International Journal of Engineering Inventions
e-ISSN: 2278-7461, p-ISSN: 2319-6491
Volume 11, Issue 3 [May.-June. 2022] PP: 344-358

A Fast Closest Neighbor Search Scheme over Outsourced Encrypted Medical Images

C.DASTAGIRAIAH[1], N. LIKHITHA SRI[2], M. THRISUDHA[3], SK. MOISE[4], D. MONIKA[5]

[1]*Associate Professor,Dept. of CSE, Sai Spurthi Institute of Technology,Khammam,Telangana,India*
[2,3,4,5,6]*B.Tech Student, Dept. of CSE, Sai Spurthi Institute of Technology, Khammam,Telangana,India*

Abstract

Medical imaging is crucial for medical diagnosis, and the sensitive nature of medical images necessitates rigorous security and privacy solutions to be in place. In a cloud-based medical system for Healthcare Industry 4.0, medical images should be encrypted prior to being outsourced. However, processing queries over encrypted data without first executing the decryption operation is challenging and impractical at present. In the paper, we propose a secure and efficient scheme to find the exact nearest neighbor over encrypted medical images. Instead of calculating the Euclidean distance, we reject candidates by computing the lower bound of Euclidean distance that is related to the mean and standard deviation of data. Unlike most existing schemes, our scheme can obtain the exact nearest neighbor rather than an approximate result. We then evaluate our proposedapproach to demonstrate its utility.

Index Terms—Cloud Computing, Efficiency, Medical Images, Nearest Neighbor Search, Privacy

---

Date of Submission: 02-06-2022 Date of Acceptance: 15-06-2022

---

1. INTRODUCTION

CLOUD computing is becoming a norm in our society [1],and in such a deployment, the data owner can outsource databases and management functionalities to the cloud server. The latter stores the databases and supplies access mechanisms to query and manage the outsourced database. This allows data owners to reduce data management expenses and improve quality of service. However, the cloud may not be fully trusted because it may leak sensitive information to unauthorized entities (e.g., compromised) or foreign government agencies
[2].
The rapid evolution of cloud computing is revolutionizing e-Health and the whole Industry 4.0 in the field of healthcare. The cloud-based electronic healthcare system is one popular application for Healthcare Industry 4.0. A well-designed electronic healthcare system can obviously improve the quality

This paper is supported by the National Natural Science Foundation of China under grant No. 61501080, 61572095, and 61871064, in part by the Cloud Technology Endowed Professorship, and in part by NSF CREST under Grant HRD-1736209.
C. Guo, S. Su, X. Tang are with the School of Software Technology, Dalian University of Technology, Dalian, 116620, China. C. Guo is also with the Key Laboratory for Ubiquitous Network and Service Software of Liaoning Province, Dalian, 116620, China (e-mail: guocheng@dlut.edu.cn, sushenghao@mail.dlut.edu.cn, tangxinyu.dut@gmail.com).
K.-K.R. Choo is with the Department of Information Systems and Cyber Security, University of Texas at San Antonio, San Antonio, TX 78249-0631, USA (e-mail: raymond.choo@fulbrightmail.org). of access and experience of healthcare users. In recent years, applications for cloud computing and big-data computing in the Healthcare Industry 4.0 have been enthusiastically discussed [3], [4], [5]. In an electronic healthcare system, patients' medical images may be outsourced to a third-party [6], such asa professional community healthcare cloud server. Within thehealthcare industry, many forms of medical imaging play a crucial role in diagnosis and quality of care, including magnetic resonance imaging (MRI), ultrasonic, computed tomography (CT) and computed radiography. In addition, for the healthcare industry, prior or archival records (e.g., disease outbreaks) can have significant reference value. When a new patient is seen, doctors can efficiently and accurately make a diagnosis and develop the appropriate treatment programs by finding

similarcases in the database and analyzing them. For example, hospitals or related medical institutions can store patients' medical images in a professional and secure database of a practical electronic healthcare system. If doctors acquire medical images from a new patient, they may find similar images in the outsourced database to be used as a reference. This is particularly crucial when dealing with symptoms fromrare diseases such as those listed on https://globalgenes.org/rarelist/ (last accessed January 16, 2018). Nearest neighbor search, therefore, can be used in such a scenario. However, the sensitivity and privacy of medical images require that the security and privacy of such images beensured and preserved.

Encrypting data by the data owner is a naive method to ensure privacy [7], while it ensures the secrecy of the outsourced data from the cloud and unauthorized users. Additionally, to protect query privacy, permitted users shouldsend their requests to the cloud for evaluation after encryption.However, by analyzing the data access patterns, the cloud (or a malicious insider) can derive private information about the real data items even though the data and queries are encrypted [8],[9]. In detail, the access pattern includes not only the content of the data block accessed by the user but also the way how the user accesses the data block, such as frequency, location, order, habit and so on. By statistical analysis, data mining or other techniques, the cloud server can infer user's type, hobbies andthe frequency of accessing specific content. For example, traffic analysis technique can obtain some sensitive information about access pattern. When a user searches through a service provider such as Google, the search history will leak the user's search habits even the identity. Also, the frequency of the search may leak the popularity of the retrieved data. Additionally, the cloud server can establish some correlation among successive accesses. In other words, we need to ensuresecrecy of the outsourced data and a user's query record in secure query processing, as well as hiding data access patterns. In order to be practical and efficient, the proposed scheme needs to reduce the computation cost on the end-user as muchas possible.

The nearest neighbor search is a vital operation in data mining, machine learning, and information retrieval, and more recently the healthcare industry as well. Recently, due to the emergence of high-dimensional medical images, the importance of having an efficient and effective nearest neighbor search algorithm (in terms of speed and space) has become more pronounced. Generally, it is a difficult task to process encrypted data without first executing the decryption operation. The challenge is how a cloud server can process the queries over encrypted medical images. Motivated by this challenge, we propose an efficient and effective scheme to search for the exact nearest neighbor over outsourced encrypted medical images.

Specifically, in the paper, we discuss the problem of exact nearest neighbor search over encrypted medical images and propose a secure and efficient solution. Our scheme supports dynamic updates. It allows data users to easily add or delete medical images whenever necessary.

In the next section, we will introduce related work on nearest neighbor search. In Section III, we discuss relevant background materials including the system model and the design goals. The proposed scheme is explained in Section IV. Our security and performance analysis are outlined in Section V, and in the last section we conclude this paper.

2. RELATED WORK

Nearest neighbor search, first studied by Knuth [10] in 1973, has been the focus of ongoing research. In the literature, there are a large number of methods relating to the query process over encrypted data, such as searchable encryption [11]. In 2006, Curtmola et al. presented a new searchable symmetric encryption (SSE) construction to resist chosen keyword attacks [12]. Kamara et al. proposed a dynamic SSE scheme and a new security framework to optimize dynamic operations [13]. In 2013, Kamara et al. proposed a new dynamic SSE scheme, which is designed to support parallelizable search [14]. However, these schemes focus on keyword search, rather than the nearest neighbor search. We refer interested readers to [15] for a survey of existing SSE schemes.

One could also use sequential scan (brute-force search), which sequentially calculates and compares the Euclidean distance between the query item and every record in the encrypted database. However, this scheme needs significant time and space requirements, which are proportional to the number and dimension of the data. Thus, such a scheme is not appropriate for dealing with large-scale and high-dimensional data despite the existence of methods to minimize the costs. In 2009, Wong et al. [16] suggested searching for nearest neighbors over encrypted data using the asymmetric scalar product preserving encryption (ASPE) scheme. However, using the scheme requires linear search time relating to the number of data records. Two years later in 2011, Hu et al. [17] introduced a scheme with tree-based data structures and ASPE, which results in faster search time. Unfortunately, in this scheme the client who wishes to execute queries needs to perform numerous interactions with the server. Also, the need to maintain a local index will incur (significant) local storage costs. In 2006, Zhu et al. [18] designed a safer

scheme which can resist potential attacks of cloud server and avoid key-sharing. Zhu et al. [19] in 2017 proposed an efficient scheme for k-nearest neighbor search, which enhances the protection of the decryption key and reduces the burden on data owners.

There are a number of efficient schemes to find the approximate nearest neighbor, designed to improve efficiencyin space and time at the cost of accuracy. The scheme based onLocality Sensitive Hashing (LSH) is a well-known and effective method that solves the nearest neighbor query problem in high-dimensional space. The LSH scheme [20] embeds data in low-dimensional subspaces and utilizes hash tables to improve efficiency. In 2004, Datar et al. [21] provided a basic LSH based on the original LSH scheme, which exploited the property of p-stable distribution to generalize theLSH method from the Hamming space to the Euclidean space. This scheme, however, incurs significant overhead in space. In 2007, Andoni et al. [22] introduced the Leech lattice into the LSH scheme of [20], which reduces the query time and memory consumption. Hashing algorithms are useful to deal withhigh-dimensional and large-scale data, but they require additional cost when implemented in the exact nearest neighbor search. Due to the need for accuracy in the healthcare industry,these LSH-based schemes are not suitable for solving the medical images problems. Researchers have also proposed methods using tree-based data structures to efficiently find the nearest neighbor in the plaintext domain. As early as 1975, Bentley [23] proposed KD-tree and used this special data structure to store information that can be retrieved by associative searches. Many query types can be efficiently handled by this single data structure. Threedecades later, Jagadish et al. [24] presented an efficient B+-tree structure to address the K-nearest neighbor (KNN) search problem involving high-dimensional data. Other typical examples include R-tree variations [25], [26], [27] and Covertree [28]. These schemes, which are based on special data structures can reduce the search time cost because of the tree-structured organization of data. However, such data structures require large preprocess time and memory space, and so they are not appropriate for high-dimensional andlarge-scale data.

In 2012, Ahn et al. [27] proposed a nearest neighbor search algorithm for plaintext. This algorithm reduces dimensions of data points by embedding them in a low-dimensional space. Non-nearest neighbors are eliminated via a comparison of thedistances in this space. Due to this property, this algorithm issuitable for processing high-dimensional and large-scale data.

In particular, unlike most existing algorithms, this method canobtain the exact nearest neighbor rather than an approximate one. The beauty of this method is simplicity, in the sense of simple preprocessing without involving complex data structures.

In summary, most of the existing techniques have limitations and are not applicable to the healthcare industry's medical imaging problem. However, we also observe that the algorithm proposed by Ahn et al. [29] can simultaneously ensure both efficiency and accuracy. Based on this algorithm, we present an efficient scheme to search for the exact nearest neighbor in an outsourced medical image database. In our scheme, we compute the lower bound of Euclidean distance that is related to the mean and standard deviation of data. It is not necessary to load all data to compute the Euclidean distances in the original high-dimensional space. A large number of the nearest neighbor candidates can be eliminated by checking the lower bound.

3. PROBLEM FORMULATION

We will now introduce relevant notation (see Table I) and secure primitives [30], [31] used in our scheme, prior to describing the composition of the system model and design goals.

A. Preliminaries

Now, we introduce the secure primitives and generic protocols used in constructing our proposed scheme. These protocols are designed for a two-party semi-honest environment.

1) Paillier Cryptosystem

In the Paillier cryptosystem [30], given any two plaintexts $a, b \in \mathbb{Z}_N$, the scheme has the following features:

- **Homomorphic Addition.**
  $E_{pk}(a+b) \leftarrow E_{pk}(a) * E_{pk}(b) \bmod N^2$;
- **Homomorphic Multiplication.**
  $E_{pk}(a*b) \leftarrow E_{pk}(a)^b \bmod N^2$;
- **Semantic Security.** The Paillier cryptosystem is semantically secure [32]. Briefly, adversaries cannot infer any additional information about the plaintext from any given set of ciphertexts.

In this paper, we assume that data owner encrypts data of interest using the Paillier cryptosystem prior to outsourcing (e.g. to the cloud). For simplicity, we omit mod N^2 term during homomorphic operations for the rest of this paper.

2) Secure Multiplication (SM) Protocol

We assume that C_1 has private input $(E_{pk}(a), E_{pk}(b))$, and C_2 owns the secret key *sk*. The secure multiplication (SM) protocol is designed to obtain the encrypted result of $a*b$, i.e., $E_{pk}(a*b)$, which will be used as the output to C_1. In this protocol, C_1 and C_2 cannot obtain any private information regarding a and b. For any given $a, b \in \mathbb{Z}_N$:

$$a*b = (a + r_a) * (b + r_b) - a * r_b - b * r_a - r_a * r_b \quad (1)$$

All arithmetic operations are processed under $\mathbb{Z}_N$.

Algorithm 1 describes the SM protocol. In addition, for any given $x \in \mathbb{Z}_N$, "*N*-x" is equivalent to "-x" under $\mathbb{Z}_N$.

TABLE I
NOTATIONS

Notation	Description
sk, *pk*	Secret and public key
E_{pk}	Encryption function via the public key *pk*
E_{sk}	Decryption function via the secret key *sk*
C_1	Cloud server with the encrypted database
C_2	Cloud server owns the secret key *sk*
M	Medical image database, $M = \langle M_1, \ldots, M_n \rangle$
ID	Identifier of record in *M*
d	Dimension of a record in *M*.
Q	Query image, $Q = \langle Q_1, \ldots, Q_d \rangle$
μ	The mean of a medical image
σ	The standard deviation of a medical image

Algorithm 1 SM($E_{pk}(a)$, $E_{pk}(b)$)

Input:
$E_{pk}(a)$: the encrypted data a
$E_{pk}(b)$: the encrypted data b

Output:
$E_{pk}(a*b)$: the encrypted result of a multiplied b

1: C_1 picks two random numbers $r_a, r_b \in \mathbb{Z}_N$
2: C_1 computes $a' \leftarrow E_{pk}(a) * E_{pk}(r_a)$, 3: $b' \leftarrow E_{pk}(b) * E_{pk}(r_b)$
4: then sends a', b' to C_2
5: C_2 decrypts $h_a \leftarrow D_{sk}(a')$, $h_b \leftarrow D_{sk}(b')$
6: C_2 computes $h \leftarrow h_a * h_b \bmod N$, $h' \leftarrow E_{pk}(h)$
7: then sends h' to C_1
8: C_1 computes $H \leftarrow h' * E_{pk}(a)^{N-r_b}$
9: $H' \leftarrow H * E_{pk}(b)^{N-r_a}$
10: C_1 computes $E_{pk}(a * b) \leftarrow H' * E_{pk}(r_a * r_b)^{N-1}$

Algorithm 2

SSEDLB($\mathcal{E}_{pk}(\mathcal{Q})$, $\mathcal{E}_{pk}(\mathcal{Q}_x)$, $\mathcal{E}_{pk}(\mathcal{Q}_y)$, $\mathcal{E}_{pk}(\mathcal{P}_x)$, $\mathcal{E}_{pk}(\mathcal{P}_y)$)

Input:

$\mathcal{E}_{pk}(\mathcal{Q})$, $\mathcal{E}_{pk}(\mathcal{Q}_x)$, $\mathcal{E}_{pk}(\mathcal{Q}_y)$, $\mathcal{E}_{pk}(\mathcal{P}_x)$, $\mathcal{E}_{pk}(\mathcal{P}_y)$

Output:

$\mathcal{E}_{pk}(\mathcal{D}(\mathcal{Q}, \mathcal{P}))$: the encrypted lower bound of the squared Euclidean distance between X and Y

1: C_1 computes $\mathcal{E}_{pk}(\mathcal{Q}_x - \mathcal{Q}_y) \leftarrow \mathcal{E}_{pk}(\mathcal{Q}_x) * \mathcal{E}_{pk}(\mathcal{Q}_y)^{N-1}$

2: $\mathcal{E}_{pk}(\mathcal{Q}_x - \mathcal{Q}_y) \leftarrow \mathcal{E}_{pk}(\mathcal{Q}_x) * \mathcal{E}_{pk}(\mathcal{Q}_y)^{N-1}$

3: C_1 and C_2 compute $\mathcal{E}_{pk}((\mathcal{Q}_x - \mathcal{Q}_y)^2)$ and $\mathcal{E}_{pk}((\mathcal{P}_x - \mathcal{P}_y)^2)$ using the SM protocol

3: C_1 computes $\mathcal{E}_{pk}((\mathcal{Q}_x - \mathcal{Q}_y)^2 + (\mathcal{P}_x - \mathcal{P}_y)^2) \leftarrow \mathcal{E}_{pk}((\mathcal{Q}_x - \mathcal{Q}_y)^2) * \mathcal{E}_{pk}((\mathcal{P}_x - \mathcal{P}_y)^2)$

4: C_1 and C_2 compute $\mathcal{E}_{pk}(\mathcal{D} * ((\mathcal{Q}_x - \mathcal{Q}_y)^2 + (\mathcal{P}_x - \mathcal{P}_y)^2))$

3) Secure Squared Euclidean Distance (SSED) Protocol

Consider C_1 has two encrypted vectors $(\mathcal{E}_{pk}(\mathcal{Q}), \mathcal{E}_{pk}(\mathcal{P}))$, and C_2 has the secret key sk. Note that, X and Y are both d-dimensional vectors, i.e., $\mathcal{E}_{pk}(\mathcal{Q}) = \langle \mathcal{E}_{pk}(\mathcal{Q}_1), \ldots, \mathcal{E}_{pk}(\mathcal{Q}_d) \rangle$ and $\mathcal{E}_{pk}(\mathcal{P}) = \langle \mathcal{E}_{pk}(\mathcal{P}_1), \ldots, \mathcal{E}_{pk}(\mathcal{P}_d) \rangle$. The goal of SSED is to securely calculate $\mathcal{E}_{pk}(|\mathcal{Q} - \mathcal{P}|^2)$, where $|\mathcal{Q} - \mathcal{P}|$ represents the Euclidean distance between X and Y. In the protocol, C_1 and C_2 cannot obtain any private information about X and Y. The prime idea of SSED is based on the following equation:

$$|\mathcal{Q} - \mathcal{P}|^2 = \sum_{i}^{d} (\mathcal{Q}_i - \mathcal{P}_i)^2 \quad (2)$$

For $1 \le \mathcal{i} \le \mathcal{d}$, C_1 computes $\mathcal{E}_{pk}(\mathcal{Q}_i - \mathcal{P}_i)$ using the homomorphic properties. Then, C_1 and C_2 cooperatively compute $\mathcal{E}_{pk}((\mathcal{Q}_i - \mathcal{P}_i)^2)$ using the SM protocol, for $1 \le \mathcal{i} \le \mathcal{d}$. Note that only C_1 knows the outputs of SM. Based on Equation 2, C_1 computes $\mathcal{E}_{pk}(|\mathcal{Q} - \mathcal{P}|^2)$ locally by applying homomorphic properties to $\mathcal{E}_{pk}((\mathcal{Q}_i - \mathcal{P}_i)^2)$.

4. Secure Squared Euclidean Distance Lower Bound (SSEDLB) Protocol

We assume that C_1 has the encrypted statistical information, including $(E_{pk}(d), E_{pk}(\mu_x), E_{pk}(\sigma_x), E_{pk}(\mu_y), E_{pk}(\sigma_y))$, and C_2 has sk, where d denotes the dimension of data, and μ_x, σ_x, μ_y, σ_y respectively are the mean and standard deviation of d-dimensional data points and the query point. The main goal of SSEDLB is to calculate the lower bound of the squared Euclidean distance between X and Y, and this result is in encrypted form. In this protocol, no sensitive information pertaining to the statistical information is revealed to C_1 and C_2. SSEDLB can be described as follows, and Algorithm 2.

$$\mathcal{D}(\mathcal{Q}, \mathcal{P}) = \mathcal{D}((\mathcal{Q}_x - \mathcal{Q}_y)^2 + (\mathcal{P}_x - \mathcal{P}_y)^2).$$

B. System Model

As shown in Fig.1, there exists four entities in our scheme, namely: a data owner (e.g., hospital), multiple data users (e.g., doctors in different regions), and two semi-honest cloud servers. Note that data users are authorized by the data owner.

The data owner Alice owns a medical image database M, which she would like to outsource to the cloud server. Due to the sensitivity and privacy of medical images, Alice encrypts these images as well as computing and

encrypting the mean and standard deviation for each image in the database. In order to process the query, Alice outsources the encrypted calculated results and database to the cloud server.

A legitimate data user Bob can upload a medical image to the cloud server and query the exact nearest neighbor (i.e., $Q = \langle q_1, \ldots, q_d \rangle$). To ensure the privacy of the query, Bob encrypts Q with the Paillier public key *pk*. Additionally, Bob computes the mean and standard deviation of Q, denoted by μ_q and σ_q. Then, he sends his encrypted query and related information after encrypting *d*, μ_q, and σ_q.

In our scheme, we use two cloud servers to securely and efficiently execute related calculations. Both cloud servers areassumed to be "honest-but-curious." Specifically, " Honest" represents that the cloud servers follow the rules of the protocol and our program honestly, whereas "curious" represents that the cloud server may try to analyze the data flow and execution of the protocol to gain additional information.

C. Security Guarantee

Our scheme achieves semantic security under quantifiable leakage functions. In other words, we can formally define theviews of the cloud server in stateful leakage functions. Specifically, the cloud servers only know the information defined in leakage functions and no other valid knowledge

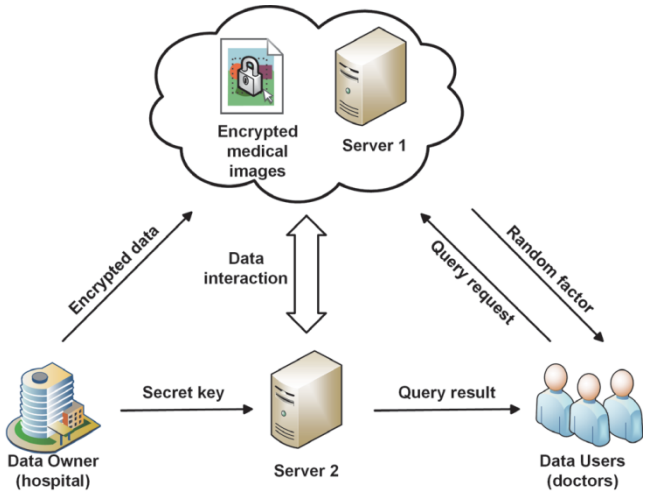

Fig. 1. Architecture for searching over encrypted cloud data

when a polynomial number of adaptive queries are executed. We define three stateful leakage functions for the views of theencrypted database, the query record, and the access pattern. When the encrypted database $E_{pk}(D)$ is outsourced, the serverwill know its size and dimension. The corresponding leakage function L_1 is defined as follows:

$$L_1(E_{pk}(D)) = (n, m, |D|)$$

where n is the size of the medical image database, m is the dimension of a medical image and $|D|$ is the bit lengths of encrypted data.

When the query request is submitted, the cloud server can seethe dimension of the encrypted query image. The leakage function L_2 is defined as:

$$L_2(\text{pk}, E_{pk}(Q), E_{pk}(\mu_q), E_{pk}(\sigma_q), E_{pk}(Q)) = (E_{pk}(\mu_q), E_{pk}(\sigma_q), E_{pk}(Q))$$

where $E_{pk}(Q)$ is the encrypted query image, $E_{pk}(\mu_q)$ and these images as well as computing and encrypting the mean and standard deviation for each image in the database. In order to

pk

$(E_{pk}(\mu_q)$ are the encrypted mean and standard deviation of Q.

process the query, Alice outsources the encrypted calculated results and database to the cloud server.
A legitimate data user Bob can upload a medical image to the cloud server and query the exact nearest neighbor (i.e., $\mathbb{Q}$ = $\langle \mathbb{Q}_1, \ldots, \mathbb{Q}_d \rangle$). To ensure the privacy of the query, Bob encrypts $\mathbb{Q}$ with the Paillier public key *pk*. Additionally, Bob computes the mean and standard deviation of $\mathbb{Q}$, denoted by $\mathbb{Q}_q$ and $\mathbb{Q}_q$. Then, he sends his encrypted query and related information after encrypting *d*, $\mathbb{Q}_q$, and $\mathbb{Q}_q$.
In our scheme, we use two cloud servers to securely and efficiently execute related calculations. Both cloud servers areassumed to be "honest-but-curious." Specifically, " Honest" represents that the cloud servers follow the rules of the protocol and our program honestly, whereas "curious" represents that the cloud server may try to analyze the data flow and execution of the protocol to gain additional information.

D. Security Guarantee

Our scheme achieves semantic security under quantifiable leakage functions. In other words, we can formally define theviews of the cloud server in stateful leakage functions. Specifically, the cloud servers only know the information defined in leakage functions and no other valid knowledge

By executing the Algorithm 3, the cloud server will find out the exact nearest neighbor of query $\mathbb{Q}$. The leakage function isdefined as:
$\mathbb{Q}_3(\mathbb{Q}\mathbb{Q}\mathbb{Q}\mathbb{Q}\mathbb{Q}\mathbb{Q}) = (\mathbb{Q}\mathbb{Q}, \mathbb{Q}\mathbb{Q}\mathbb{Q}\mathbb{Q}\mathbb{Q})$
where *ID* is corresponding with the exact nearest neighbor ofquery $\mathbb{Q}$, $\mathbb{Q}\mathbb{Q}\mathbb{Q}\mathbb{Q}\mathbb{Q}$ is the encrypted image.
Based on the above three stateful leakage functions, we givea formal security definition for our scheme. In simple, a probabilistic polynomial time simulator $\mathbb{Q}$ can simulate the query process and get a result, which are indistinguishable with the real algorithm.

Theorem 1. Our scheme is semantic secure with leakage functions ($\mathbb{Q}_1$, $\mathbb{Q}_2$, $\mathbb{Q}_3$) in the random oracle model if symmetric encryption and Paillier cryptosystem are IND-CPA security.
The proof will be discussed in section IV-C and it shows that the server only can know the above defined leakage and no other information else.

E. Design Goals

To meet the requirements of Healthcare Industry 4.0 and enable an exact nearest neighbor search over encrypted medical images using the above system model, our scheme should satisfy the following goals:

1) Accuracy: To meet the need for accuracy in the healthcare industry, the proposed scheme achieves a precise nearest neighbor search instead of an approximation.
2) Security: Healthcare Industry 4.0 requires high security and privacy. Our scheme is designed to ensure the confidentiality of all related medical images. To ensure query privacy, the database and query need to be encrypted before sending to the cloud server. The cloudservers can infer nothing useful when analyzing the dataflow in the query process.
3) Efficiency: Efficiency is a remarkable feature of Healthcare Industry 4.0. Our scheme has significantly less computational expense and resource consumption than existing schemes. In our scheme, we do not need to compute the Euclidean distance for all data, and the query process incurs low computational overhead on the user.
4) Dynamic: Our scheme supports dynamic updates. Data users can easily add or delete medical images that have been outsourced to the cloud server whenever necessary.

4. THE PROPOSED SCHEME

In this section, we introduce our scheme using the previously mentioned protocols as building blocks. After introducing thebasic scheme, we will give an additional explanation ondynamics and security.

A. Basic Scheme

Let $x=(\square_1,\ldots,\square_{\square})^T$ be a d-dimensional vector. It is easy to compute the mean and variance of the elements in $\square \in \square$, and they are given by $\mu_x=\frac{1}{d}\sum^{d}\square_i$ and $\square^2=\frac{1}{d}\sum^{d}(\square_i-\square_x)^2$ respectively. When another d-dimensional vector $y=(\square_1,\ldots,\square_d)^T$ is given, we can easily compute $\square_y$ and $\square^2$. Firstly, we make adefinition of $LB(x, y)$ as follows:

$$\square\square(\square,\square)=\square((\square_x-\square_y)^2+(\square_x-\square_y)^2). \quad (1)$$

$\square\square(\square,\square)$ is the lower bound of the squared Euclidean distance between x and y, which is denoted by $\square\square\square\square(\square,\square)^2$, as shown in Lemma 1:

Lemma 1. $\square\square\square(\square,\square)^2 \geq \square((\square_x-\square_y)^2+(\square_x-\square_y)^2)$

We will directly utilize this lemma, the detailed proof for which you can find in [29]. Suppose we have a dataset $\square =$ $\{x_1,\ldots, x_n\}$. We can calculate and store μ_x and σ_x for all $x \in X$ in O(nd) time and O(n) space. When a query y comes, we can get μ_y and σ_y in O(d) time, and $LB(x, y)$ is calculated in O(1) time for any $x \in X$. The algorithm proposed in [29] uses the lower bound to dramatically reduce the calculation of Euclidean distance. Specifically, we can eliminate lots of nearest neighbor candidates using simple computations and comparisons. All $\square \in \square$ and query y can be embedded in the two–dimensional space based on their mean and variance of elements. We can easily calculate the lower bounds of the squared Euclidean distance. Then we can reject $\square_i$ if $LB(\square_i, y)$is larger than the squared Euclidean distance to the current nearest neighbor from y. Using this algorithm, many data pointsare eliminated in constant time rather than linear time. When the elements are high-dimensional data, the computational cost reduction will be significant.

As mentioned earlier, we assume that Alice owns a medicalimage database M. Note that these medical images can be x-ray

Algorithm 3

Secure Nearest Neighbor Search ($\square_{pk}(\square)$, $\square_{pk}(\square)$)

Input:

$\square_{pk}(\square)$: the encrypted database

$\square_{pk}(\square)$: the encrypted query

Output:

$\square_{min}$: the nearest neighbor of Q

1: Bob sends $\square_{pk}(\square_q)$, $\square_{pk}(\square_q)$ and $\square_{pk}(\square)$ to C_1

2: C_1 do

3: picks a seed $\square_{pk}(\square_{min})$ from $\square_{pk}(\square)$.

4: computes $\square_{pk}(\square\square\square\square\square\square\square\square)$ ← $\square\square\square\square(\square_{pk}(\square_{min}), \square_{pk}(\square))$

5: **for** $\square = 1 \rightarrow \square$ **do**

6: $\square_{pk}(\square\square_i)$ ←

$\square\square\square\square\square\square\square(\square_{pk}(\square), \square_{pk}(\square_{m_i}), \square_{pk}(\square_{m_i}), \square_{pk}(\square_q), \square_{pk}(\square_q))$

7: **end for**

8: sends $\square_{pk}(\square\square\square\square\square\square\square\square)$, $\square_{pk}(\square\square)$ and $\square_k(\square\square)$ to C_2

.9: C_2 do

10: decrypts $\square_{pk}(\square\square\square\square\square\square\square\square)$and $\square_{pk}(\square\square)$

11: **for** $\square = 1 \rightarrow \square$ **do:**

12: **if** $\square_{sk}(\square\square_i) \geq \square_{sk}(\square\square\square\square\square\square\square\square\square)$ **then**

13: **continue**

end if
if $\mathcal{D}_{sk}(\mathcal{E}_i) < \mathcal{D}_{sk}(\mathcal{E}_{min})$ **then**
  sends $\mathcal{E}_k(\mathcal{E})$ to C_1
  C_1 and C_2 compute $\mathcal{E}_{pk}(\mathcal{D}_{sk}^2(\mathcal{E}_i, \mathcal{E}))$
  C_1 sends $\mathcal{E}_{pk}(\mathcal{D}_{sk}^2(\mathcal{E}_j, \mathcal{E}))$ to C_2
  C_2 do
  decrypts $\mathcal{D}_{sk}(\mathcal{D}_{sk}^2(\mathcal{E}_j, \mathcal{E}))$
  if $\mathcal{D}_{sk}(\mathcal{D}_{sk}^2(\mathcal{E}_j, \mathcal{E})) \geq \mathcal{D}_{sk}(\mathcal{E}_{min})$ **then**
    continue
  end if
  if $\mathcal{D}_{sk}(\mathcal{D}_{sk}^2(\mathcal{E}_j, \mathcal{E})) < \mathcal{D}_{sk}(\mathcal{E}_{min})$ **then**
    $\mathcal{E}_k(\mathcal{E}_{min}) \leftarrow \mathcal{E}_k(\mathcal{E})$
    $\mathcal{D}_{sk}(\mathcal{E}_{min}) \leftarrow \mathcal{D}_{sk}(\mathcal{D}_{sk}^2(\mathcal{E}_j, \mathcal{E}))$
  end if
 end if
end for
C_2 sends $\mathcal{E}_k(\mathcal{E}_{min})$ to
C_1 C_1 do
receives $\mathcal{E}_k(\mathcal{E}_{min})$ and find $\mathcal{E}_{pk}(\mathcal{E}_{min})$
picks a random number $\mathcal{E} \in \mathcal{D}_N$ and send r to Bob
computes $\mathcal{E}_{pk}(\mathcal{E}_{min} + \mathcal{E})$ and send it to C_2
C_2 computes and sends $\mathcal{D}_{sk}(\mathcal{E}_{min} + \mathcal{E})$ to Bob
Bob computes $\mathcal{E}_{min}$ via $\mathcal{D}_{sk}(\mathcal{E}_{min} + \mathcal{E})$ and r

images, MRI (Magnetic Resonance Image), CT (Computed Tomography) images, and so on. Initially, Alice computes themean and standard deviation for each image in the database, expressed as $\square_i$ and $\square_i$ for $1 \le i \le \square$. Then Alice encrypts the $\square_i$and $\square_i$ for $1 \le \square \le \square$ with Paillier cryptosystem, denoted by $\square_{pk}(\square_i)$ and $\square_{pk}(\square_i)$. After that, Alice encrypts her database attribute-wise. In fact, the database is encrypted attribute-wise, so the dimension *d* is public. Then the cloud server can compute $E_{pk}(d)$. Let the encrypted database be expressed as $E_{pk}(M)$. Theindex is built using the $E_k(ID)$, where k is a symmetric encryption key. The owner outsources $E_k(ID)$, $E_{pk}(\mu_i)$, $E_{pk}(\sigma_i)$, and $E_{pk}(M)$, and subsequent query process to the cloud.

In our proposed model, there exist two honest-but-curious cloud servers, denoted by C_1 and C_2, and together they form a federated cloud. Then, we outsource the encrypted database to C_1 and C_2 keeps the secret key *sk*. In fact, it is not a fresh assumption and has been widely used in the related research domains [33]. The premise of this assumption is as follows. In general, the cloud service providers are famous and formal IT companies, such as Amazon and Google. Therefore, in case damaging their reputation and effecting revenue, a collusion between them is almost impossible to happen.

In this situation, the owner Alice outsources the encrypted image database $\square_{pk}(\square)$ to C_1 and sends the secret key of the Paillier cryptosystem to C_2. The intent of our scheme is to efficiently and securely retrieve the record which is closest to the query in the encrypted database. In brief, now a legitimate user wants to search out the exact nearest neighbor of his query record $\square = \langle \square_1, \dots, \square_d \rangle$ based on the encrypted medical image database $\square_{pk}(\square)$ stored in C_1. Firstly, the user sends his encrypted query information to C_1. After this, C_1 and C_2 utilize the above three protocols to securely search the nearest neighbor of the input query $\square$. In the end, only the user will know the exact nearest neighbor to $\square$. The key steps of our algorithm are shown in Algorithm 3.

Now a legitimate user Bob wants to search for an image that is the most similar to his query $\square$ based on $\square_{pk}(\square)$ in C_1. First of all, Bob computes the mean and standard deviation of $\square$, denoted by $\square_q$ and $\square_q$. Then he encrypts $\square_q$ and $\square_q$ with the Paillier public key, expressed as $\square_{pk}(\square_q)$, and $\square_{pk}(\square_q)$. After that, Bob encrypts the query image $\square$ as $\square_{pk}(\square)$ and sends $\square_{pk}(\square_q)$, $\square_{pk}(\square_q)$, and $\square_{pk}(\square)$ to the cloud server C_1.

After C_1 receives the query request and related informationfrom Bob, the cloud servers will execute the query process. The interaction between the two servers is shown in Fig. 2.

Firstly C_1 picks an encrypted image in $\square_{pk}(\square)$ as the seed $\square_{pk}(\square_{min})$ and its *ID* is $\square_k(\square\square\square)$. Then C_1 and C_2 jointly compute the secure squared Euclidean distance between $\square_{pk}(\square_{min})$ and the query $\square_{pk}(\square)$, denoted by $\square_{pk}(\square\square\square\square\square\square\square\square)$.

Next, for each encrypted record, the servers securely calculate the lower bound of the squared Euclidean distance between this record and the query $\square_{pk}(\square)$ using the SSEDLB protocol. We symbolize these encrypted values as $\square_{pk}(\square\square_i)$. Notice that each image $\square_i$ corresponds to $\square_{pk}(\square\square_i)$. However, to streamline thenotation, we will delete the subscript and use $\square_{pk}(\square\square)$ to represent all $\square_{pk}(\square\square_i)$. Finally, C_1 sends $\square_{pk}(\square\square)$ and $\square_{pk}(\square\square\square\square\square\square\square\square)$ as well as the corresponding $\square_k(\square\square)$ to the server C_2.

Upon receiving $\square_{pk}(\square\square\square\square\square\square\square\square)$ and $\square_{pk}(\square\square)$ from C_1, C_2 decrypts them with the secret key *sk* of the Paillier cryptosystem, denoted by $\square_{sk}(\square\square\square\square\square\square\square\square)$ and $\square_{sk}(\square\square)$. Afterward, C_2 makes comparisons between $\square_{sk}(\square\square\square\square\square\square\square\square)$ and $\square_{sk}(\square\square_i)$ for $1 \le \square \le \square$. If $_{sk}(\square\square_i)$ value is bigger than $\square_{sk}(\square\square\square\square\square\square\square\square)$, we can know that the $\square\square\square\square^2(\square_i, \square)$ is larger than the $\square\square\square\square^2(\square_{min}, \square)$. Therefore, the relevant image $\square_i$ cannot bethe nearest neighbor to the query. Else, two servers jointly

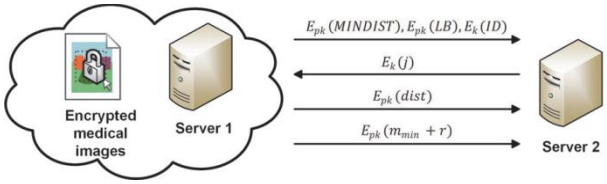

Fig. 2. Interaction between two cloud servers

calculate the secure squared Euclidean distance between $\square_{pk}(\square_i)$ and the query $\square_{pk}(\square)$, denoted by $\square_{pk}(\square\square\square\square^2(\square_i, \square))$. C_1 sends $\square_{pk}(\square\square\square\square^2(\square_i, \square))$ to C_2, then C_2 decrypts it and compares $\square_{sk}(\square\square\square\square\square\square\square\square)$ and $\square_{sk}(\square\square\square\square)$. If $\square_{sk}(\square\square\square\square)$ is smaller than $\square_{sk}(\square\square\square\square\square\square\square\square)$, C_2 will update the value of $\square_{sk}(\square\square\square\square\square\square\square\square)$ using $\square_{sk}(\square\square\square\square)$ and modify the *ID*

$\square_k(\square\square\square)$. If not, C_2 has no operations. In our algorithm, we check each $\square_{sk}(\square\square)$ and update the nearest neighbor we have found according to the situation. After traversing all records in the database, we can discover the minimum value of the secure squared Euclidean distance between $\square_{pk}(\square)$ and the data item in the outsourced dataset. The corresponding data item is the nearest neighbor of query $\square$ whose *ID* is $\square_k(\square\square\square)$. At the same time, C_2 sends the corresponding *ID* $\square_k(\square\square\square)$ to C_1.

C_1 receives the encrypted *ID* $\square_k(\square\square\square)$ and searches out the corresponding encrypted data $\square_{pk}(\square_{min})$. Now we get the exact nearest neighbor of our query $\square$, but it is in encrypted form. Next, C_1 picks a random number *r* in $\mathbb{Z}_N$ and sends *r* to the authorized user Bob through a secure channel. In addition, C_1 encrypts *r* and computes $\square_{pk}(\square_{min} + \square)$ using the homomorphic properties. At the end of our algorithm, C_1 sends $\square_{pk}(\square_{min} + \square)$ to C_2. Then C_2 decrypts it, denoted by $\square_{sk}(\square_{min} + \square)$, and securely sends $\square_{sk}(\square_{min} + \square)$ to Bob.

Bob now receives *r* and $\square_{min} + \square$, so he can easily compute to get $\square_{min}$, which is the exact nearest neighbor of his query $\square$.

B. Scalability

Our proposed scheme has good scalability performance. And the dynamic changes to the database have almost no impact on our algorithm. The scalability is mainly reflected in two aspects. On the one hand, the owner Alice wants to add one or more medical images to the database that has been encrypted and outsourced. In this case, Alice first computes the mean and standard deviation of these new images, then she encrypts and uploads these images and related means, standard deviations and *IDs* to the cloud server C_1.

On the other hand, if Alice wants to delete one or more images that are stored in C_1, she only needs to send the corresponding *IDs* to C_1 and the server will delete the relative records. The owner can easily modify his or her hosted database that is in the remote cloud server, and the addition or deletion of data will not affect the user experience.

C. Security Analysis

When the encrypted database is outsourced to the cloud servers, secure query protocols and algorithms need to preserve the secrecy and privacy of the outsourced database as well as the user's query at all times. At the same time, data access patterns should be hidden from the cloud. In our scheme, because of the encryption of query $\square$ and the semantic security of the Paillier cryptosystem, a user's input query $\square$ is well protected from other participants in the whole process.

In the SM protocol, all parameters are in encrypted form. The decryptions at line 5 of Algorithm 1 reveal two meaningless values because they have been confused by random numbers. Therefore, we know that this SM protocol is secure and it will not leak sensitive information. In addition, the SSED protocol and SSEDLB protocol are extended from the SM protocol which means that their security is based on the SM protocol security. Therefore, our three basic protocols are secure under our setting.

As for the security analysis of Algorithm 3, it is illustrated as follows. First, we should know that all parameters are transmitted through secure channels and the user Bob needs to be certified. So we need not consider illegal access and attacks during the transmission process. At line 10 of Algorithm 3, C_2 gets $\square_{sk}(\square\square\square\square\square\square\square\square)$ and $\square_{sk}(\square\square)$ in decrypted form. However, these values will not reveal information related to our sensitive medical images and the user's query, as these calculation results are weakly correlated with the original data. Also, C_2 can't derive the sensitive data. In fact, if one wants better security, they can make a small change in our algorithm. C_1 can add a random number $\square \in \mathbb{Z}_N$ to $\square_{pk}(\square\square\square\square\square\square\square\square)$ and $\square_{pk}(\square\square)$ (denoted by $\square_{pk}(\square\square\square\square\square\square\square\square + \square)$ and $\square_{pk}(\square\square + \square)$) before sending them to C_2. The same operation can apply to line 20 of Algorithm 3 to protect $\square_{pk}(\square\square\square\square)$. At the end of our algorithm, C_1 sends the exact nearest neighbor to C_2 after adding a random number interference, preventing C_2 from knowing the search result.

Now we provide the formal security proof for our scheme.

Definition 1. Given our scheme with three stateful leakage functions $(\square_1, \square_2, \square_3)$, a probabilistic polynomial time adversary $\square$ and a probabilistic polynomial time simulator $\square$. Then we define two probabilistic games $\square\square\square\square(\square)$ and $\square\square\square\square\square_{\square,\square}(\square)$ as follows:

$\square\square\square\square(\square)$: a challenger $\square$ owns public/private key pair and a symmetric key. The adversary $\square$ firstly selects a dataset *D* and asks $\square$ to construct the encrypted index. After that, $\square$ adaptively executes a polynomial number of secure queries. Finally, $\square$ returns a bit as the game's output.

$\square\square\square\square\square_{\square,\square}(\square)$: $\square$ selects *D*, and $\square$ generates a random index based on $\square_1$. Next $\square$ adaptively executes a polynomial number of secure queries. According to $\square_2$ and $\square_3$ of each query record, $\square$ generates the relevant result. Finally, $\square$ returns a bit as the game's output.

Our scheme is ($\mathcal{L}_1$, $\mathcal{L}_2$, $\mathcal{L}_3$) -semantic security if for all probabilistic polynomial time adversary $\mathcal{A}$ there exists a probabilistic polynomial time simulator $\mathcal{S}$ such that
$\Pr[\mathcal{A}(\mathcal{R}_{\mathcal{A}}) = 1] - \Pr[\mathcal{A}(\mathcal{S}_{\mathcal{A},\mathcal{S}}) = 1] \leq \mathcal{N}(\mathcal{K})$
where $\mathcal{N}(\mathcal{K})$is a negligible function in $\mathcal{K}$.

Proof of Theorem 1:

Proof. According to $\mathcal{L}_1$, the simulator $\mathcal{S}$ can generate a randomindex, which has the same format and size as the real encrypted index. Due to the semantic security of symmetric encryption and Paillier cryptosystem, the adversary $\mathcal{A}$ cannot distinguish between the random index and real index.
When executing the first query, $\mathcal{S}$ generates a query request firstly. Then a random oracle $\mathcal{L}_1$ selects data from the dataset. After that another random oracle $\mathcal{L}_2$ gets the result: *ID*. Note that this *ID* is exactly identical to the real result indicated in $\mathcal{L}_3$. Because of the IND-CPA security of Paillier cryptosystem, this query request is computationally indistinguishable from real query request. For the following queries, $\mathcal{Q}$, $\mathcal{L}_1$ and $\mathcal{L}_2$ get the results in the same way. And the results derived from the random index are the same as real results. In other words, $\mathcal{A}$ cannot differentiate simulated requests and results from real data.
Based on the above discussions, our proposed scheme obviously meets the security requirements. It protects the secrecy and privacy of data as well as the user's input query while simultaneously hiding data access patterns.

5. PERFORMANCE ANALYSIS

In this section, we evaluate the performance of our scheme under different parameter environments. We utilized the Paillier cryptosystem [30] to execute homomorphic encryption operation and implemented the proposed scheme in C. Our experiments were completed on a Linux machine with the following features:

- CPU: Intel(R) Core(TM) i7-7700 CPU @ 3.60GHz
- Physical Memory: 3.8G
- OS: Ubuntu 15.10 Linux 4.2.0-16-generic SMP x86_64 To evaluate our exact nearest neighbor search scheme, we used the LIDC-IDRI dataset. The LIDC-IDRI dataset [34] is areal medical image dataset with a total of 1018 CT images and we use part of these images to experiment. The size of the original image is 512*512. We list eight samples in Fig.4. In fact, we usually only care about specific areas of the medical images, such as the area where the lesion occurs. In this case, the rest of the image is pointless but consumes a lot of computational resources. Therefore in the practice application, we were able to remove those meaningless parts and only save apart of the original image. In our experiments, we intercept thecentral part of the original images to build a new dataset, where the size of the image is 256*256 as shown in Fig.5.

In our scheme, the data owner needs to execute a simple
preprocessing step, which is the calculation of the mean and standard deviation for each image in the database. Fig.3a shows the preprocessing time cost for the different dataset sizes. We can see that the preprocessing takes very little time and it willnot cause problems for users. In particular, Fig.3b shows that the time required to compute the lower bound is unrelated to data dimensions. Therefore our scheme is suitable for solvingthe high-dimensional medical images problems.
To evaluate the efficiency, we compared our scheme (FNN)with the Brute-Force search scheme (BF), which computes theEuclidean distance for all data items sequentially. Fig.3c shows the performance of two schemes for the exact nearest neighbor search on the original dataset and Fig.3d shows the results on the new 256*256 dataset. In the matter of the query processing
time, on both datasets the proposed scheme obviously outperforms the BF scheme. We can know that if the size of the image is reduced, the search time will be greatly reduced.
Table I and Table II present more details about the performance of our scheme. Obviously, the time required to calculate the lower bound is almost negligible. The reject ratio of candidates refers to the proportion of data that can be excluded by comparing the lower bound of the squared Euclidean distance. And the reject ratio directly determines the performance of our scheme. In the datasets used in our experiments, the average reject ratio of the FNN scheme can reach more than 37%, which saves a lot of time.

TABLE I
DETAILED EXPERIMENTAL RESULTS FOR FNN ON 512*512 DATASET

Numbers of data	100	200	300	400	500
LB time(ms)	549.7	1121.4	1650.8	2148.8	2674.4
Reject ratio(%)	39.0	38.5	42.0	40.3	40.8
Search time(h)	6.39	12.76	17.93	24.66	30.44

TABLE II
DETAILED EXPERIMENTAL RESULTS FOR FNN ON 265*256 DATASET

Numbers of data	100	200	300	400	500
LB time(ms)	554.9	1098.5	1646.4	2195.6	2745.1
Reject ratio(%)	31.0	31.0	35.7	38.3	42.4
Search time(h)	1.78	3.54	4.95	6.33	7.47

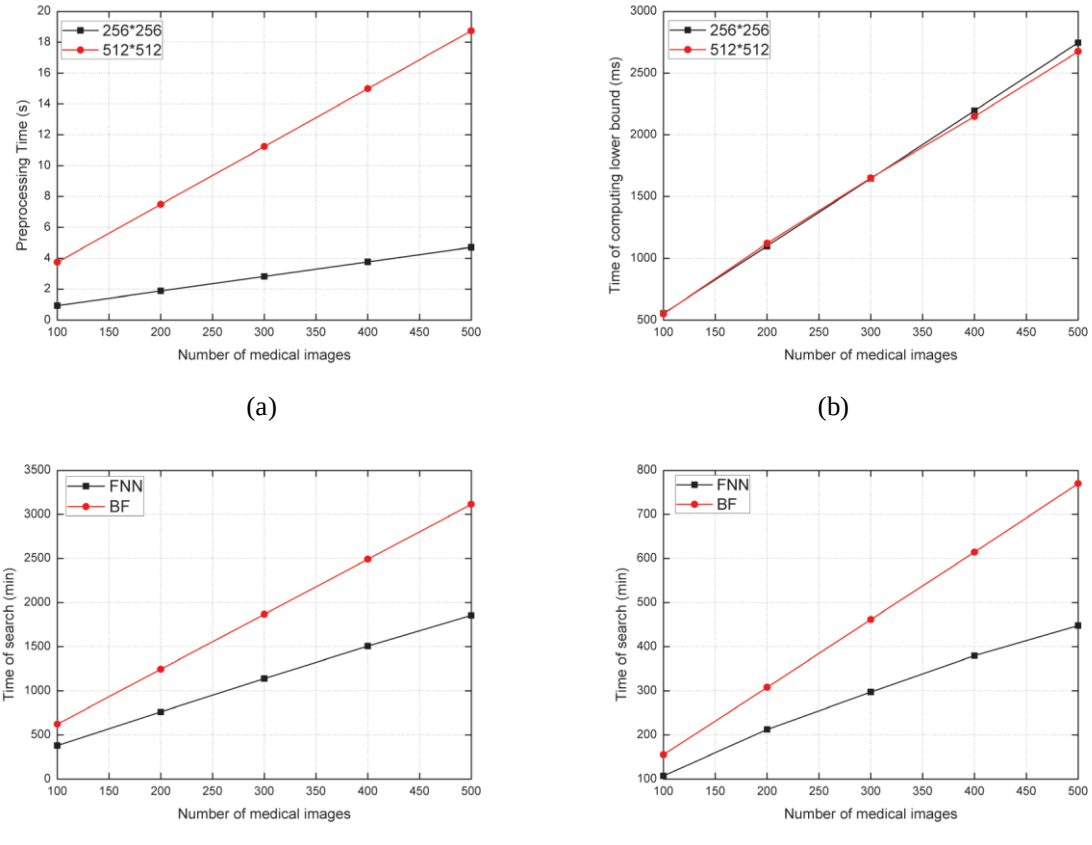

Fig. 3. Performance for the exact nearest neighbor search: (a) preprocessing time of FNN; (b) time of computing lower bound (c) query time on 512*512 dataset; (d) query time on 256*256 dataset

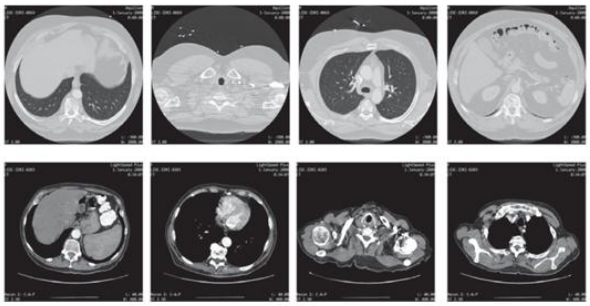

Fig. 4. Original images 512*512.

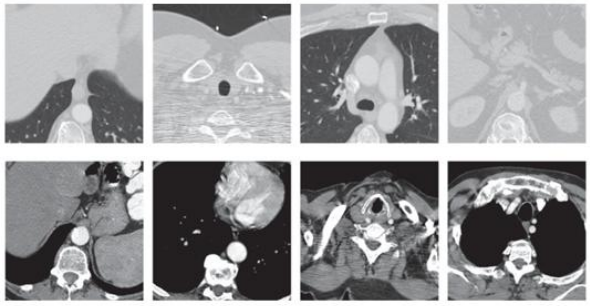

Fig. 5. Intercepted images 256*256.

6. CONCLUSION

Cloud-based electronic healthcare systems will be increasing popular, particularly due to the capability to share and access data in real-time across organizations (e.g., between medical practitioners and healthcare providers) and countries. One process becomes challenging, if not impractical.

In the paper, we presented a secure and efficient scheme to locate the exact nearest neighbor over encrypted medical images stored in the remote cloud server. For the purpose of rejecting candidate data points, our scheme securely computes the lower bound of the squared Euclidean distance between a data point in the database and the query submitted by a legitimate user. The performance of our scheme is evaluated using real-world medical images.

Future research includes finding a real-world healthcare organization to design and implement a prototype of our proposed approach. This will allow us to evaluate the real-world utility of the proposed system as well as its scalability in practice. In addition, it will also allow us to identify weaknesses / limitations, if any, that we are not awareof.

7. REFERENCES

[1]. J. Li, L. Huang, Y. Zhou, S. He, Z. Ming, "Computation partitioning for mobile cloud computing in big data environment," *IEEE Trans. Ind. Informat.*, vol. 13, no. 4, pp. 2009-2018, Feb. 2017.

[2]. K.-K. R. Choo, "Cloud computing: Challenges and future directions," *Trends & Issues in Crime and Criminal Justice*, vol. 400, no. 400, pp. 1– 6, Oct. 2010.

[3]. M. Pajic, R. Mangharam, O. Sokolsky, D. Arney, J. M. Goldman, and I. Lee, "Model-driven safety analysis of closed-loop medical systems," *IEEE Trans. Ind. Informat.*, vol. 10, no. 1, pp. 3–16, Feb. 2014.

[4]. B. Xu, L. D. Xu, H. Cai, C. Xie, J. Hu, and F. Bu, "Ubiquitous data accessing method in IoT-based information system for emergency medical services," *IEEE Trans. Ind. Informat.*, vol. 10, no. 2, pp. 1578–1586, May. 2014.

[5]. G. Yang *et al.*, "A health-IoT platform based on the integration of intelligent packaging, unobtrusive bio-sensor, and intelligent medicine box," *IEEE Trans. Ind. Informat.*, vol. 10, no. 4, pp. 2180–2191, Nov. 2014.

[6]. H. Huang, T. Gong, N. Ye, R. Wang, and Y. Dou, "Private and Secured Medical Data Transmission and Analysis for Wireless Sensing Healthcare System," *IEEE Trans. Ind. Informat.*, vol. 13, no.3 pp. 1227-1237, June. 2017.

[7]. M. Li, S. Yu, W. Lou, and Y. T. Hou, "Toward privacy-assured cloud data services with flexible search functionalities," in *Proc. ICDCSW. IEEE*, Macau, CHN, 2012, pp. 466–470.

[8]. P. Williams, R. Sion, and B. Carbunar, "Building castles out of mud: practical access pattern privacy and correctness on untrusted storage," in *Proc. CCS. ACM*, Alexandria, VA, USA, 2008, pp. 139–148.

[9]. M. S. Islam, M. Kuzu, and M. Kantarcioglu, "Access pattern disclosure on searchable encryption: Ramification, attack and mitigation," in *NDSS*, San Diego, CA, USA, 2012.

[10]. D. E. Knuth, "Sorting and searching," in *The art of computer programming*, vol. 3, Boston, USA: Addison-Wesley, 1973.

[11]. D. Song, D. Wagner, and A. Perrig, "Practical Techniques for Searches on Encrypted Data," in *Proc. of IEEE S&P*, DC, USA, 2000, pp. 44-55.

[12]. R. Curtmola, J. Garay, S. Kamara, and R. Ostrovsky, "Searchable symmetric encryption: improved definitions and efficient constructions," *J. Comput.Secur.*, vol. 19, no. 5, pp. 895-934, 2011.

[13]. S. Kamara, C. Papamanthou, and T. Roeder, "Dynamic Searchable Symmetric Encryption," in *Proc. of ACM CCS*, Raleigh, NC, USA, 2012, pp. 965–976.

[14]. S. Kamara, C. Papamanthou, "Parallel and dynamic searchable symmetric encryption," *Financial Cryptography and Data Security*, Springer Berlin Heidelberg, 2013, pp. 258-274.

[15]. G. S. Poh, J.–J. Chin, W.-C. Yau, K.-K. R. Choo, and M. S. Mohamad, "Searchable Symmetric Encryption: Designs and Challenges," ACM Comput. Surv. vol. 50, no. 3, pp. 40:1-40:37, 2017.

[16]. W. Wong, D. W. Cheung, B. Kao, and N. Mamoulis, "Secure KNN Computation on Encrypted Databases," in *Proc. ACM SIGMOD*, Providence, RI, USA, 2009, pp. 139–152.

[17]. H. Hu, J. Xu, C. Ren, and B. Choi, "Processing Private Queries over Untrusted Data Cloud through Privacy Homomorphism," in *Proc. IEEEICDE*, Hannover, NI, GER, 2011, pp. 601–612.

[18]. Y. Zhu, Z. Huang, and T. Takagi, "Secure and controllable k-NN query over encrypted cloud data with key confidentiality," *Journal of Parallel & Distributed Computing*, vol. 89, pp. 1-12, Mar. 2016.

[19]. L. Zhou, Y. Zhu, and A. Castiglione, "Efficient k -NN query over encrypted data in cloud with limited key-disclosure and

offline data owner," *Computers & Security*, vol. 69, pp. 84-96, Aug. 2017.
[20]. P. Indyk and R. Motwani, "Approximate nearest neighbors: Towards removing the curse of dimensionality," In *Proc. STOC*, Dallas, TX, USA, 1998, pp. 604– 613.
[21]. M. Datar, N. Immorlica, P. Indyk, and V. S. Mirrokni, "Locality sensitive hashing scheme based on p-stable distributions," in *Proc. SCG. ACM*, Brooklyn, NY, USA, 2004, pp. 253–262.
[22]. A. Andoni, P. Indyk, "Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions," *IEEE Symposium on Foundationsof Computer Science*, Berkeley, CA, USA, 2006, pp. 459-468.
[23]. J. L. Bentley, "Multidimensional binary search trees used for associative searching," *Communications of ACM*, vol. 18, no. 9, pp. 509–517, Sep.1975.
[24]. H. V. Jagadish, B. C. Ooi, K.-L. Tan, C. Yu, and R. Zhang, "iDistance: An adaptive B+-tree based indexing method for nearest neighbor search,"*ACM Trans. Database Syst.*, vol. 30, no. 2, pp. 364–397, June. 2005.
[25]. S. Arya, D. Mount, N. Netanyahu, R. Silverman, and A. Wu, "An optimal algorithm for approximate nearest neighbor searching fixed dimensions," *Journal of the ACM*, vol. 45, no. 6, 1998, pp. 891–923.
[26]. S. Berchtold, D. A. Keim, and H. P. Kriegel, "The x-tree: An index structure for high-dimensional data," in *Proc. VLDB Conf.*, Mumbai (Bombay), India, 1996, pp. 28– 39.
[27]. A. Guttman, "R-trees: A dynamic index structure for spatial searching," in *Proc. SIGMOD. ACM*, Boston, MA, USA, 1984, pp. 47–57.
[28]. A. Beygelzimer, S. Kakade, and J. Langford, "Cover trees for nearest neighbor," In *Proc. ICML*. ACM, Pittsburgh, PA, USA, 2006, pp. 97– 104.
[29]. Ahn H K, Han B, and Hwang Y, "A fast nearest neighbor search algorithm by nonlinear embedding," in *Proc. CVPR. IEEE*, Providence, RI, USA, 2012, pp. 3053-3060.
[30]. P. Paillier "Public-key cryptosystems based on composite degree residuosity classes," in *Proc. EUROCRYPT*, Prague, CZE, 1999, pp. 223-238.
[31]. Y. Elmehdwi, B. K. Samanthula, and W. Jiang, "Secure k-nearest neighbor query over encrypted data in outsourced environments," in *Proc. ICDE*, Chicago, IL, USA, 2014, pp. 664-675.
[32]. S. Goldwasser, S. Micali, and C. Rackoff. "The knowledge complexity of interactive proof systems." *Siam Journal on Computing*, vol. 18, no. 1, pp. 186-208. Feb. 1989.
[33]. S. Bugiel, S. Nurnberger, A.-R. Sadeghi, and T. Schneider, "Twin clouds: An architecture for secure cloud computing (extended abstract)," in *Workshop on Cryptography and Security in Clouds*, March 2011.
[34]. Armato III, Samuel G., McLennan, Geoffrey, Bidaut, Luc, McNitt-Gray, Michael F., Meyer, Charles R., Reeves, Anthony P., … Clarke, Laurence
[35]. P. (2015). Data From LIDC-IDRI. The Cancer Imaging Archive.http://doi.org/10.7937/K9/TCIA.2015.LO9QL9SX.