Received March 21, 2020, accepted March 28, 2020, date of publication April 7, 2020, date of current version April 23, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2986245

A Privacy-Preserving and Efficient *k*-Nearest Neighbor Query and Classification Scheme Based on k-Dimensional Tree for Outsourced Data

JIANGYI DU[1,2], **AND FULING BIAN**[1]

[1]State Key Laboratory of Information Engineering in Surveying Mapping and Remote Sensing, Wuhan University, Wuhan 430079, China
[2]School of Computer Science, Hubei University of Technology, Wuhan 430068, China

Corresponding author: Jiangyi Du (jydu_0504@hotmail.com)

ABSTRACT Cloud computing technology has attracted the attention of researchers and organizations due to its computing power, computing efficiency and flexibility. Using cloud computing technology to analysis outsourced data has become a new data utilization model. However, due to the severe security risks that appear in cloud computing, most organizations now encrypt data before outsourcing data. Therefore, in recent years, many new works on the k-Nearest Neighbor (denoted by k-NN) algorithm for encrypted data has appeared. However, two main problems are existing in the current research: either the program is not secure enough or inefficient. In this paper, based on the existing problems, we design a non-interactive privacy-preserving k-NN query and classification scheme. Our proposed scheme uses two existing encryption schemes: Order Preserving Encryption and the Paillier cryptosystem, to preserve the confidentiality of encrypted outsourced data, data access patterns, and the query record, and utilizes the encrypted the k-dimensional tree (denoted by kd-tree) to optimize the traditional k-NN algorithm. Our proposed scheme strives to achieve high query efficiency while ensuring data security. Extensive experimental results prove that this scheme is very close to the scheme using plaintext data and the existing non-interactive encrypted data query scheme in terms of classification accuracy. The query runtime of our scheme is superior to the existing non-interactive k-NN query scheme.

INDEX TERMS Privacy preservation, k-nearest neighbors, k-dimensional tree, outsourced data.

I. INTRODUCTION

Nowadays, machine learning and cloud computing have been widely used. Machine learning can mine hidden knowledge or patterns from massive data and is one of the most attractive technologies. The k-Nearest Neighbor (denoted by k-NN) algorithm is one of the classic machine learning algorithms, which can find the nearest k points from a large-scale data set based on a test object. It has been used in many studies, such as pattern recognition, Location-Based Services [1], DNA sequencing, online recommendation systems, and data analysis, etc.

As emerging information technology, cloud computing has unique advantages in computing power, computing efficiency, and flexibility. It plays important roles in many fields such as machine learning, data search, analysis and processing, medical services and e-commerce [2], attracting a large number of researchers.

Although cloud computing offers great advantages, it raises concerns about the privacy and security of the cloud. How to achieve the privacy preservation of all data in the cloud has become a new research hotspot. Many cloud-based privacy-preserving schemes have been proposed [3]–[5], which effectively solve cloud privacy and security problems in applications.

Especially with the increasing popularization of machine learning technology in multiple application scenarios, how to achieve the confidentiality of all data while using massive data to perform machine learning tasks in the cloud has also attracted widespread attention.

In particular, when users are accustomed to the cloud that can perform machine learning tasks on their own data,

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

TABLE 1. Samples of heart disease data set.

id	age	sex	cp	trestbps [mmHg]	chol[mg/dl]	fbs	slope	num
t_1	56	1	3	135	230	1	2	0
t_2	54	0	4	138	226	0	1	0
t_3	60	1	2	128	200	1	3	1
t_4	58	0	3	140	201	0	1	0
t_5	48	1	4	145	206	0	2	1

they usually encrypt all data (especially privacy information). However, with encrypted data outsourced to the cloud, it is no longer easy to perform any machine learning task efficiently and securely.

Example 1: Suppose the hospital outsources its encrypted data and relevant machine learning tasks to a cloud. If a doctor wants to know the risk level of a new patient's heart disease, he/she can use a classification method, such as a k-NN query, to confirm it. At first, the doctor generates data record q for the patient that contains the patient's specific privacy information, such as age, gender, cp (the type of chest pain), trestbps (resting blood pressure), *et al.*, as shown in Table 1. Then, this q can be sent to the cloud, which can calculate the class label of q to determine the risk level. However, since q contains the privacy information, it should be encrypted to preserve patient privacy before being sent to the cloud. Some samples of the heart disease data set are shown in Table 1 [6].

In **Example1**, a patient must provide medical information to the cloud to get medical services. The problem of exposing sensitive data has become one of the main obstacles to the broader use of cloud computing. Therefore, how to preserve users' sensitive information more securely has become a crucial research topic. Moreover, existing studies have found that under the current technical means, even if the data has been encrypted, the cloud can still obtain the sensitive information related to the original database through some means, such as observing the data access pattern.

Hence, machine learning over encrypted outsourced data should also consider the confidentiality of data access patterns while focusing on the security of encrypted data and authorized users' query records [7].

Some existing methods can not only achieve more accurate results in an honest and curious environment, but also meet the three requirements above, but the performance needs to be improved [7]. Therefore, we design a scheme for the efficiency of the analysis and processing of encrypted outsourced data. Specifically, we study the key issues in machine learning: classification. There are many typical classification algorithms, and the k-NN algorithm is one of them, which is usually learned by comparing a given training sample with similar training samples.

In this paper we consider the existing problems. We devise a non-interactive and efficient privacy-preserving k-NN query and classification scheme. Our contributions in this work are summarized as below:

- First, we achieve privacy preservation. We utilize two encryption schemes to encrypt all outsourced data and users' query records. In the process of algorithm performing, the cloud services always operation in ciphertext format to hides the data access pattern.
- Second, the new scheme is non-interactive. The data owner has no interaction with the cloud after he/she uploads the encrypted data. Similarly, when a query user submits his/her query record, he/she will not have other interactions with the cloud server, except to obtain classification results from the cloud server. We design it to reduce the interaction between the user, including the data owner and authorized users and the cloud server, and also reduces the potential security risks of data in the communication process.
- Third, we implement an encrypted kd-tree to reduce the runtime of the whole query and hide data access patterns. Through extensive experiments, we assess the accuracy and computational efficiency of our new scheme. The results give evidence that our scheme can accomplish efficient queries on the k-NN algorithm.

The rest of the paper is organized as follows. First of all, we outline some related work in Section II. Then we briefly introduce the k-NN classifier, kd-tree, order-preserving encryption (denoted by OPE), and the Paillier cryptosystem as preliminary information in Section III "Preliminaries." In Section IV, we present two system models of this scheme: the system model and the security model, then describe the design goals of this scheme, followed by an explanation of the proposed k-NN scheme and its security analysis in Section V. In Section VI, we assess the performance of our new scheme. In the end, we conclude and introduce future works in Section VII.

II. RELATED WORK

This section will outline the relevant works of the privacy-preserving k-NN algorithm.

In 2000, some scientists first proposed the concept of privacy preservation in data mining [8], [9]. Since then, a lot of privacy-preserving machine learning schemes, including classification schemes, were proposed [10], [11].

The k-NN search is one of the most familiar solutions in many applications, such as ranked key-words search [12], [13], classification [7], [11], [14], recommender system [15], [16], intrusion detection [17], [18], location privacy protection [19]. In these works, many new schemes are designed to implement the privacy-preserving k-NN algorithm [20], [21], some of which utilize data encryption techniques and some of which utilize Secure Multiparty

Computation. In recent years, many studies on the security of the k-NN search over the encrypted outsourced data in the cloud have been carried out and deepened. An efficient Asymmetric Scalar-Product-Preserving Encryption (ASPE) scheme [22] is proposed, which can be used in the k-NN query processing because it can achieve the distance comparison between the encrypted data. However, there is still room for improvement in the security of this scheme. It is not secure under the chosen-plaintext attacks, and it cannot preserve the data access pattern from being leaked to the cloud [14]. Zheng *et al.* [23] utilized kd-tree, AES, and a homomorphic encryption technique to design a k-NN scheme for privacy preservation for outsourced eHealthcare data. This scheme can preserve the confidentiality of all data in high efficiency. To ensure the security of all data, this scheme requires that the user must be a trusted authorized user. But how to solve the k-NN query request of untrusted users is also a challenge, and this scheme still needs to be verified on the real eHealthcare data set. For this reason, in this literature, a k-NN query scheme for untrusted users was proposed [24]. To solve the security problem, the data owner is required to be online for a long time, which will bring frequent interactions between the data owner and the cloud, so this scheme still has room for improvement in terms of efficiency. A secure k-NN query scheme using a homomorphic encryption technique to encrypt the database is proposed in [25]. In [7], an optimized privacy-preserving k-NN query processing scheme base on [25] is proposed. These two schemes while preserving the confidentiality of all data can ensure the security of user access patter. However, since they both utilize two cloud servers, message transmission with encrypted data is required. As a result, the query cost is not low. Considering the confidentiality of the key and the controllability of the query, Zhu *et al.* [26] propose a new scheme. While preserving data privacy, this scheme also achieves key confidentiality and query controllability. But since the data owner is required to be online at the time of encrypting each query, this also brings inconvenience to the data owner.

Many existing protocols are impractical for widely used in the real world because of quite inefficient due to frequent interactions [27]. Some schemes mentioned above require multiple interactions between server and client, or between server and server, which seriously affects their efficiency. If the interactions can be reduced, then the efficiency must be improved.

Some work [28]–[30] have studied on non-interactive privacy-preserving classification scheme. However, these schemes have drawbacks regarding classification accuracy or the weak level of security. In [31], a non-interactive k-NN scheme encrypted the outsourced data by using order-preserving encryption and homomorphic encryption is proposed. In this scheme, outsourced data encrypted by two cryptosystems to preserving data privacy, and extensive experiments prove that the classification accuracy of ciphertext is very little different from that of plaintext. Nevertheless, the runtime still needs to be improved because this scheme needs to traverse the entire data set to achieve the data query. Besides, because the OPE is deterministic, and if the cloud server can have a part of the information related to all encrypted data, the scheme is not secure under inference attacks based on frequency analysis. Therefore, this scheme is more suitable for a private data set.

In summary, there still has much work to do in terms of guaranteeing the confidentiality of data and access patterns, as well as the performance of the k-NN algorithm.

III. PRELIMINARIES

We first briefly describe the k-NN classifier and the kd-tree technique. As mentioned above, when users perform data processing tasks on their data, such as data hiding [32] and data analysis, they usually adopt encryption algorithms to protect all data. Therefore, two encryption algorithms are used in our proposed scheme. Then we review the two encrypt algorithms utilized to encrypt the raw data and query data records in our proposed scheme: the Order Preserving Encryption (denoted by OPE) and the Paillier cryptosystem.

A. THE k-NN CLASSIFIER AND KD-TREE

The k-Nearest Neighbor classifier is a typical machine learning algorithms. Find k objects closest to the test object, then according to the class value of these k objects, vote on the class that appears most frequently and assign it to the unlabeled instance. Each object has an associated class label and several attributes. A query record is "the test object" that needs to be labeled.

The traditional k-NN algorithm must traverse the entire data set to find the nearest k points. Moreover, in the traversal process, to hide the data access pattern, it must perform encryption-based operations, which will increase the execution time of the algorithm [7], [25].

As mentioned above, many PPkNN (Privacy-preserving k-NN) algorithms can achieve privacy-preserving, but they will bring a huge computing overhead and affect the efficiency of the scheme.

Therefore, to fix this problem, we adopt the kd-tree technique to improve the efficiency of retrieval.

The kd-tree is a space-partitioning data structure that organizes k-dimensional data points [33], and it can efficiently process the nearest neighbor search problem of multi-dimensional data.

Definition of kd-tree: A kd-tree is a special binary tree where each node is a k-dimensional point. All non-leaf nodes can be divided into two half-spaces by acting as a hyperplane. The left subtree of the node represents the point to the left of the hyperplane, and the right subtree represents the point to the right of the hyperplane. Each node is related to the one dimension in the k dimension that is perpendicular to the hyperplane, so in order to achieve a balanced kd-tree, the method of selecting the hyperplane usually determines the segmentation axis based on the largest variance of each dimension of all nodes. Therefore, if the "x" axis is selected, all nodes with "x" value less than the specified

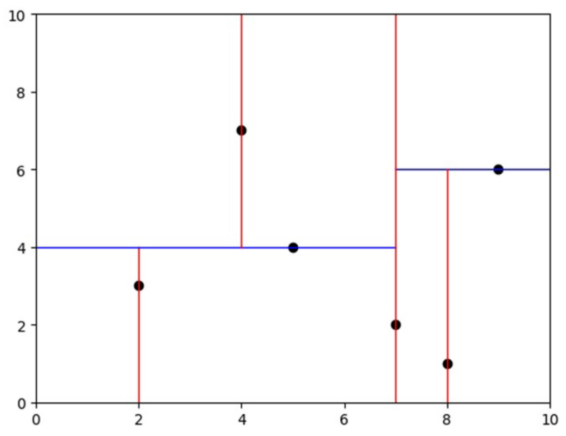

FIGURE 1. An example in two-dimensional space.

value will be placed in the left subtree, and all nodes with "x" value greater than the specified value will be placed in the right subtree. In this way, the hyperplane can be determined by this "x" value.

The advantage of kd-tree: Using kd-tree to implement the traditional k-NN algorithm can improve the search speed, especially in the face of facing massive high-dimensional data. Because the k-NN algorithm needs to traverse the entire data set to achieve the query, but in general, the kd-tree technique does not need to traverse the entire data set. Hence, the average time complexity of a traditional k-NN algorithm is $\boldsymbol{O}(\mathbf{n})$, while that of kd-tree search is $\boldsymbol{O}(\mathbf{logn})$. Obviously, the kd-tree technique can effectively improve the efficiency of a privacy-preserving k-NN algorithm. Therefore, the kd-tree technique has become a key component of our scheme.

Example 2: Assume Alice (the data owner) has eight two-dimensional data points. The data is divided into four planes according to the kd-tree algorithm, as shown in Figure 1 [33].

There is a pseudo code for the kd-tree as follows, for a kd-tree storing the multidimensional location of a set of points in a data set, and this procedure generalizes to any dimension of input.

Firstly, a root node is constructed. The hyperplane region corresponding to the root node contains all the instance points in the k-dimensional space. Secondly, the following recursion method is used to segment the k-dimensional space and generate child-nodes.on the hyperplane; We select some attribute and the median of this attribute values, which is a split point to determine a hyperplane, such that the database is split into two-part. Therefore, all records are assigned to the two sub-areas. Repeat the process until there are no instances to assign. During this process, the instances are saved on the corresponding node.

B. ORDER PRESERVING ENCRYPTION

In the field of practical application and research, many systems use order-preserving encryption schemes to facilitate untrusted servers to perform comparison operations on ciphertexts. As a deterministic encryption scheme,

Algorithm BKT: BuildKDTree()

```
1:  Input: D //D is the data set
2:  Output: node of the kd-tree
1:   tree_Node properties:
2:   {
3:       values of the point;
4:       axis; // the axis used for the pivot element
5:       tree_Node* parent
6:       tree_Node* left
7:       tree_Node* right
8:   }
9:   // Select axis based on the largest variance of each
       dimension through all points
10:   // Sort point list based the axis and select median
       as pivot element
11:   select median by axis from pointList;
12:   // Create node and construct subtrees
13:   node.location = median;
14:   node.leftChild = BuildKDTree(points in pointList
       that are less than median);
15:   node.rightChild = BuildKDTree(points in pointList
       that are greater than median);
16:   return node;
17:   }
```

the ciphertexts generated by the Order Preserving Encryption (denoted by OPE) can retain the numerical order of the plaintexts. The OPE was first proposed [34] in 2004, and Boldyreva *et al.* [35] conducted the first formal study of its concept and security in 2009.

Order Preserving Encryption assures that the order of plaintexts remains in the ciphertexts. That is, given the plaintexts m_1 and m_2 and their corresponding ciphertexts c_1 and c_2, the following expressions are correct:

$$m_1 \leq m_2 \Rightarrow c_1 \leq c_2 \tag{1}$$

The OPE contains three algorithms: key generation $KeyGen(K_{ope})$, encryption $Enc_{ope}(K, m)$ and decryption $Dec_{ope}(K, c)$.

In addition to the usual correctness requirement:

$$Dec_{ope}(Enc_{ope}(K, m)) = m \tag{2}$$

for every plaintext m and key K_{ope}, we require that: $m_1 \leq m_2$ iff $Enc_{ope}(K, m1) \leq Enc_{ope}(K, m2)$.

C. THE PAILLIER CRYPTOSYSTEM

The Paillier cryptosystem is a typical homomorphic encryption scheme [36], which has been widely used in many fields of information security. Especially in recent years, based on its homomorphism, it is widely used to encrypt the outsourced data in the field of privacy-preserving. It relies on three algorithms for data encryption and decryption: key generation $KeyGen(\lambda)$, encryption $E_{pk}(pk,m)$ and decryption $D_{sk}(sk,c)$.

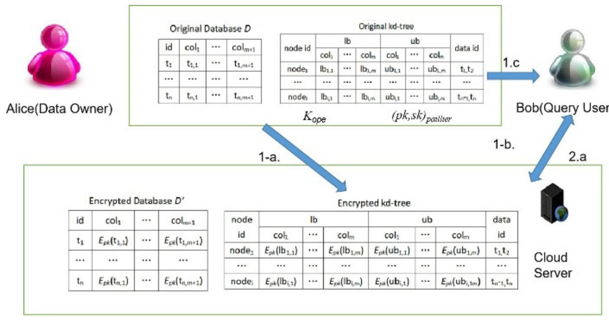

FIGURE 2. System model.

The Paillier cryptosystem has the following two properties:

(1) Homomorphic addition: Given two ciphertexts $E_{pk}(a)$ and $E_{pk}(b)$, we have:

$$E_{pk}(a+b) = E_{pk}(a)^{*}E_{pk}(b) \tag{3}$$

(2) Homomorphic multiplication: Given a ciphertext $E_{pk}(a)$ and a plaintext $b \in \mathbb{Z}_n$:

$$E_{pk}(a * b) = E_{pk}(a)^{b} \tag{4}$$

IV. MODELS AND DESIGN GOALS

This section will introduce the system model and the security model of this new scheme and confirm design goals.

A. SYSTEM MODEL

As shown in Figure 2, we designed three entities, namely the data owner, an authorized user, and a cloud server that offers k-NN query service.

In Figure 2, it can be seen that the system is comprised of three kinds of entities: the data owner (Alice), a cloud server and authorized user (Bob). As mentioned above, Alice owns an unencrypted data set D of n records $t_1, \ldots, t_n,$ and $m+1$ attributes. To facilitate the provision of secure query services, Alice encrypts data attributes, then utilizes the kd-tree technique to index the data so that the corresponding data records can be found more quickly.

For the purpose of preserving data privacy, Alice encrypts D with the secret key K_{OPE} of the OPE and the public key pk of the Paillier cryptosystem before outsourcing. More specifically, we use the K_{OPE} to encrypt the attributions $\mathrm{t}_{i,j}$ which denote the jth attribute value of record t_i, for $1 \leq i \leq n$ and $1 \leq j \leq m$, while the pk is used to encrypt the special attribution: the class label of each record, that is $\mathrm{t}_{i,j}$, for $1 \leq i \leq n$ and $j = m+1$. Therefore, Alice generates $\mathrm{Enc}(\mathrm{t}_{i,j})$, for $1 \leq i \leq n$ and $1 \leq j \leq m+1$.

To support the k-NN algorithm over encrypted data, the data owner Alice outsources the encrypted data set and its encrypted index to the cloud (1-a in Figure 2) and sends the K_{OPE} to the user Bob as well as the $sk_{paillier}$ (1-b in Figure 2).

The query user Bob receives the K_{OPE}, then encrypt his query request. Like Alice, Bob sends the encrypted query request to the cloud (2-a in Figure 2).

The cloud server gets all the necessary information, perform the proposed protocols. At last, it returns the query result to the query user Bob (2-b in Figure 2).

B. SECURITY MODEL

In this security model, we assume that Alice is *honest*, i.e., Alice outsources the encrypted raw data honestly. But we assume the cloud server is *honest-but-curious*, which means it will faithfully follow the protocol and the procedure to provide classification service and strive to return the answers the designer wants, but the cloud server is also *curious*, it will always attempt to find out as much as possible sensitive information during program execution, including encrypted raw data, user query records, and intermediate results in the program. For users, we assume that these authorized users are honest.

The cloud server cannot find out any value of any component of any vector it gets, including the user's query record. It also cannot find out any information about the class labels, and because the Paillier homomorphic encryption scheme used in encrypting class labels is probabilistic, the cloud cannot even learn the number of different class labels in the data set. In addition, due to the ciphertext addition used in the scheme, it cannot recover the class label that the user wants. Of course, we will be considered other forms of attacks occurring in the cloud in our future studies, such as inference attacks based on frequency analysis.

C. DESIGN GOALS

The design goal of our scheme is to accomplish a privacy-preserving, accurate and efficient k-NN query and classification scheme over outsourced data for authorized users. Specifically, the following goals should be satisfied:

- *Privacy preservation:* a privacy-preserving k-NN query scheme is proposed where privacy preservation is the most important and most basic security requirement for the scheme. Namely, all data, whether it is encrypted data that exists in the cloud or authorized users' query records, should be effectively protected from all unauthenticated entities, including the cloud server and unauthenticated users in our scheme.
- *Classification accuracy:* the k-NN query is a classifier, and the accuracy of the classification is a key factor that determines the effectiveness of the query. According to the principle of the k-NN algorithm, accurate classification can meet the query requirements of users. Therefore, our scheme must be able to obtain accurate results that are almost identical to the k-NN algorithm performed using the plaintext data.
- *Efficient computation:* The traditional privacy-preserving k-NN algorithm leads to more additional computations to preserve data privacy. In specific, performing k-NN queries on encrypted data usually consumes more computational resources than executing that algorithm in plaintext. Therefore, we use the kd-tree technique

to achieve an encrypted index search scheme. While preserving the access mode of data, the advantages of the kd-tree technique are used to improve the speed of data traversal and effectively improve computational efficiency.

V. THE PROPOSED PROTOCOLS

In this section, we introduce our k-NN query scheme. This scheme consists of the two stages: secure search of k-NN and secure selection of majority class.

- Stage 1 - Secure Search of k-NN (SSkNN):

In this stage, Alice encrypts the raw data set D with the OPE and the Paillier cryptosystem, and builds the encrypted kd-tree, then uploads them to the cloud. First, Bob encrypts his query q and submits it to the cloud. Then, the cloud server uses the k-NN algorithm to securely search the k nearest neighbors corresponding to q and obtain the class labels of the k neighbors. Note that at the end of this stage, only the cloud knows the encryption class labels for the k neighbors.

- Stage 2 - Secure Selection of Majority Class (SSMC):

After finding the k-nearest neighbor of q, in this stage, the cloud server can calculate a unique class label by a majority vote of the k-nearest neighbors of the input query q, and then send the class label to Bob. In the entire protocol, no one but Bob knows the class label corresponding to q.

A. STAGE 1: SECURE SEARCH OF K-NN (SSKNN)

According to the principle of the OPE scheme, we use the set $D = \{1, 2, \ldots, M\}$ as the plaintext space and the set $R = \{1, 2, \ldots, N\}$ as the ciphertext space. For $N \geq M \in \mathbb{N}$, these two integers represent the possible number of plaintexts and ciphertexts, respectively.

Initially, Alice proceeds the OPE and the Paillier cryptosystem on the database D, respectively, as mentioned above, we encrypt the attributions of all records except the class label by OPE and encrypt the special attribution-class label by the Pailliar cryptosystem. The procedure of the initialization is shown in Algorithm 1.

Algorithm 1 Initialization for D

```
1: Input: $(t_1 \ldots\ldots t_n) \in \mathbb{Z}^{n\times p}$, $(c_{1\ldots\ldots}c_\omega) \in \mathbb{N}$
2: $K$ = OPE.$keygen(M, N,$ s)
3: $(pk, sk)$ = Paillier.$keygen(\lambda)$
4: for i ← 1 to $n$ do
5:     $v_i \leftarrow$ OPE.$Enc$ $(K, t_i)$
6:     $x_i \leftarrow$ Paillier.$Enc$ $(pk, 2^{\mathfrak{C}_i \cdot \Delta})$
7: end for
8: Send $(t_1, \cdots, t_n)$ and $(x_1, \cdots, x_\omega)$ to the cloud
9: Send $K$ and $sk$ to the query user Bob
```

In Algorithm 1, a secret key K and a key pair (pk, sk) are generated according to the OPE.$keygen(M, N,$ s) algorithm and the Paillier.$keygen$ (λ) algorithm, respectively. As mentioned earlier, K is used to encrypt the attribute values of records other than the class labels, and pk is used to encrypt the class labels.

In order to better protect data privacy, we do not directly use pk to encrypt the class label of the data. Instead, we return the integer $2^{\mathfrak{C}_i * \Delta}$ [31] by utilizing the class label $c \in \mathbb{N}$ and a positive integer Δ. For instance, assume $\Delta = 16$, class 1 is mapped to $2^{2*16} = 2^{32}$.

As mentioned before, the querying user Bob also encrypts his query; in the meantime, we submit the $k \in \mathbb{N}$ (the number of nearest neighbors) to the cloud. The procedure of the initialization is shown in Algorithm 2. In this algorithm, the query q from Bob uses the same K which is sent from Alice.

Algorithm 2 Query for Bob's q

```
1: Input : $q \in \mathbb{Z}^p$
2: $q' \leftarrow$ OPE.$Enc$ $(K, q)$
3: Choose the number of neighbors $k \in \mathbb{N}$ and send $(q', k)$ to the cloud.
```

The traditional privacy-protected k-NN scheme usually generates higher computational cost because of traversing all the encrypted data and performing encryption-based operations [5]. For this reason, in our proposed scheme, we devise an encrypted index search scheme. We use kd-tree as the index structure, and then encrypt each node of the kd-tree by the OPE. Note that the "k" of kd-tree used in the word " k -dimension" means the number of the dimension of each record, and the "k" of k-NN used in the word "k-Nearest Neighbors" means the number of nearest neighbor(s) of the query q.

According to the Algorithm BKT mentioned in section 3.2, In Algorithm 3, we utilize the BKT algorithm to get the kd-tree based on the encrypted database D'. Each node contains all the encrypted information except the class label to each record in the database D'.

We input the encrypted data set D' and then invoke the BKT algorithm to get an encrypted kd-tree. This process is a pre-processing process, according to our algorithm, which can be completed before the query work.

Algorithm 3 Build an Encrypted kd-tree(BEKDTree)

```
Input: $D'$
Output: An encrypted kd-tree, each node of the kd-tree build based on $D'$
1: BKT($D$)
2: return node
```

According to the previous algorithm, the cloud server has received encrypted q, the number of nearest neighbors k and encrypted kd-tree. Let $\boldsymbol{L}$ be the $E(cand)$, a list of k vacancies to store the k nearest point(s) which will be found. The function "Encrypted kd-tree search" is shown in Algorithm 4.

First, search downward according to the coordinate value of $E(q)$ and the split axis of each node. That is if the nodes

Algorithm 4 Encrypted kd-tree Search

```
Input: $E(q)$, $E(kd\text{-}tree)$, $k$(in $k$-NN search),
Output: $E(cand)$ // an array contains all data $t_i$ inside nodes related to a query
1. $L \leftarrow 0$ // $L$ is the $E(cand)$, a list of $k$ vacancies to store
        the nearest point(s) which will be found.
2. $z \leftarrow 0$ //($z$ is the number of nearest neighbor(s))
3. $k$NN_index is ready for the indexes of the $k$ nearest points.
4. Getdis( ){
5.    for $j \leftarrow 1$ to $m$ do
6.      $dis \leftarrow \| (E(t_i)\text{-}E(q) \|$ //get the distance
7.    end for
8.  }
10. if $E(q_1) \leq E(t_{root1})$// Compare the values of the one dimension between two points.
11. then enter the left tree
12. else enter the right tree
13. $E(t_{leaf}) \leftarrow$ recurve (10)-(12) //$E(t_{leaf})$ is the leaf node of the current tree
14. if $z < k$
15. then put the $t_{leaf}$ into $L$, $z \leftarrow z + 1$, save the index and
        calculate dis (the distance between the node and $q$)
16. else if dis < MaxDis //MaxDis is the biggest distance in $L$
17. then replace the farthest point.
18. Traverse upward, and for each node:
19. Calculate the distance dis' between $E(q)$ and another
        child node of the parent node the current
        node belongs to.
22. if dis' < MaxDis || $z < k$
23. then execute (10)
24. execute (18)-(21)
25. compute the indexes (index[$1$], . . . , index[$k$]) of the $k$
        smallest distances among (d$1$, . . . , d$n$)
26. $class_q \leftarrow x_{index[i]}$
27. for $j \leftarrow 2$ to $k$ do
28.   $class_q \leftarrow$ Paillier.add($class_q$, $x_{index[i]}$)
29. end for
30. return $class_q$ to the query user Bob.
```

of the tree are split according to the x-coordinate, and the x-coordinate value of $E(q)$ is less than the root node of the encrypted kd-tree or the subtree, search to the left tree, otherwise go the right tree.

Second, When we reach a bottom node, it is marked as accessed. If there are less than k points in L, add the current node to L; if L is not empty and the distance between the current node and $E(q)$ is less than the longest distance in L, then the farthest point in L is replaced by the current point.

Third, if the current node is not the top node of the whole tree, execute the step (a); Instead, output L and the algorithm is complete.

Step (a): Traverse upward a node.

If the current node has not been accessed, mark it as accessed and then execute the step(b) and (c); if the current node is accessed, execute (a) again.

Step (b): If there are less than k points in L at this time, add the node to L; if L is full of k points, and the distance between the current node and $E(q)$ is less than the longest distance in L, replace the farthest point in L with the node.

Step(c): Calculate the distance between $E(q)$ and the splitting plane the current node belongs to.

If the distance is less than the biggest distance in L or less than k points in L, there may have a closer point on the other side of the splitting line, so another branch of the current node starts from the first step.

After this traversal, we can get the k nearest neighbor. Save the index of the nearest neighbors. E.g., if $k = 5$, the *1st*, the *3rd*, the *6th*, the *8th,* and the *10th* vectors were the five nearest vectors of q, then d_1, d_3, $d_{6,}d_{8,}d_{10}$ were the five smallest distances required for the next stage, and this function would return the corresponding indexes.

B. STAGE 2: SECURE SELECTION OF MAJORITY CLASS(SSMC)

The authorized user receives a ciphertext class$_q$, decrypts it and extracts the class of the query vector q', as shown in Algorithm 5.

Algorithm 5 Select Majority Class

```
Input: Encrypted class classy
1. $class_{Bob} \leftarrow$ Paillier. $D_{sk}(sk, class_q)$
2. Compute maximum coefficient from $class_{Bob}$ as a polynomial in base $2^{\Delta}$.
3. Assign class $c_q$ to the query vector.
```

As shown in Algorithm 5, the server calculates the number of occurrences of each class in the k nearest neighbor vectors by accumulating the encrypted class labels. As mentioned earlier, because the i-th class is an integer $2^{i*\Delta}$, the sum of the classes is the integer $a_0 + a_{1*}2^{\Delta} + a_{2*}2^{2*\Delta} + \ldots + a_{s*}2^{s^*\Delta}$, where each coefficient a_i records the number of occurrences of the i-th class. In addition, since each class label is encrypted by the Paillier cryptosystem, the server can implement the addition of the class by using the "homomorphic addition for and decrypt the obtained ciphertext. At the same time, we obtain the largest coefficient by shifting and reducing the decrypted sum $class_{Bob}$ modulo 2^{Δ}.

C. SECURITY ANALYSIS

We analyze the security/privacy of our scheme. We mainly focus on the confidentiality of all data and data access patterns. In other words, all data, including encrypted data belonging to Alice and Bob, respectively, intermediate results generated by program execution, etc., and data access pattern should be privacy-preserving. Namely, the access patterns of these data and data cannot be leaked to the cloud.

As mentioned above, we assume that the cloud server is *honest-but-curious*, that is, it will follow the protocol and execute the k-NN algorithm as required, and return the correct results, even though it will be very curious about the information during the execution of the query processing.

- *The Encryption technique is Privacy-Preserving:* In this scheme, the cloud can't obtain the value of any data (including query data records) because we use two encryption schemes, the OPE and Paillier cryptosystem, to encrypt the raw data and execute the k-NN algorithm. As mentioned earlier, the *honest-but-curious* cloud server cannot know any information about the encryption class, or even how many different classes in the scheme. And all intermediate results and the final result of the query are encrypted data, and the cloud server cannot learn their plaintext data. Therefore, the encryption technique is privacy-preserving.
- *The Encrypted Data Comparison Protocol is Privacy-Preserving:* The data comparison protocol is a key component of the k-NN query algorithm. As mentioned earlier, the comparison protocol is used in both the kd-tree technique and the comparison of distances. In our proposed scheme, we implemented the k-NN algorithm with the stateless OPE scheme presented in 2011 [37]. From this protocol, we can only obtain a comparison of two encrypted data $E(a)$ and $E(b)$ without leaking any plaintext values of a and b. Therefore, the cloud server cannot directly get the plaintext data content. Similarly, Nor can unauthorized users. Thus, this comparison protocol is privacy-preserving.
- *The Euclidean Distance Computation Protocol is Privacy-Preserving*: In the k-NN query algorithm, the usual method for determining the distance between two data points is to compute the Euclidean distance between the two points. Thus, the Euclidean distance computation protocol is the key component of the k-NN algorithm. We adopt this protocol which runs on the cloud server. When the cloud server computes the distance between two points, that is, when computing the distance between the authorized user's query record q and the outsourced data, these data both have already been encrypted with the OPE. The security of the OPE scheme guarantees that it is difficult for the cloud server to obtain any data records. Therefore, in this protocol, the data cannot be obtained from the encrypted data.

Meanwhile, when computing the Euclidean distance between two encrypted data records $E(x)$ and $E(q)$, the cloud server obtains the encrypted $dis(x, q)$. However, since the OPE scheme used in our proposed scheme does not leak the plaintext data and the exact value of the distance between different plaintext data [37], therefore the cloud server is unable to recover the $dis(x, q)$ from the Euclidean distance computation protocol, let alone the data content.

From the above analysis, it can be seen that neither the cloud server nor the unauthorized user can acquire any data from the Euclidean distance computation protocol. Therefore, this distance computation protocol used in our proposed scheme is privacy-preserving.

TABLE 2. Data set used in the experiments.

Data set	Instances	Attributes
IIRIS	150	4
WINE	178	13
Climate Model	540	18
HEART DISEASE (Cleveland)	297	13
WFR (Wall-Following Robot Navigation)	5456	24
Skin Segmentation	226821	3

VI. EXPERIMENTAL RESULTS

We now use some experiments to prove the performance of our privacy-preserving k-NN protocol. All our various experiments are performed by using C++ on the Ubuntu 16.04 LTS 64-bit with Intel(R) Xeon(R) Silver 4110 CPU @ 2.10GHz(X2) processor and 16GB RAM. We assess the performance of our scheme, using several real data sets. For comparison, we use the real data sets from the UCI KDD archive [38], as shown in Table 2.

In this paper, three schemes are used to implement the k-NN algorithm: the classical plaintext k-NN classifier, the scheme in [31] and our proposed scheme.

In the scheme we proposed and the scheme proposed in [31], a stateless OPE scheme [37] and the Paillier homomorphic encryption scheme are used to encrypt data, including outsourced data and query records of authorized users. To be specific, the two schemes adopt the OPE scheme to encrypt the property values of the data and use the Paillier encryption scheme to encrypt the class labels of the data. Moreover, in the process of data query and classification, the comparison protocol and the computation protocol are executed between encrypted data, so the privacy-preservation of all data is achieved. Therefore our proposed scheme and the scheme proposed in [31] have the same security.

We collect the results of two metrics: the accuracy of the classification and the runtime of the query. Specifically, we compare the accuracy of the three schemes for the same data sets, and the runtime of the query between our scheme and the scheme proposed by [31]. For fair performance analysis, we made the same parameter with [31] to test.

- *Accuracy of classification:* In Tables 4, the table has five columns. Each column represents the comparison of the data classification accuracy of the three schemes facing different data sets when the nearest neighbor number k is equal to 1, 3, 5, 7, and 9, respectively.

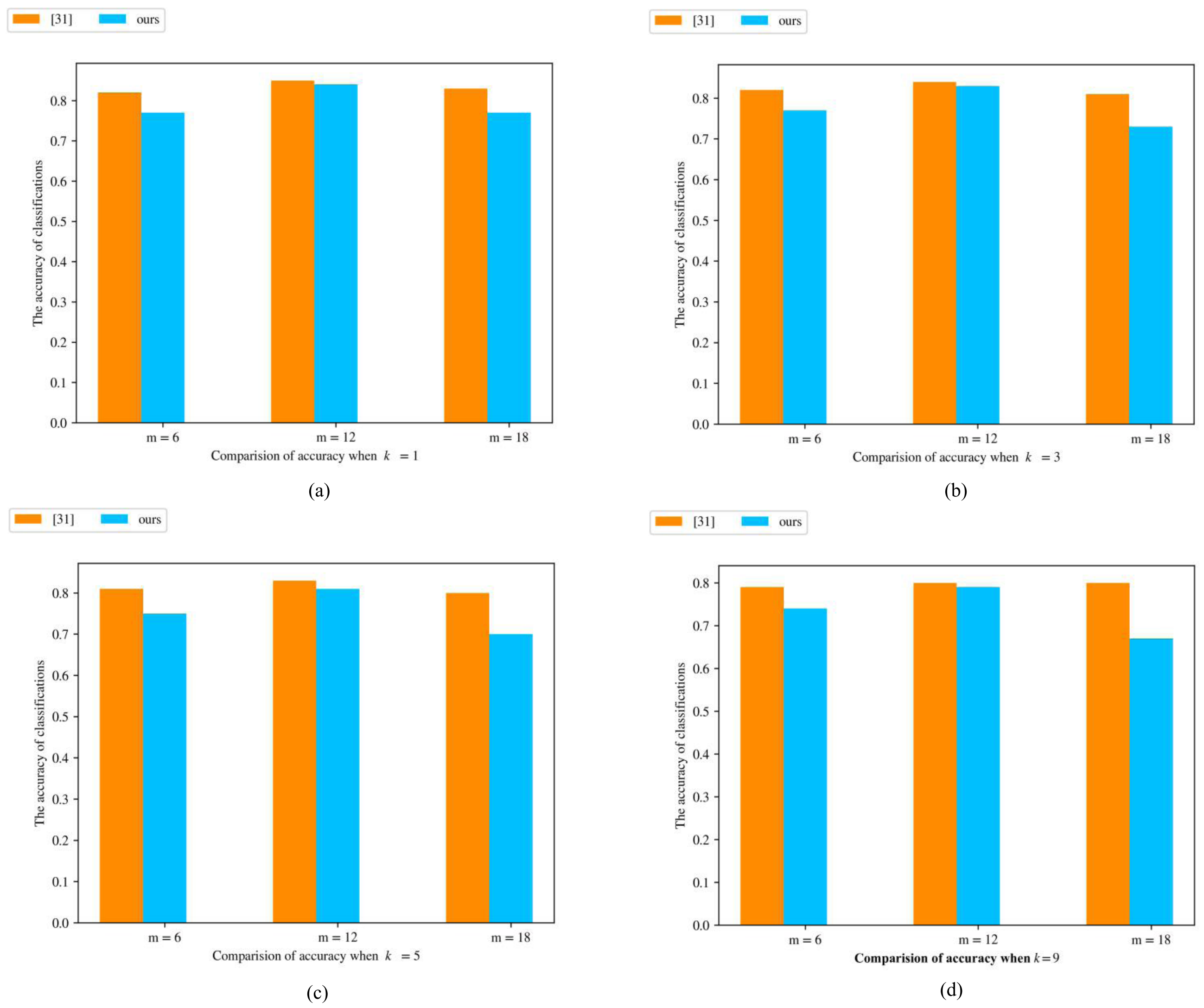

FIGURE 3. (a). Comparison of accuracy of different dimensions when $k = 1$. (b). Comparison of accuracy of different dimensions when $k = 3$. (c). Comparison of accuracy of different dimensions when $k = 5$. (d). Comparison of accuracy of different dimensions when $k = 9$.

First of all, as shown in Table 4, the comparison of the accuracies of the classification between the traditional k-NN with plaintexts and the two schemes with encrypted data. Moreover, the two schemes use the same OPE parameters $(M, N) = (2^{32}, 2^{40})$.

Over six different data sets, our scheme, and plaintext scheme are similar in most data sets, such as Iris data set, Wine data set, Heart Disease (Cleveland) data set, and Climate Model data set and Skin Segmentation data set. Especially on the Climate Model data set and Skin Segmentation data set, our proposed scheme is even better than plaintext scheme. The accuracy of the two methods based on encrypted data is also similar, and on some data sets, our solution is even better. Such as the Wine data set. However, when facing the WFR (wall-following Robot Navigation) data set, the accuracy gap between the two encryption schemes and the plaintext scheme is obvious. The main reason is that the data is processed before the k-NN query. Because the data type of the attribute characteristics in the WFR data set is real, but our encryption scheme can only handle integers, so rounding is taken. So obviously, in most cases, the accuracy of our scheme is as good as that of [31]. However, it should be noted that for data set ''Heart Disease (Cleveland)'', the accuracy of both schemes (the plaintext scheme, [31] and our scheme) is not good enough. It is quite clear that the k-NN algorithm is not suitable for this data set.

Second, in order to study the impact of different dimensions on the accuracy of the query, we compared the accuracy of the two schemes – the scheme proposed by [31] and our scheme – with a partial subset of the WFR data set, as shown in Figure 3. The main parameters are set as follows: the dimension of the data set is set as $m = 6, 12, 18$, and the number of the nearest neighbors is set as $k = 1, 3, 5, 9$. According to Figure 3, we can see that under different dimensions,

TABLE 3. Comparison of accuracies.

Data sets	*k*=1			*k*=3			*k*=5			*k*=7			*k*=9		
	Plain	[31]	Ours	Plain	[31]	Ours	Plain	[31]	Ours	Plain	[31]	Ours	Plain	[31]	Ours
Iris	0.98	0.98	**0.96**	0.96	0.98	0.98	0.94	0.98	**0.96**	0.92	0.96	0.96	0.94	0.98	**0.96**
Wine	0.68	0.64	0.63	0.68	0.58	0.58	0.71	0.54	**0.51**	0.71	0.54	**0.56**	0.66	0.54	0.54
WFR	0.99	0.81	0.82	0.99	0.82	0.82	0.99	0.81	0.81	0.99	0.80	0.80	0.98	0.80	0.79
Heart Disease (Cleveland)	0.47	0.44	0.47	0.46	0.52	**0.46**	0.49	0.49	0.51	0.49	0.49	0.49	0.49	0.51	**0.49**
Climate Model	0.89	0.91	**0.89**	0.93	0.93	0.93	0.93	0.93	0.93	0.92	0.92	0.92	0.92	0.92	0.92
Skin Segmentation	0.97	0.97	0.97	0.98	0.97	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.97

TABLE 4. Comparison of the runtime in seconds to classify.

Data sets	*k*=1		*k*=3		*k*=5		*k*=7		*k*=9	
	[31]	Ours	[31]	Ours	[31]	Ours	[31]	Ours	[31]	Ours
Iris	0.01	**0.000712**	0.01	**0.001248**	0.01	**0.001787**	0.01	**0.002347**	0.01	**0.003028**
Wine	0.03	**0.002964**	0.03	**0.004057**	0.03	**0.005779**	0.04	**0.007249**	0.04	**0.009222**
WFR	64.19	**0.249829**	64.68	**0.34541**	67.46	**0.441639**	68.38	**0.543563**	71.24	**0.643324**
Heart Disease (Cleveland)	0.10	**0.005046**	0.10	**0.008466**	0.10	**0.01102**	0.10	**0.014116**	0.10	**0.017032**
Climate Model	0.47	**0.013752**	0.48	**0.020002**	0.47	**0.02906**	0.47	**0.034813**	0.47	**0.04112**
Skin Segmentation	36933	**2.75073**	37842	**3.7058175**	37983	**4.7089375**	38359	**5.22613**	38842	**5.9262475**

the accuracy of our scheme is not much different from that of the scheme in [31]. Especially when $m = 12$, the accuracy of the two schemes is almost the same. The accuracy of [31] is 0.84, and the accuracy of our scheme is 0.83. Based on the complete WFR data set, the accuracy calculated by the above two schemes (according to Table 3, the accuracy of the two schemes is 0.80 and 0.79 on average) is the same.

In addition, under the same dimension m and different k values, the accuracy difference between the two schemes is not so big. For example, when $m = 6$, $k = 1, 3, 5$ and 9, the accuracy of our scheme is 0.77, 0.77, 0.75 and 0.74. However, when $m = 18$ and $k = 1, 3, 5, 7, 9$, the accuracy of our scheme was significantly reduced, which required further research on the influence of the algorithm and attributes of these data on the classification results. In particular, when $k = 9$, the accuracy dropped to 0.67, as shown in Figure 4. It can also be seen from Figure 4 that accuracy does not necessarily increase with the increase of dimensions, which is also related to the influence of data quality.

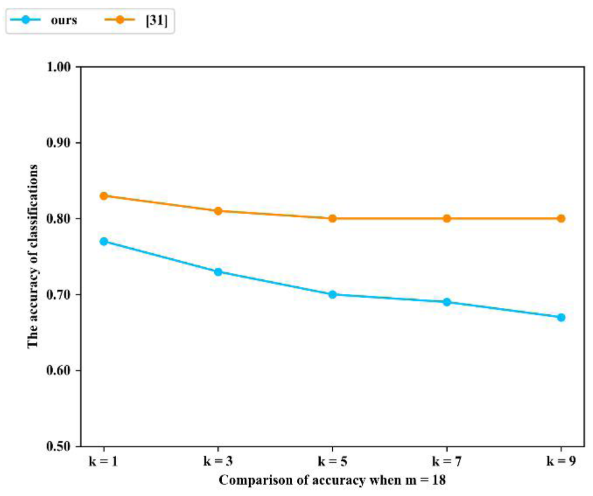

FIGURE 4. Comparison of accuracy of different *k* when m = 18.

- *Runtime of the query:* A comparison of the runtime to classify the data sets is shown in Table 4. We divide each

TABLE 5. Comparison of the runtime in seconds of a subset of the WFR dataset.

dataset	k=1		k=3		k=5		k=7		k=9	
	[31]	Ours	[31]	Ours	[31]	Ours	[31]	Ours	[31]	Ours
WFR(1000)	1.993	**0.040**	1.985	**0.056**	2.002	**0.074**	1.963	**0.090**	1.986	**0.109**
WFR(2000)	8.170	**0.099**	8.105	**0.131**	8.210	**0.167**	8.096	**0.205**	8.180	**0.251**
WFR(3000)	18.684	**0.133**	18.428	**0.184**	18.908	**0.231**	18.729	**0.282**	18.846	**0.332**
WFR(4000)	35.014	**0.189**	34.798	**0.243**	34.812	**0.313**	35.708	**0.384**	35.687	**0.451**
WFR(5000)	56.066	**0.244**	55.914	**0.340**	56.083	**0.436**	57.483	**0.523**	56.079	**0.615**

TABLE 6. Comparison of the runtime of subsets of the WFR data set when $m = 6, 12, 18$.

data set	*k*=1		*k*=3		*k*=5		*k*=7		*k*=9	
	[31]	ours	[31]	ours	[31]	ours	[31]	ours	[31]	ours
WFR (*m*=6)	23.35	**0.08**	23.35	**0.12**	23.21	**0.15**	23.36	**0.18**	23.36	**0.22**
WFR (*m*=12)	38.54	**0.15**	39.16	**0.22**	39.80	**0.27**	39.32	**0.33**	38.39	**0.39**
WFR (*m*=18)	53.24	**0.21**	53.68	**0.28**	53.60	**0.36**	53.51	**0.44**	53.19	**0.52**

data set into a training set containing 70% of the raw data set and a testing set containing 30% 0f the remaining data. Afterward, we perform the two schemes-the scheme proposed in [31] and our scheme ten times respectively and calculate the average runtime.

What needs our attention is that, as we can see in Table 5, the runtime of our scheme is significantly less than that of the scheme proposed in [31]. Especially when the number of instances is very huge, such as data set WFR (the number of instances is 5,000) and Skin Segmentation (the number of instances is 226,821), the runtime of our scheme is far less than that of the scheme [31]. This is also the design goal of our scheme.

In order to investigate the relationship between the runtime and data set size, we perform experiments using abbreviated versions of the WFR data set. By limiting the number of instances, we only considered a subset of the data set, as shown in Table 5.

Where WFR (1000) denotes a subset of the WFR data set with 1000 instances, WFR (2000), WFR (3000), WFR (4000), and WFR (5000) are data sets with similar meanings.

In Figure 5, we can clearly see the similarities and differences in the runtime of the two schemes. First, as shown in Figure 5, the average runtime of the two schemes have a similar trend: the runtime grows linearly with the number of instances. Especially the increase in the runtime of the scheme in [31] is very large when the number of instances gradually increases. Instead, our scheme, with the increase of the number of instances, the runtime are increasing, but the extent of increase is far less than [31], exactly when m = 1000, 2000, 3000, 4000, 5000, the average runtime is **0.07s, 0.17s, 0.23s, 0.32s, 0.43s**, respectively.

Second, as the number of instances increases, the performance advantage of our scheme becomes more and

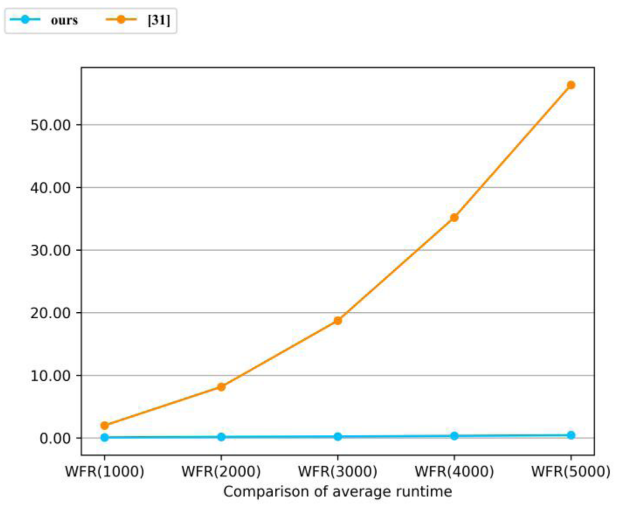

FIGURE 5. Comparison of the average runtime of subsets of the WFR data set when $k = 1, 3, 5, 7, 9$.

more obvious. When the number of instances is 5000 and k = 1, 3, 5, 7, 9, respectively, our runtime is **0.24s**, **0.34s**, **0.44s**, **0.52s**, **0.62s**, while the scheme proposed in [31] runs up to **56.07s**, **55.91s**, **56.83s**, **57.48s**, **56.08s**, respectively, about 100 times as long as the runtime of our scheme.

This is also because the scheme in [31] requires traversing all instances, which we do not need to traverse in so many instances.

This conclusion can also be reached according to the runtime of the above two schemes in Table5 based on the data set "Skin Segmentation." The same conclusion can be drawn from the runtime of the two schemes under different dimensions in Table 6.

In summary, based on the comparison of the accuracy and computational efficiency of the two schemes, our scheme is significantly better than the previous one.

VII. CONCLUSION AND FUTURE WORKS

In this paper, we have presented a new privacy-preserving and efficient k-NN query scheme over encrypted and outsourced. In this scheme, we use the OPE and the Paillier cryptosystem to encrypt the original data, the kd-tree technique to index the data to improve the query speed. Our proposed scheme achieves two characteristics: 1) the privacy-preserving k-NN query over encrypted outsourced data. 2) Efficient and accurate classification. Specifically, the accuracy of our proposed scheme is not much different from that of k-NN query which adopts plaintext data, but in terms of the runtime, we are much more efficient than the existing scheme [31], and the more data we have, the more significant our advantage will be. Security analysis shows that our solution is privacy protected.

As future work, because of the drawback of the OPE we used in this scheme, we will improve the security of our method by study the novel OPE. In addition, we will study how to design a privacy-preserving k-NN query scheme based on rational numbers. Also, we will study another strategy that is more effective for real data sets like the ''Heart disease (Cleveland)'' in Table 4 than encrypted kd-tree for better query processing performance.

REFERENCES

[1] L. Ou, H. Yin, Z. Qin, S. Xiao, G. Yang, and Y. Hu, ''An efficient and privacy-preserving multiuser cloud-based LBS Query scheme,'' *Secur. Commun. Netw.*, vol. 2018, Mar. 2018, Art. no. 4724815.

[2] M. Zhang, Y. Yao, Y. Jiang, B. Li, and C. Tang, ''Accountable mobile E-commerce scheme in intelligent cloud system transactions,'' *J. Ambient Intell. Hum. Comput.*, vol. 9, no. 6, pp. 1889–1899, Nov. 2018.

[3] M. Zhang, Y. Zhang, Y. Jiang, and J. Shen, ''Obfuscating EVES algorithm and its application in fair electronic transactions in public clouds,'' *IEEE Syst. J.*, vol. 13, no. 2, pp. 1478–1486, Jun. 2019, doi: 10.1109/JSYST.2019.2900723.

[4] Z. Liu, X. Huang, Z. Hu, M. Khurram Khan, H. Seo, and L. Zhou, ''On emerging family of elliptic curves to secure Internet of Things: ECC comes of age,'' *IEEE Trans. Dependable Secure Comput.*, vol. 14, no. 3, pp. 237–248, Jun. 2017.

[5] Y. Zhu, X. Li, J. Wang, and J. Li, ''Cloud-assisted secure biometric identification with sub-linear search efficiency,'' *Soft Comput.*, vol. 24, no. 8, pp. 5885–5896, Apr. 2020.

[6] A. Janosi, W. Steinbrunn, M. Pfisterer, and R. Detrano, ''UCI machine learning repository-heart disease data set,'' School Inf. Comput. Sci., Univ. California, Irvine, CA, USA, 1988. [Online]. Available: http://archive.ics.uci.edu/ml/datasets/Heart+Disease

[7] 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.

[8] R. Agrawal and R. Srikant, ''Privacy-preserving data mining,'' *ACM Sigmod Rec.*, vol. 29, no. 2, pp. 439–450, 2000.

[9] Y. Lindell and B. Pinkas, ''Privacy preserving data mining,'' in *Proc. Annu. Int. Cryptol. Conf.* Berlin, Germany: Springer, 2000, pp. 36–54, doi: 10.1007/3-540-44598-6_3.

[10] X. Li, Y. Zhu, J. Wang, Z. Liu, Y. Liu, and M. Zhang, ''On the soundness and security of privacy-preserving SVM for outsourcing data classification,'' *IEEE Trans. Dependable Secure Comput.*, vol. 15, no. 5, pp. 906–912, Sep. 2018.

[11] D. Zhu, H. Zhu, X. Liu, H. Li, F. Wang, H. Li, and D. Feng, ''CREDO: Efficient and privacy-preserving multi-level medical pre-diagnosis based on ML-kNN,'' *Inf. Sci.*, vol. 514, pp. 244–262, Apr. 2020.

[12] Z. Fu, X. Wu, C. Guan, X. Sun, and K. Ren, ''Toward efficient multi-keyword fuzzy search over encrypted outsourced data with accuracy improvement,'' *IEEE Trans. Inf. Forensics Security*, vol. 11, no. 12, pp. 2706–2716, Dec. 2016.

[13] Z. Xia, X. Wang, X. Sun, and Q. Wang, ''A secure and dynamic multi-keyword ranked search scheme over encrypted cloud data,'' *IEEE Trans. Parallel Distrib. Syst.*, vol. 27, no. 2, pp. 340–352, Feb. 2016.

[14] H.-I. Kim, H.-J. Kim, and J.-W. Chang, ''A secure kNN Query processing algorithm using homomorphic encryption on outsourced database,'' *Data Knowl. Eng.*, vol. 123, Sep. 2019, Art. no. 101602, doi: 10.1016/j.datak.2017.07.005.

[15] T. Ma, J. Zhou, M. Tang, Y. Tian, A. A-D, and M. A-R, ''Social network and tag sources based augmentin collaborative recommender system,'' *IEICE Trans. Inf. Syst.*, vols. E98–D, no. 4, pp. 902–910,2015.

[16] B. Amini, R. Ibrahim, M. S. Othman, and M. A. Nematbakhsh, ''A reference ontology for profiling Scholar's background knowledge in recommender systems,'' *Expert Syst. Appl.*, vol. 42, no. 2, pp. 913–928, Feb. 2015.

[17] A. A. Aburomman and M. B. Ibne Reaz, ''A novel SVM-kNN-PSO ensemble method for intrusion detection system,'' *Appl. Soft Comput.*, vol. 38, pp. 360–372, Jan. 2016.

[18] W. C. Lin, S. W. Ke, and C. F. Tsai, ''CANN: An intrusion detection system based on combining cluster centers and nearest neighbors,'' *Knowl.-Based Syst.*, vol. 78, pp. 13–21, Apr. 2015, doi: 10.1016/j.knosys.2015.01.009.

[19] T. Ma, J. Jia, Y. Xue, Y. Tian, A. Al-Dhelaan, and M. Al-Rodhaan, ''Protection of location privacy for moving kNN queries in social networks,'' *Appl. Soft Comput.*, vol. 66, pp. 525–532, May 2018.

[20] M. Shaneck, Y. Kim, and V. Kumar, ''Privacy preserving nearest neighbor search,'' in *Machine Learning in Cyber Trust*. Boston, MA, USA: Springer, 2009, pp. 247–276, doi: 10.1007/978-0-387-88735-7_10.

[21] Y. Qi and M. J. Atallah, ''Efficient privacy-preserving k-Nearest neighbor search,'' in *Proc. 28th Int. Conf. Distrib. Comput. Syst.*, Jun. 2008, pp. 311–319, doi: 10.1109/ICDCS.2008.79.

[22] W. K. Wong, D. W.-L. Cheung, B. Kao, and N. Mamoulis, ''Secure kNN computation on encrypted databases,'' in *Proc. 35th SIGMOD Int. Conf. Manage. Data (SIGMOD)*, 2009, pp. 139–152.

[23] Y. Zheng, R. Lu, and J. Shao, ''Achieving efficient and privacy-preserving k-NN Query for outsourced eHealthcare data,'' *J. Med. Syst.*, vol. 43, no. 5, p. 123, May 2019.

[24] Y. Zhu, R. Xu, and T. Takagi, ''Secure k-NN computation on encrypted cloud data without sharing key with Query users,'' in *Proc. Int. workshop Secur. cloud Comput. Cloud Comput.*, 2013, pp. 55–60.

[25] Y. Elmehdwi, B. K. Samanthula, and W. Jiang, ''Secure k-nearest neighbor Query over encrypted data in outsourced environments,'' in *Proc. IEEE 30th Int. Conf. Data Eng.*, Mar. 2014, pp. 664–675.

[26] Y. Zhu, Z. Huang, and T. Takagi, ''Secure and controllable k-NN Query over encrypted cloud data with key confidentiality,'' *J. Parallel Distrib. Comput.*, vol. 89, pp. 1–12, Mar. 2016.

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

[28] D. J. Wu, T. Feng, M. Naehrig, and K. Lauter, ''Privately evaluating decision trees and random forests,'' *Proc. Privacy Enhancing Technol.*, vol. 2016, no. 4, pp. 335–355, Oct. 2016.

[29] T. Graepel, K. Lauter, and M. Naehrig, ''ML confidential: Machine learning on encrypted data,'' in *Proc. Int. Conf. Inf. Secur. Cryptol.*, vol. 7839. Cham, Switzerland: Springer, 2012, pp. 1–21.

[30] J. W. Bos, K. Lauter, and M. Naehrig, ''Private predictive analysis on encrypted medical data,'' *J. Biomed. Informat.*, vol. 50, pp. 234–243, Aug. 2014.

[31] H. V. L. Pereira and D. F. Aranha, ''Non-interactive privacy-preserving k-NN classifier,'' in *Proc. 3rd Int. Conf. Inf. Syst. Secur. Privacy*, 2017, pp. 362–371.

[32] M. Zhang, S. Zhang, and L. Harn, ''An efficient and adaptive data-hiding scheme based on secure random matrix,'' *PLoS ONE*, vol. 14, no. 10, 2019, Art. no. e0222892. [Online]. Available: https://doi.org/10.1371/journal.pone.0222892

[33] *K-D Tree*. Accessed: Sep. 10, 2019. [Online]. Available: https://en.wikipedia.org/wiki/K-d_tree

[34] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu, ''Order preserving encryption for numeric data,'' in *Proc. ACM SIGMOD Int. Conf. Manage. Data (SIGMOD)*, 2004, pp. 563–574.

[35] A. Boldyreva, N. Chenette, and Y. Lee, ''Order-preserving symmetric encryption,'' in *Proc. Annu. Int. Conf. Theory Appl. Cryptograph. Techn.* Cham, Switzerland: Springer, 2009, pp. 224–241.

[36] P. Paillier, ''Public-key cryptosystems based on composite degree residuosity classes,'' in *Advances in Cryptology*, vol. 99. Berlin, Germany: Springer-Verlag, 1999, pp. 223–238.

[37] A. Boldyreva, N. Chenette, and O. A. Neill, "Order-preserving encryption revisited: Improved security analysis and alternative solutions," in *Proc. 31st Annu. Cryptol. Conf.*, Santa Barbara, CA, USA, Aug. 2011, pp. 578–595.

[38] *UCI Machine Learning Repository*. Accessed: Oct. 20, 2019. [Online]. Available: http://archive.ics.uci.edu/ml/data sets.html

JIANGYI DU received the M.S. degree from the Hubei University of Technology, in 2008. He is currently pursuing the Ph.D. degree with the State Key Laboratory of Information Engineering in Surveying Mapping and Remote Sensing, Wuhan University. His research interests include data mining and privacy preservation.

FULING BIAN is currently a Professor with the State Key Laboratory of Information Engineering in Surveying Mapping and Remote Sensing, Wuhan University. Her research interests include GIS, remote sensing, and spatiotemporal data mining.