Secure Nearest Neighbor Search over Encrypted Data in Cloud Environment using Skip Graph

Upinder Kaur
Research Scholar
Dept. Computer Sci. & Applications,
Kurukshetra University, Kurukshetra, Haryana, India.

Pushpa R. Suri, PhD
(Retd) Professor
Dept. Computer Sci. & Applications,
Kurukshetra University, Kurukshetra, Haryana, India.

ABSTRACT
Nearest neighbor search is the most general search queries on large data sets, location base services, spatial databases, graph applications, etc. With the growth of cloud computing, the trend of outsourcing the sensitive data demands the fast and secure nearest neighbor solutions over the existing solutions. In our paper, propose a new secure and efficient nearest neighbor search for encrypted data with mOPE (mutable order preserving encryption). In our proposed model, use the probabilistic data structure skip graph for the efficient indexing. Then, encrypt the indexes using mOPE for efficient nearest neighbor search. With the thorough analysis of our scheme achieve a perfect balance between security and nearest neighbor search query compared to another scheme.

Keywords
Nearest Neighbor Search, Cloud Computing, Skip Graph.

1. INTRODUCTION
The nearest neighbor problem aims at finding the most similar point to a query point from the database. It is used in different applications like location-based services, data mining, multimedia databases, spatial data, etc. Several amounts of research have focused on efficiently nearest neighbor queries with minimum computational overhead. The large data set is managed locally with various known spatial indexes for approx. Nearest neighbor queries. But with the emergence of cloud computing, data owner outsourced their data into the cloud to remove the burden of a massive amount of data management in a cost effective manner. Cloud computing is a new paradigm that helps the user the metered services “pay as per use.” The result of the cloud environment is losing in controlling over the remote placement of data and became the security concern for data owner. Therefore the past research for nearest neighbor search is not efficient enough. Many researchers have proposed various methods to address these problems of security and privacy of nearest neighbor query. The general approach adopts by data owner is to encrypt the outsourced data.

Many research schemes have been proposed to enable functionality over the encrypted data over cloud environment, but these are not compatible with advance queries like nearest neighbor search. The authors [1-5] proposed some nearest neighbor search queries on encrypted data, but they still face some limitations. Some have proposed k NN queries are failing to support high dimensional data or building efficient k NN algorithm. The challenge here is to calculate the distance from the query point to the data point in encrypted data and compare the distance with the other data points to finally retrieve the minimum distance to the query point.

In this paper, our primary focus on the problem of nearest neighbor search in encrypted data in the cloud environment. There are existed cryptographic primitives themselves do not preserve the distance between the data after encryption. The Order preserving encryption allows performing various database functions over encrypted data. Here we are using the more efficient mOPE combined with our secure index skip graph for a more efficient solution for nearest neighbor search in encrypted data in a cloud environment.

2. RELATED WORK
The existing traditional searchable encryption schemes [5-10], cannot be directly processed as plain text. Various researchers proposed efficient schemes to support executing the query on encrypted data. Wong et al. [11], proposed ASPE (Asymmetric Scalar Product Preserving Encryption), for nearest neighbor search on encrypted data, but only limited to the linear search. Similarly, Hu et al. [12] proposed scheme with tree structure and ASPE, for faster and better search efficiency. Also, Yao et al. presented another solution with Voronoi Diagrams. Some new scheme for secure nearest neighbor search proposed by [13][14], they consider search between two parties and manages their database, but not efficient for large data sets. Some proposed only query privacy not covered data privacy [15].

3. PROBLEM STATEMENT
3.1 System Model
In our scheme, consider a cloud server, data owner and a user as shown in Figure 1. The data owner outsourced its data to the cloud server. The user wants to execute the query from the outsourced data, and we mainly focus on the nearest neighbor search inherently multidimensional view. As we know the cloud server is honest-but-curious to the data owner and user, which may result in the leakage of the user query. For this, it is required to encrypt the outsourced data as well as the query submitted to the cloud server.

Our system model consists of three parties in the cloud environment.

1. Data Owner: A data owner is one who manages a database D with d dimensional points outsourced the database to a cloud server and in encrypted form e(D) and no local backup is there.

2. Cloud Service Provider: It provides the cloud storage and computing services to store the encrypted database e(D) and on that run queries for the user.
3. **User:** A user who requests the query $Q$ from the database and result in the formation of the data point in the database.

![Figure 1. System Architecture](image)

Based on the above model we define our nearest neighbor search. To achieve the fast and efficient query result, we built a secure index using the skip graph. We indexed all the data records and each data record is a point of data space. For the privacy of the query we encrypt the contents of query by a secure cryptography scheme $E$. To search the encrypted data from the cloud server we submit both the encrypted index data and the query $e(D)$ and cloud server returns the nearest point to the $q$ as $(q_1, \ldots, q_d)$.

### 3.2 Security Model

The cloud service providers are honest as well as curious to know the additional information stored and processed by a user. This assumed that the cloud service provider provides data storage and return correct answer to the user. A user has authenticated the user as there is mutual trust between the data owner and user. The secret key is reasonable means of authentication and establishes a secure channel between the data owner and user.

### 4. METHODOLOGY

In this section, provides our methodology. For efficient and secure nearest neighbor search, make a choice for the better data structure for secure indexes, than search algorithm designed and then compare the encrypted query and secure indexes with high efficiency. To manage the data in the cloud, the data owner built the secure index and outsourced the encrypted data to the cloud server. The cloud service provider performs the required nearest neighbor search on the secure index $e(q)$ of the user. As the query reaches the destination data query point, the progressively lower level is used. It is highly efficient for the fast nearest neighbor query.

#### 4.1 Secure Skip Graph Indexing

In this, apply hybrid data structure, KD tree on the $d$ dimensional data and organize nodes according to the membership key in skip graph (SG). For secure indexing, encryption the data content, to make it safe form the cloud environment. To efficiently search the encrypted data on the cloud, we adopt the mOPE( mutable order-preserving encoding), which enables us to compare the search query point with the encrypted data. So first we construct the index then encrypt it before outsourced it to the cloud.

### 4.2 SG Index Construction

In this, we use the hybrid approach as shown in Figure 2 (a) and (b); firstly partitioned the set of data points by KD tree space decomposition method. Then further split the set of points into equal cardinality using horizontal and vertical splits. Each splitting nodes organized in their region without overlapping. After that, we arrange the data points into a skip graph using the region codes as the membership keys. For retrieving the data point, use the standard routing in skip graph structure i.e. starting from the highest level node and routes the query to the neighboring data nodes. As the query reaches the destination data query point, the progressively lower level is used. It is highly efficient for the fast nearest neighbor query.

### 4.3 SG Index Encryption

The SG index structure is then encrypted using the mOPE (mutable Order-Preserving Encryption scheme) as shown in Figure 2. (c), and the actual data is encrypted with a secure symmetric encryption method, AES. In this, we use mOPE to encrypt the SG index into secure indexes. So each node in the graph contains ciphertext $C$ for indexed data point based on the order of its plain text. The cloud server does not know the actual plaintext data. The SG index is encrypted into ciphertext $C$ and preserves the order of plaintext by using mOPE encoding.
The user query. In this, the service provider allowed to search the secure index for the required encrypted query search. But as the query and database both are encrypted, the cloud server can learn e(q) and the encrypted index c. The cloud server can generate the query result in an encrypted form. Hence, our scheme doesn’t provide and information access to the cloud service provider or attacker.

### 5. PROPOSED SECURE NEAREST NEIGHBOR SCHEME

The proposed scheme includes the encryption of the data, secures indexing of data and encrypts the user query. In this, present the detailed working of our proposed scheme. It consists of the following algorithms.

1. **Generate key** (GK): GK(1) → SK_opE

   In this step, the data owner generates a secret key(SK_opE), which is used to encrypt the both data D and Query q. And SK_opE is a secret key generates while index construction.

2. **Built Secure Index:**

   SGIndex(SK_opE, D, m) → c, Order(c) → E

   In the first step from the given data set, the data owner uses the SG index to build the securely encrypted index. It takes the input as secret key SK_opE, Data set D and the membership vector m, and output the ciphertext c. It is used to generate encrypted data using the secret key. The secure SG index and encrypted text both stored on the cloud server. In the second step, we encode the encrypted data using membership key in skip graph with mOPE scheme. It takes the ciphertext c and generates output e using mOPE scheme. As shown in fig. 3 the data point is encoded by constructing the mOPE scheme, and cloud server does not know the data as well as the query searched.

3. **Encryption:** Enc(SK_opE, D) → e(D)

   Enc(SK_opE, q) → e(q)

   The data owner encrypts all the data points in database D {d1,d2,……dn} using secret key(SK_opE), data stored on cloud server in encrypted form e(D) {{d1},{d2},……,{dn}}. AES is the encryption algorithm used in our scheme. After this, the data owner outsourced the encrypted database e(D), Secure SG Index c, and mOPE encoding E to the cloud server. The registered users can access the data from the cloud server with their generated secret key.

4. **Secure Nearest Neighbor Search:**

   NNSearch(e(q), c(q)) → e_q

   This step is performed on the cloud server in encrypted data. When the user generates query q, it runs on data stored on the cloud server, and the cloud serves to search the secure SG Index c. After receiving user query q, the cloud server encodes c(q) using the encoding scheme and denotes it as e(q). Now the cloud server traverses the indexed data source while traversing the SG index we have to compute the secure distance of data point from the query point and then select the minimum distance among all the data point according to the query point. After that, return the corresponding nearest neighbor data query point in encrypted form e(q).

5. **Decryption:** DecKey(e_q, SK_opE) → d_q

   To check the output, the user needs to decrypt the query result using secret key(SK_opE). Once the user decrypts the output received, the corresponding nearest neighbor query point retrieved.

### 6. SECURITY ANALYSES

#### 6.1 Data Privacy

In our scheme, the database is encrypted using the secret key before outsourced it to the cloud server. So without the secret decryption key, the data cannot be decoded into plaintext. It also analyze that the cloud service provider cannot analyses the information from the encrypted data. But in case, if the attacker gets the access to the encrypted text, plain text cannot be decrypted the secret key. Because indexing is done in encrypted, it becomes difficult for the attacker or the cloud service provider to deduce the plaintext from the cipher text. So we analyze that the cloud service provider can access the encrypted index but nothing more than that.

#### 6.2 Query Privacy

This uses the encoding scheme to encode the encrypted query with the secure index, the cloud service provider allowed to search the secure index for the required encrypted query search. But as the query and database both are encrypted, the cloud server can learn e(q) and the encrypted index c. The cloud server can generate the query result in an encrypted form. Hence, our scheme doesn’t provide and information access to the cloud service provider or attacker.
7. EXPERIMENTAL EVALUATION

In this proposed scheme, developed a Java based code and tested under simulated environment. AES ECB scheme is used for encryption. For the experiment, use two data sets, a synthetic data set uniformly distributed and a real graph data set from road networks of the central USA. This performed test over above datasets with different scale and selected 30 random Nearest neighbor queries, and evaluated. We use the three metrics for evaluating our scheme over the existing nearest neighbor scheme i.e. 1) data processing time 2) query response time and communication cost at the user end.

7.1 Data Processing Time

It consists of two phase at the data owner end in the first phase, create the index and in the second phase encrypt the data for secure indexing. We observe that the data processing time increases with the increase in data size as shown in figure 5(a). It also observe that the both the scheme linearly increases with the increase in data. The time consumed in the second phase is less as compared to the first phase because of the encryption time as shown in figure 5(b). However, this is the only one-time index building cost for efficient, optimized query processing time.

![Figure 5 a) Time for index construction by Data Owner. b) Data Processing time is taken by Data Owner](image)

7.2 Query Response Time

Our performance metrics indicate the time required to execute the query on the above datasets. It includes the computation time at the user end, proxy server, and the cloud server. Since all the values here are treated in the same way, they are encrypted. Our query response time is slightly better than the existing scheme as shown in figure 6. It concludes the query processing time according to the CPU response time into end user, server communication time on real datasets. We achieve the lower end user CPU time, fewer servers CPU and better performance on interactive communication time between the user and cloud server as consuming in existing schemes.
7.3 Communication Cost

The amount of data communicated between the cloud server and the end user. Our scheme achieves the negligible amount of communication cost as compared to the other schemes as shown in figure 7. It is because of the indexing structures used and the better data partition scheme. Also, our scheme provides better interaction between user and cloud server, and better result due to mutable order-preserving encryption.

8. CONCLUSION

This paper proposed a new and efficient scheme for the secure nearest neighbor search in the encrypted data. The outsourced the data and secure index to the cloud service provider in the encrypted form. Due to the encrypted data, the computation of the query becomes complex difficult. Then use the better encoding scheme OPE, use mutable order preserving scheme. It protects both the secure index and the query. It provides the protection of privacy of data content on the cloud server. From the extensive experiment evaluation; we found our scheme is more efficient and scalable as compared to the existing schemes. In future, extend this work to support other environments and similarity queries.

9. REFERENCES


