_i | i = 1, \ldots, n\}$ with the new secret key $\boldsymbol{S}'$ to obtain $\boldsymbol{G}\boldsymbol{t}_i \leftarrow \mathbf{VHE.Dec}(s'_i, \boldsymbol{S}')$, as shown in (3), in which $\boldsymbol{q}^T\boldsymbol{t}_i[1:m]$ is the similarity score and c_i is the class label

$$\boldsymbol{G}\boldsymbol{t}_i = \begin{bmatrix} \boldsymbol{q}^T\boldsymbol{t}_i[1:m] \\ c_i \end{bmatrix}. \tag{3}$$

For the batch computation of N queries, the control server decrypts to obtain $\boldsymbol{G}\boldsymbol{t}_i$ with the new secret key $\boldsymbol{S}'$, as shown in (4), in which ${\boldsymbol{q}_j}^T\boldsymbol{t}_i[1:m]$ are the similarity scores and c_i is the class label

$$\boldsymbol{G}\boldsymbol{t}_i = \begin{bmatrix} {\boldsymbol{q}_1}^T\boldsymbol{t}_i[1:m] \\ {\boldsymbol{q}_2}^T\boldsymbol{t}_i[1:m] \\ \cdots \\ {\boldsymbol{q}_N}^T\boldsymbol{t}_i[1:m] \\ c_i \end{bmatrix}. \tag{4}$$

4) *MajorityVote:* The control center counts the majority class from the votes of k-nearest neighbors and finally acquires the class label $c_{\boldsymbol{q}_j}$ for each query $\boldsymbol{q}_j$.

C. Security Remarks

SEED-*k*NN protects the confidentiality of the training data set $\boldsymbol{D}$, the query vector $\boldsymbol{q}$, and the corresponding classification result $c_{\boldsymbol{q}}$. Specifically, the training data set $\boldsymbol{D}$ is encrypted by the SE-VHE, whose IND-CPA security can be reduced to the LWE problem. Without $\boldsymbol{S}_0$, no adversary can acquire any information about $\boldsymbol{D}$. The query vector $\boldsymbol{q}$ is transformed in Algorithm 2 to generate $\boldsymbol{M}$, which can be also considered as the encryption of $\boldsymbol{G}$ under the secret keys $\boldsymbol{S}_0$ and $\boldsymbol{S}'$. With the transformation in Algorithm 2, no information of $\boldsymbol{G}$ can be extracted from $\boldsymbol{M}$, except for the degree of $\boldsymbol{G}$. According to the homomorphic operation of the SE-VHE, the class label $c_{\boldsymbol{q}}$ is also encrypted by the SE-VHE and only the control center is capable of recovering $\boldsymbol{G}t_i$ with the new secret key $\boldsymbol{S}'$. Since the SE-VHE is IND-CPA secure, the confidentiality of class label $c_{\boldsymbol{q}}$ is based on the LWE problem. In summary, as long as $\boldsymbol{S}_0$ and $\boldsymbol{S}'$ are kept privately by the control center, and the LWE problem is intractable, no adversary can acquire any knowledge about $\boldsymbol{D}$, $\boldsymbol{q}$, or $c_{\boldsymbol{q}}$ in SEED-*k*NN.

V. Performance Evaluation

In the section, we evaluate the performance of the SEED-*k*NN in terms of classification accuracy, computational overhead, and communication cost. The extensive experiments are conducted on real data sets in UCI [23]. The computing tasks of the devices and the control center in SEED-*k*NN are conducted on a laptop with i3-4130 CPU and 8-GB memory, running Windows 10, and the computations of the cloud are executed on a server with E5-2430 CPU and 24-GB memory, running Ubuntu 16.04.

A. Classification Accuracy

The SE-VHE can support the encryption and decryption over integers. In order to achieve *k*NN classification over real numbers, we use rational numbers to approximate real numbers in SEED-*k*NN. That is, a real number is first set as the desired precision times a scaling factor and then rounded to the nearest integer. In specific, we use the *Z*-score data standardization method [24] to normalize the data to be classified.

To demonstrate the effectiveness of the SEED-*k*NN, both the traditional *k*NN algorithm and our proposed SEED-*k*NN are executed on the Breast Cancer Wisconsin data set [23] with the following setting. The traditional *k*NN algorithm is run directly over real numbers in the Breast Cancer Wisconsin data set. To execute SEED-*k*NN, we set the precision of the data as 1, 2, and 3 digits, which means that real numbers are, respectively, scaled by 10 times, 100 times, and 1000 times and then rounded to integers. The classification accuracy of both *k*NN and SEED-*k*NN is shown in Table I with respect to the different digits. If data precision is 2 or 3 digits, SEED-*k*NN can obtain the same classification accuracy with the traditional *k*NN algorithm, i.e., 98%, while preserving the privacy of data owners. If data precision is 1 digit, the accuracy of SEED-*k*NN declines to 57%. Only in this case, SEED-*k*NN is not as accurate as the traditional *k*NN, so we do not recommend to set the precision to be 1 digit. In the following experiments, we set the precision to be 2 digits.

Then, we run the traditional *k*NN algorithm and our proposed SEED-*k*NN on multiple data sets in UCI [23]. As shown in Table II, our SEED-*k*NN can achieve almost the same classification accuracy as the traditional *k*NN algorithm. Here, n is the number of data records in training data, N is the number of *k*NN queries, and m is the number of attributes in each training sample. Note that although SEED-*k*NN does not achieve higher classification accuracy than the traditional *k*NN algorithm, it possesses the important merit of privacy preservation. The original *k*NN cannot offer this property while it is imperative in IIoT since the privacy preservation for users is critical. The training data set is extracted from data flows, system states, or measurements of various sensors in IIoT. These data contain plenty of important information that manufacturers or corporations are unwilling to expose to the public, such as industrial process flows, product information, business secrecy, and traceable data of individuals. Therefore, it is necessary to propose SEED-*k*NN to protect the private information in IIoT.

TABLE I
Accuracy of *k*NN and SEED-*k*NN

Precision	**Accuracy of *k*NN**	**Accuracy of SEED-*k*NN**
3	0.98	0.98
2	0.98	0.98
1	0.98	0.57

TABLE II
Accuracy on Multiple Data Sets

Dataset	n	N	m	*k*NN	SEED-*k*NN
banknote	1079	275	4	0.98	0.95
creadit	522	131	15	0.87	0.86
vote	185	47	16	0.89	0.89
spambase	3680	921	57	0.88	0.87
letter	1244	311	16	0.99	0.99
mfeat-zernike	320	80	47	0.82	0.82
nursery	6868	1718	8	0.99	0.99
page-blocks	162	41	10	0.95	0.97
pendigits	1828	458	16	0.99	0.99
satimage	1787	447	36	0.99	0.99
splice	1228	307	60	0.91	0.91
waveform	2646	662	40	0.88	0.88
wine	104	26	13	0.95	0.95

B. Computation and Communication Overhead

In *initialization*, the control center executes *Setup* once to generate the system parameters such that the executing time is constant. *Classification* and *MajorityVote* can be efficiently performed since they are run in the plaintext domain. Therefore, we mainly discuss the performance in *DataUpload*, *QueryGen*, and *ScoreCalculate* phases.

In *DataUpload*, each data record $\boldsymbol{t}_i (1 \leq i \leq n)$ in the training data set $\boldsymbol{D}$ is encrypted before uploading. For the privacy-preserving *k*NN protocol in [14], each attribute in $\boldsymbol{t}_i$ would be encrypted using the Paillier encryption separately. As a consequence, the time complexity of the data encryption is $\mathcal{O}(nm)$. While in SEED-*k*NN, we employ the SE-VHE to encrypt the data record in a batch manner, and thus the complexity is $\mathcal{O}(n)$.

In *QueryGen*, the generation of secure query request $\boldsymbol{q}'$ is more efficient than that in [14]. Samanthula *et al.* [14] attributewise encrypted the query vector $\boldsymbol{q}$ into $\boldsymbol{q}'$ such that the running time is linear with the number of attributes; in our

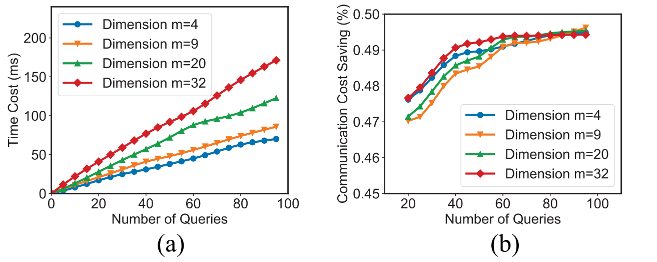


Fig. 3. Performance in SEED-*k*NN. (a) Time Cost. (b) Communication Overhead.

TABLE III
RUNNING TIME ON SIMILARITY SCORE

Number of queries	SEED-*k*NN (ms)	Ref. [14] (ms)
10	4	1006
30	11	3021
70	27	7075
100	39	10105

SEED-*k*NN, the linear-transforming operation is constructed to generate an encrypted query $\boldsymbol{q}'$ (a.k.a $\boldsymbol{M}$), which can transform $\boldsymbol{q}^T\boldsymbol{S}_0$ to $\boldsymbol{S}'$. The batch computation technique is leveraged to handle all queries at the same time, thereby the performance of SEED-*k*NN is significantly improved. Therefore, our SEED-*k*NN is more efficient than the scheme in [14]. In addition, the time cost for generating $\boldsymbol{M}$ is increasing with the number of queries, as shown in Fig. 3(a). The communication overhead in *QueryGen* is slowly increased with the number of queries, as illustrated in Fig. 3(b). In particular, if the number of queries is more than 100, the SEED-*k*NN can reduce the communication overhead by up to 50%.

In *ScoreCalculate*, the SEED-*k*NN calculates the similarity score based on the SE-VHE with efficient linear-transforming operation, and Samanthula *et al.* [14] generated the similarity score based on the secure two-party computation. As shown in Table III, the time cost of similarity score calculation in SEED-*k*NN is much lower than that in [14]. The reduction of time consumption becomes larger with the increase in the number of queries.

VI. Related Work

ML algorithms are being increasingly utilized for a variety of applications that process individuals' private data [25]–[27]. Especially, the *k*-nearest neighbors classifier is one of the essential ML algorithms that have been widely used to achieve data search, sample classification, and object recommendation. Many privacy-preserving *k*NN-based search schemes [28]–[31] have been proposed to support encrypted data search in the cloud. Nevertheless, privacy-preserving *k*NN-based search schemes cannot be directly used in data classification. To achieve secure data classification, some privacy-preserving *k*NN classifiers have been designed based on data perturbation [16], [32], [33] or data distribution [34], [35]. Aggarwal and Philip [16] proposed a new framework for privacy-preserving data mining of multidimensional data. The *k*NN classifiers with different security levels were designed on the anonymized data set mapped from the original data. Chen and Liu [32] proposed a random rotation perturbation approach for privacy-preserving data classification. Gursoy *et al.* [33] designed a differentially private *k*NN classification protocol on the basis of the radius neighbor classifier. However, data perturbation may lose valuable information and thus decrease *k*NN classification accuracy after the data are perturbed. Furthermore, the data perturbation cannot yet truly preserve privacy due to reidentification attacks [18]. Data distribution based on multiparty computation, e.g., horizontally or vertically partitioned data sets, can also protect the data confidentiality in *k*NN classification. Zhan *et al.* [34] designed a *k*NN classifier for vertically partitioned data, and Xiong *et al.* [35] presented a privacy-preserving *k*NN framework for mining horizontally partitioned data. Unfortunately, these works either sacrifice the classification accuracy or cause heavy computational and communication overhead to the participants.

With the rapid development of homomorphic encryption (HE), a large number of privacy-preserving *k*NN classification schemes have been proposed based on HE. Samanthula *et al.* [14] designed a privacy-preserving *k*NN classification protocol based on the Paillier encryption. Subsequently, still by using the Paillier encryption, Rong *et al.* [36] explored privacy preserving the *k*NN computation in multiple cloud environments. However, the Paillier encryption only can achieve the linear operation for the *k*NN algorithm. Li *et al.* [15] also presented a privacy-preserving *k*NN classification scheme with multiple data owners, but their scheme could not hide the data access pattern. Subsequently, Wu *et al.* [37] studied the privacy-preserving *k*NN classification over hybrid encrypted outsourced data, where the data attributes are encrypted using the Paillier encryption and the class labels are encrypted based on the ElGamal encryption. The reencryption technique has to be utilized to generate the intermediate computation values, which results in the heavy computational overhead. Recently, Yang *et al.* [38] proposed a privacy-preserving *k*NN classification protocol by exploiting a new VHE [19] that encrypts a data item with multiple attributes in a batch way. Unfortunately, this VHE suffers from security and performance weaknesses [20].

VII. Conclusion

In this article, we have proposed a SEED-*k*NN for intelligent industrial control systems. Specifically, we have designed a new VHE scheme that satisfies semantic security and has high efficiency on public-key storage and vector encryption. By leveraging the designed VHE, SEED-*k*NN has been proposed to efficiently classify the large-scale encrypted data on distributed servers based on the similarity scores. The training data set, *k*NN queries, and class labels are well protected to prevent information leakage. The proposed algorithm can be implemented in the intelligent industrial control systems to support a variety of applications, such as defective product identification, fault detection and classification, and anomaly detection. For the future work, we will design privacy-preserving deep learning schemes to secure control flow and enable more complex functions in intelligent industrial control systems.

References

[1] J. Lin and R.-J. Lian, "Intelligent control of active suspension systems," *IEEE Trans. Ind. Electron.*, vol. 58, no. 2, pp. 618–628, Feb. 2011.

[2] W. Kohn, Z. B. Zabinsky, and A. Nerode, "A micro-grid distributed intelligent control and management system," *IEEE Trans. Smart Grid*, vol. 6, no. 6, pp. 2964–2974, Nov. 2015.

[3] T. F. Megahed, S. M. Abdelkader, and A. Zakaria, "Energy management in zero-energy building using neural network predictive control," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 5336–5344, Jun. 2019.

[4] T. Wang, H. Gao, and J. Qiu, "A combined adaptive neural network and nonlinear model predictive control for multirate networked industrial process control," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 2, pp. 416–425, Feb. 2016.

[5] H. Hu, B. Tang, X. Gong, W. Wei, and H. Wang, "Intelligent fault diagnosis of the high-speed train with big data based on deep neural networks," *IEEE Trans. Ind. Informat.*, vol. 13, no. 4, pp. 2106–2116, Aug. 2017.

[6] N. Lu, N. Cheng, N. Zhang, X. Shen, and J. W. Mark, "Connected vehicles: Solutions and challenges," *IEEE Internet Things J.*, vol. 1, no. 4, pp. 289–299, Aug. 2014.

[7] B. Xiao, S. Yin, and H. Gao, "Reconfigurable tolerant control of uncertain mechanical systems with actuator faults: A sliding mode observer-based approach," *IEEE Trans. Control Syst. Technol.*, vol. 26, no. 4, pp. 1249–1258, Jul. 2018.

[8] Z. Zhou, C. Wen, and C. Yang, "Fault isolation based on k-nearest neighbor rule for industrial processes," *IEEE Trans. Ind. Electron.*, vol. 63, no. 4, pp. 2578–2586, Apr. 2016.

[9] A. Dairi, F. Harrou, Y. Sun, and M. Senouci, "Obstacle detection for intelligent transportation systems using deep stacked autoencoder and *k*-nearest neighbor scheme," *IEEE Sensors J.*, vol. 18, no. 12, pp. 5122–5132, Jun. 2018.

[10] O. B. Sezer, E. Dogdu, and A. M. Ozbayoglu, "Context-aware computing, learning, and big data in Internet of Things: A survey," *IEEE Internet Things J.*, vol. 5, no. 1, pp. 1–27, Feb. 2018.

[11] J. Ni, K. Zhang, Y. Yu, X. Lin, and X. Shen, "Providing task allocation and secure deduplication for mobile crowdsensing via fog computing," *IEEE Trans. Depend. Secure Comput.*, vol. 17, no. 3, pp. 581–594, Jun. 2020, doi: 10.1109/TDSC.2018.2791432.

[12] H. Li, D. Liu, Y. Dai, T. H. Luan, and X. Shen, "Enabling efficient multi-keyword ranked search over encrypted mobile cloud data through blind storage," *IEEE Trans. Emerg. Topics Comput.*, vol. 3, no. 1, pp. 127–138, Mar. 2015.

[13] Y. Zhang, C. Xu, H. Li, K. Yang, J. Zhou, and X. Lin, "HealthDep: An efficient and secure deduplication scheme for cloud-assisted ehealth systems," *IEEE Trans. Ind. Informat.*, vol. 14, no. 9, pp. 4101–4112, Sep. 2018.

[14] B. K. Samanthula, Y. Elmehdwi, and W. Jiang, "k-nearest neighbor classification over semantically secure encrypted relational data," *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 5, pp. 1261–1273, May 2015.

[15] F. Li, R. Shin, and V. Paxson, "Exploring privacy preservation in outsourced k-nearest neighbors with multiple data owners," in *Proc. ACM Workshop Cloud Comput. Security Workshop (CCSW)*, 2015, pp. 53–64.

[16] C. C. Aggarwal and S. Y. Philip, "A condensation approach to privacy preserving data mining," in *Proc. Int. Conf. Extending Database Technol. (EDBT)*, 2004, pp. 183–199.

[17] W. Huang, S. Zhou, Y. Liao, and H. Chen, "An efficient differential privacy logistic classification mechanism," *IEEE Internet Things J.*, vol. 6, no. 6, pp. 10620–10626, Dec. 2019.

[18] Y. Chen, X. Zhu, and S. Gong, "Person re-identification by deep learning multi-scale representations," in *Proc. Int. Conf. Comput. Vis. Workshops (ICCVW)*, Venice, Italy, 2017, pp. 2590–2600.

[19] H. Zhou and G. Wornell, "Efficient homomorphic encryption on integer vectors and its applications," in *Proc. Inf. Theory Appl. Workshop (ITA)*, San Diego, CA, USA, 2014, pp. 1–9.

[20] S. Bogos, J. Gaspoz, and S. Vaudenay, "Cryptanalysis of a homomorphic encryption scheme," *Cryptography Commun.*, vol. 10, no. 1, pp. 27–39, 2018.

[21] O. Regev, "On lattices, learning with errors, random linear codes, and cryptography," *J. ACM*, vol. 56, no. 6, pp. 1–40, 2009.

[22] J. Dean and S. Ghemawat, "MapReduce: Simplified data processing on large clusters," *Commun. ACM*, vol. 51, no. 1, pp. 107–113, 2008.

[23] D. Dheeru and E. Karra Taniskidou. (2017). *UCI Machine Learning Repository*. [Online]. Available: http://archive.ics.uci.edu/ml

[24] S. Patro and K. K. Sahu, "Normalization: A preprocessing stage," 2015. [Online]. Available: arXiv:1503.06462.

[25] M. I. Jordan and T. M. Mitchell, "Machine learning: Trends, perspectives, and prospects," *Science*, vol. 349, no. 6245, pp. 255–260, 2015.

[26] J. Ni, X. Lin, and X. Shen, "Toward privacy-preserving valet parking in autonomous driving era," *IEEE Trans. Veh. Technol.*, vol. 68, no. 3, pp. 2893–2905, Mar. 2019.

[27] L. Zhu, M. Li, Z. Zhang, and Z. Qin, "ASAP: An anonymous smart-parking and payment scheme in vehicular networks," *IEEE Trans. Depend. Secure Comput.*, early access, Jun. 26, 2018, doi: 10.1109/TDSC.2018.2850780.

[28] H. Li, Y. Yang, T. H. Luan, X. Liang, L. Zhou, and X. Shen, "Enabling fine-grained multi-keyword search supporting classified sub-dictionaries over encrypted cloud data," *IEEE Trans. Depend. Secure Comput.*, vol. 13, no. 3, pp. 312–325, Jun. 2016.

[29] B. Wang, Y. Hou, and M. Li, "Practical and secure nearest neighbor search on encrypted large-scale data," in *Proc. INFOCOM 35th Annu. IEEE Int. Conf. Comput. Commun.*, San Francisco, CA, USA, 2016, pp. 1–9.

[30] Y. Elmehdwi, B. K. Samanthula, and W. Jiang, "Secure k-nearest neighbor query over encrypted data in outsourced environments," in *Proc. 30th Int. Conf. Data Eng. (ICDE)*, Chicago, IL, USA, 2014, pp. 664–675.

[31] H. Li, D. Liu, Y. Dai, T. Luan, and S. Yu, "Personalized search over encrypted data with efficient and secure updates in mobile clouds," *IEEE Trans. Emerg. Topics Comput.*, vol. 6, no. 1, pp. 97–109, Jan.–Mar. 2018.

[32] K. Chen and L. Liu, "A random rotation perturbation approach to privacy preserving data classification," Coll. Comput., Georgia Inst. Technol., Atlanta, GA, USA, Rep. GIT-CC-05-12, 2005.

[33] M. E. Gursoy, A. Inan, M. E. Nergiz, and Y. Saygin, "Differentially private nearest neighbor classification," *Data Mining Knowl. Discover.*, vol. 31, no. 5, pp. 1544–1575, 2017.

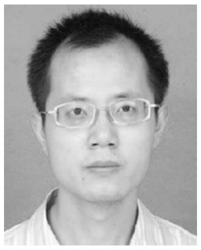
[34] J. Z. Zhan, L. Chang, and S. Matwin, "Privacy preserving *k*-nearest neighbor classification," *Int. J. Netw. Security*, vol. 1, no. 1, pp. 46–51, 2005.

[35] L. Xiong, S. Chitti, and L. Liu, "k nearest neighbor classification across multiple private databases," in *Proc. 15th ACM Int. Conf. Inf. Knowl. Manag. (CIKM)*, 2006, pp. 840–841.

[36] H. Rong, H. Wang, J. Liu, and M. Xian, "Privacy-preserving k-nearest neighbor computation in multiple cloud environments," *IEEE Access*, vol. 4, pp. 9589–9603, 2016.

[37] W. Wu, J. Liu, H. Rong, H. Wang, and M. Xian, "Efficient k-nearest neighbor classification over semantically secure hybrid encrypted cloud database," *IEEE Access*, vol. 6, pp. 41771–41784, 2018.

[38] H. Yang, W. He, J. Li, and H. Li, "Efficient and secure kNN classification over encrypted data using vector homomorphic encryption," in *Proc. Int. Conf. Commun. (ICC)*, Kansas City, MO, USA, 2018, pp. 1–7.



Haomiao Yang (Member, IEEE) received the M.S. and Ph.D. degrees in computer applied technology from the University of Electronic Science and Technology of China (UESTC), Chengdu, China, in 2004 and 2008, respectively.

He has worked as a Postdoctoral Fellow with the Research Center of Information Cross over Security, Kyungil University, Gyeongsan, South Korea, for one year until June 2013. He is currently an Associate Professor with the School of Computer Science and Engineering and Center for Cyber Security, UESTC. His research interests include cryptography, cloud security, and the cyber security for aviation communication.

Shaopeng Liang received the B.S. degree in information security from the University of Electronic Science and Technology of China, Chengdu, China, where he is currently pursuing the M.S. degree in cyberspace security.

His current research interests include big data security and artificial intelligence security.

Jianbing Ni (Member, IEEE) received the Ph.D. degree in electrical and computer engineering from the University of Waterloo, Waterloo, ON, Canada, in 2018.

He is currently an Assistant Professor with the Department of Electrical and Computer Engineering, Queen's University, Kingston, ON, Canada. His research interests are applied cryptography and network security, with current focus on cloud computing, smart grid, mobile crowdsensing, and Internet of Things.

Hongwei Li (Senior Member, IEEE) received the Ph.D. degree from the University of Electronic Science and Technology of China, Chengdu, China, in June 2008.

He is currently the Head and a Professor with the Department of Information Security, School of Computer Science and Engineering, University of Electronic Science and Technology of China. He worked as a Postdoctoral Fellow with the University of Waterloo, Waterloo, ON, Canada, from October 2011 to October 2012. His research interests include network security and applied cryptography.

Prof. Li is the Distinguished Lecturer of the IEEE Vehicular Technology Society.

Xuemin (Sherman) Shen (Fellow, IEEE) received the 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, Waterloo, ON, Canada. His research focuses on network resource management, wireless network security, social networks, 5G and beyond, and vehicular *ad hoc* and sensor networks.

Dr. Shen received the R.A. Fessenden Award in 2019 from IEEE, Canada, the James Evans Avant Garde Award in 2018 from the IEEE Vehicular Technology Society, and the 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 five times from the University of Waterloo and the Premier's Research Excellence Award in 2003 from the Province of Ontario, Canada. He served as the Technical Program Committee Chair/Co-Chair for IEEE Globecom'16, IEEE Infocom'14, IEEE VTC'10 Fall, and IEEE Globecom'07, the Symposia Chair for IEEE ICC'10, the Tutorial Chair for IEEE VTC'11 Spring, and the Chair for the IEEE Communications Society Technical Committee on Wireless Communications. He is the Editor-in-Chief of the IEEE INTERNET OF THINGS JOURNAL and the Vice President on Publications of the IEEE Communications Society. 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.