Hindawi
Mathematical Problems in Engineering
Volume 2022, Article ID 5279284, 10 pages
https://doi.org/10.1155/2022/5279284

Research Article

Encrypted Cloud Data Mark and Group Search Method Based on Singular Value Decomposition

Lingbing Tao,[1] **Jian Huang**,[1] **YingKun Song**,[1] **and Zhixin Tie**[1,2]

[1]*School of Information Science and Technology, Zhejiang Sci-Tech University, Hangzhou 310018, China*
[2]*Keyi College, Zhejiang Sci-Tech University, Shaoxing 312369, China*

Correspondence should be addressed to Zhixin Tie; tiezx@zstu.edu.cn

Received 27 February 2022; Accepted 29 March 2022; Published 22 April 2022

Academic Editor: Zaoli Yang

In order to improve the retrieval efficiency and security of the cloud server, an encrypted cloud data mark and group search method (MGSM) based on singular value decomposition is proposed in this paper. Firstly, all documents are clustered, then indexes and query marks are constructed according to classes, and then documents with low correlation are filtered according to their matching degree. Secondly, the reserved document index vector is expressed as an index matrix, and the singular value decomposition (SVD) algorithm is employed to reduce the dimensionality of the matrix. Then, the corresponding threshold is set to improve the search efficiency while ensuring accuracy. Thirdly, the reduced-dimensional indexes are grouped to reduce the high-dimensional encryption key to multiple low-dimensional keys, further reducing the index encryption time. Theoretical analysis and experimental results show that the proposed method is more feasible and more effective than the compared schemes.

1. Introduction

With the development of big data and cloud computing technology, cloud storage has been receiving more and more attention. Many users and enterprises have begun to outsource complex data from local sites to commercial public clouds in order to obtain great flexibility and economic savings, as well as to realize information sharing and processing [1]. However, in the cloud storage environment, there are also some hidden data security risks, and some privacy such as personal information or confidential files may also be easily leaked by the server. To ensure that private information is not leaked, data needs to be encrypted before being stored in the cloud. Although the encrypted data may be protected from attacks by illegal users, unauthorized users, and untrusted cloud service providers, it also brings practical problems, such as low retrieval efficiency and high retrieval difficulty [2].

So far, many scholars have conducted research on searchable encryption technology [3–6]. Through these technologies, users may enter keywords to search encrypted documents. However, it is not scientific to directly apply these technologies to complex document systems, because these methods are not only inefficient in retrieval but also not suitable for more demanding retrieval requirements. Some related studies [7–9] also try to improve the flexibility of ciphertext retrieval, but they still may not sort out the corresponding data according to the needs of users. The existing ciphertext sorting search solution in the cloud storage environment takes a long time to create and update the encrypted index, and as the number of documents increases, the retrieval efficiency will gradually decrease. For this reason, finding a solution that may reduce retrieval costs and improve retrieval efficiency is the current main research direction.

Therefore, this paper firstly filters documents with low matching degree with query words by tag matching to reduce the number of documents that need to calculate scores, then reduces the dimension of the remaining index vectors to reduce the calculation amount of index encryption, and at the same time sets a threshold to ensure certain accuracy.

The rest of this paper is organized as follows. Section 2 will describe the current research progress related to searchable encryption. Section 3 will introduce the system model, attack model, and design goals, as well as some symbols used in this paper. Section 4 presents the proposed cloud data grouping

encryption sorting search method based on singular value decomposition [10] in detail. Section 5 will conduct theoretical analysis and experimental analysis of the proposed scheme. Finally, conclusions are drawn in Section 6.

2. Related Work

Searchable encryption technology is to encrypt data and their indexes and store these encrypted data and encrypted indexes in a remote server and then search the encrypted data through a specific trapdoor which is generated according to the provided searching keywords. Except to perform storage and search, the cloud server may not obtain any relevant data information during the whole process.

The earliest searchable encryption technology was proposed by Song et al. [11]. This scheme divides the document into multiple keywords and expands them to the same length and then encrypts them with a stream password. When searching, it judges whether keywords exist by comparing encrypted files with search words, but this full-text search method is too inefficient. Goh et al. [12] used the secure index structure of the Bloom filter to store the hash value of the keywords contained in the index of the file and passed the Bloom filter to map the search result again during query. Chai et al. [13] first proposed a "semihonest and curious" cloud server model. In order to save computation and bandwidth resources, the server provider may only perform part of the search operation and return part of the search results. For the purpose of resisting this kind of server, they proposed verifiable searchable encryption scheme based on word search tree index structure. In the PKC scheme proposed by Boneh et al. [3], only those with a public key may write data, and those with a private key may search, but the encryption calculation is more complicated and does not support multiple keywords. Mahajan et al. [14] proposed a hierarchical clustering method for cloud data protection. An important part of the framework is data replication and the use of SHA1 hash strategy checking.

In terms of search efficiency, Cao et al. [15, 16] solved the sorting search problem of encrypted data, enhanced the usability of the system, and proposed a search scheme based on multikeyword sorting (MRSE), which calculated the inner product score by indexing vector and request vector to sort documents. However, for a large number of documents, the search is too computationally expensive, time-consuming, and not accurate. Saini et al. [17] proposed a keyword fuzzy search scheme. By constructing a keyword fuzzy set, users may tolerate spelling errors and format inconsistencies when searching, but they may not search for documents related to keyword semantics. Ahmed et al. [18] improved search efficiency by using encrypted dynamic index, which may dynamically update the index when the encrypted data set changes. Some scholars have proposed a tree-based search scheme. Krishna et al. [19] proposed a tree-based ranking search scheme, which uses a binary tree to establish a dynamic index, which reduces index generation and query time. Pang et al. [20] proposed a verifiable search encryption scheme, which verifies the query results through the Merkle Hash tree. Peng et al. [21] used bilinear mapping to construct a tree-based index encrypted with addition order and privacy protection function family. The cloud server merges these indexes and uses a depth-first algorithm to search for documents. Chen et al. [22] reported an efficient and dynamic multikeyword sorting search scheme. They first used coordinated matching to obtain the relevance of query keywords in outsourced documents and then used inner product similarity for analysis and finally used block sparse diagonal matrix and permutation matrix to improve search speed.

Based on the MRSE scheme, we propose an encrypted cloud data mark and group search method based on singular value decomposition (MGSM). First, the vector space model is used to build an index vector and mark for each document according to the position of its keywords in the keywords dictionary, and then an index matrix is generated using these index vectors. After that, the singular value decomposition algorithm is employed to reduce the dimensionality of the index matrix. On the basis of this, the reduced dimensionality indexes are further grouped, which improves the speed of index encryption. At last, the encrypted index will be sent to the cloud server. When querying, the cloud server calculates the inner product of the group index vector and the group query vector of each document and returns the first k documents required by the user in descending order. The contributions of this paper are summarized as follows:

(1) The documents are clustered, and the words with high correlation are extracted by class to construct a dictionary, so as to generate index marks in the feature set and filter documents with low correlation by matching with query marks.

(2) Using singular value decomposition (SVD) algorithm to reduce the dimensionality of the index matrix and query vector of the document, and by setting corresponding threshold, the accuracy and safety of the results are ensured.

(3) The reduced dimensionality index vectors are grouped to reduce the dimensionality of each encryption key, thereby improving the search efficiency.

3. Problem Formulation

3.1. System Model. Before introducing the research objectives and main content, the paper first introduces the searchable encryption system model and threat model. The system model of ciphertext retrieval is shown in Figure 1.

The system model includes the data owner, the user, the private server, and the public cloud server. These four entities and the ciphertext search method form a system model in which the data owner and the user are honest and trustworthy, and the cloud server is semitrustworthy.

Data Owner. The data owner is the entity that owns documents. It is mainly responsible for extracting the keywords of each document, establishing the document index, then encrypting the document and the document index, and finally uploading the encrypted ciphertext document and its encrypted index to the cloud server. When you need to modify the data, repeat the above process.

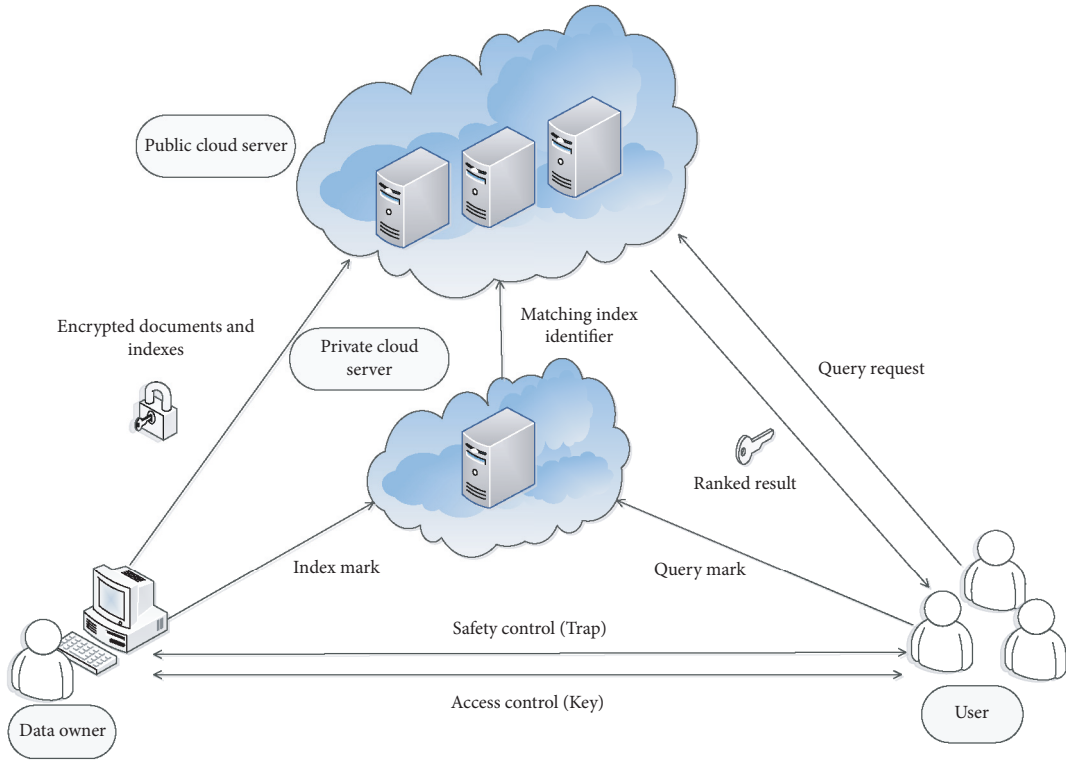

Figure 1: System model of MGSM.

Private Cloud Server. The private cloud server is used to store the index marks uploaded by the data owners and then match them with the query marks sent by users and send the index marks with high matching degree to the public cloud server.

Public Cloud Server. The public cloud server is used for storing the document index uploaded by the data owner and the encrypted document set, calculating the inner product score of the encrypted index vector corresponding to the trap door sent by the user and the index mark sent by the private cloud server, and then returning the required first k documents to the user.

User. The user is a data user. When inputting a query keyword, a trap door will be generated with the key returned by the data owner and then sent to the cloud server together with the query mark. The cloud server will return the corresponding encrypted document according to this, and the user will decrypt the document according to the key.

3.2. Attack Model. In the communication process among the data owner, the user, and the cloud server, an attacker may intercept the communication and derive additional information from the intercepted information. The cloud server is considered "honest and curious." Specifically, the cloud server will honestly perform specified operations, but at the same time it will also try to obtain and analyze private information from files, indexes, or trapdoors. In this paper, the cloud server only knows the encrypted data documents, indexes, and query trapdoors. However, the cloud server is "curious"; it will learn more information during the search process, such as query keywords and encrypted document information, and derive the encryption key based on the correlation between the trapdoors and the query keywords. According to the amount of information obtained by the cloud server, the cloud server attacks are divided into two categories.

Known Ciphertext. The cloud server only knows encrypted information, such as encrypted data sets, the encrypted indexes, and the trapdoors.

Known Background. The cloud server may know more information, such as the association relationship of search requests (trapdoors), or infer query keywords through trapdoors and query results.

3.3. Symbol Description. The symbols and descriptions used in this paper are shown as follows:

(1) A: the original document set, denoted as $A = \{A_1, A_2 \ldots A_k\}$.

(2) A_i: type i document set, denoted as a set of m documents $A_i = \{F_1, F_2 \cdots F_m\}$.

(3) W: the keyword dictionary, denoted as $W = \{W_1, W_2, \cdots W_k\}$.

(4) W_i: type i keyword dictionary, denoted as a set of nonduplicated keywords $W_i = \{w_1, w_2 \cdots w_n\}$.

(5) D: set of index vectors in class i, denoted as $D = \{D_1, D_2, \cdots D_m\}$.

(6) D_i: i-th document index vector, denoted as $D_i = (d_{i1}, d_{i2} \cdots d_{in})^T$.

(7) $\widetilde{D}$: the index vector retained after mark matching, denoted as $\widetilde{D} = \{\widetilde{D}_1, \widetilde{D}_2 \cdots \widetilde{D}_c\}$.

(8) D_i': the index vector of the i-th document after dimensionality reduction, denoted as $D_i' = (d_{i1}', d_{i2}' \cdots d_{ik}')^T$.

(9) D': set of index vectors in class i after dimensionality reduction, denoted as $D' = \{D_1', D_2' \cdots D_c'\}$.

(10) P_{ij}: the j-th group vector of the i-th document after dimension expansion.

(11) P_i: the index vector of the i-th document after dimension expansion, denoted as $P_i = (D_i'^{\mathrm{T}}, \varepsilon_1, \varepsilon_2 \cdots \varepsilon_u, 1)^T$.

(12) I: the encrypted index set of type i, denoted as $I = \{I_1, I_2, \cdots I_m\}$.

(13) q: the query vector, denoted as $q = (q_1, q_2 \cdots q_n)^T$.

(14) q': the query vector after dimensionality reduction, denoted as $q' = (q_1', q_2' \cdots q_k')^T$.

(15) Q: the query vector after dimension expansion.

(16) Q': the query vector after segmentation, denoted as $Q' = \{Q_1', Q_2', \cdots Q_g'\}$.

(17) Q_i': the i-th group query vector.

(18) T: the trapdoor.

3.4. Word Interpretation

Keyword Dictionary. Keywords extracted from all classified documents will form a keywords dictionary after deduplication.

Vector Space Model. Each document is represented by a vector, the vector's size is equal to the size of the keywords dictionary, every dimension of the vector stands for a keyword, and its value represents the score DS of the keyword at that location. In the field of encrypted search, the products of the word frequency tf and the inverse word frequency idf are usually used to calculate the score DS as shown in the following equation:

$$DS = tf \cdot idf \ . \tag{1}$$

The word frequency tf refers to the frequency of a keyword appearing in the document. The higher the word frequency is, the more important the keyword is to the document. The inverse word frequency idf refers to the number of documents containing a certain keyword, reflecting the importance of the keyword in the entire document set. The greater the number of documents, the lower the degree of discrimination of keywords from documents.

Document Score. The document score reflects the degree of matching between the query keywords and the keywords in the document. The cloud server will calculate the document scores, sort them by value, and return the search results. When the cloud server receives the query request q', it may use (2) and (3) to calculate the score of document F_i [23].

$$\mathrm{Score}(F_i) = \frac{1}{|F_i|} \sum_{w_j \in \widetilde{w}} \left(1 + \ln\ f_{ij}\right) \cdot \ \ln\left(1 + \frac{m}{f_j}\right). \tag{2}$$

$$|F_i| = \sqrt{\sum_{j=1}^{n} \left(1 + \ln\ f_{ij}\right)^2}. \tag{3}$$

In the above equation, $|F_i|$ is the Euclidean length of the i-th document, w_j is a keyword in document F_i, f_j is the number of times keyword w_j appears in document F_i, $\widetilde{w}$ is a collection of keywords contained in document F_i, m is the total number of documents, and f_j is the number of documents containing keyword w_j.

3.5. Introduction of Singular Value Decomposition. Assuming that there is a matrix A of $m \times n$, it can be transformed into the multiplication of three matrices as shown in the following equation:

$$A = U\Sigma V^T. \tag{4}$$

In the above equation, U is a matrix of $m \times n$ whose column vectors are mutually orthogonal unit vectors and $U^TU = E$, V^T is a matrix of $n \times n$ whose column vectors are mutually orthogonal unit vectors, and $V^TV = E$. DD^T and D^TD have the same eigenvalue $\lambda_i (i = 1, 2 \cdots r)$, where r is the rank of matrix D. Σ is a matrix of $m \times n$, which has a value of $\alpha_i = \sqrt{\lambda_i}$ only at the main diagonal position and 0 at other positions, where α_i is called singular value and E is called singular value matrix.

Singular values are arranged from big to small in singular matrix Σ, and the decline range is very large. Therefore, the original matrix can be roughly represented by the largest first k singular values and their left and right singular vectors, so as to achieve the purpose of dimension reduction. The specific process is shown in the following equation:

$$A_{mn} = U_{mm}\Sigma_{mn}{V_{nn}}^T \approx U_{mk}\Sigma_{kk}{V_{nk}}^T. \tag{5}$$

Thus, we can use right singular matrix V_{nk} to reduce the column from n to k as shown in (6). According to the principle of principal component analysis algorithm, the right singular matrix V_{nk} is the projection matrix.

$$A_{mk} \approx A_{mn}V_{nk}. \tag{6}$$

As an index of dimension reduction, if k is too large, it will lead to information redundancy and high computational complexity, while if k is too small, it will lead to information loss and lower accuracy. Thus, a threshold θ is introduced as the standard and uses the ratio of the sum of squares of the first k singular values to the sum of squares of all singular values to measure the magnitude of dimension reduction.

4. The Proposed MGSM

The following will be divided into eight parts to introduce the proposed cloud data grouping encryption sorting search method based on singular value decomposition.

4.1. Generating Dictionary. The data owner first extracts the features of documents and constructs corresponding feature vectors according to the weights of keywords. Then, using k-means algorithm [24], m document F is classified into k classes. For all documents in each class, their keywords are extracted and these keywords are deduplicated. The keywords of the same class can be arranged together because of their high correlation, and then a keyword dictionary with size $n = n_1 + n_2 \cdots n_k$ is constructed, where n is the total number of keywords in the dictionary and n_i is the number of keywords in class i.

4.2. Mark Matching. For the i-th document in any class, it can be represented as $D_i = (d_{i1}, d_{i2} \cdots d_{in})^T$. If the word frequency score of the corresponding position of the keyword in the document is not 0, the value of marking the position is 1, and finally a mark $B_{\varphi i} \in \{0,1\}^{(n)}$ is obtained, where φ represents the category to which the mark belongs. Similarly, for the query keywords input by the user, the corresponding query vector $q = (q_1, q_2 \cdots q_n)^T$ can be generated according to the dictionary. If the query keyword does not match the keyword at the corresponding position in the dictionary, $q_i = 0$ is set; otherwise the mark is set as 1, and finally get the query mark $b \in \{0,1\}^{(n)}$ is got. Then, the private cloud server filters documents with low matching degree by matching index mark $B_{\varphi i}$ and query mark b bit by bit and returns mark $B_{\varphi i}$ with high matching degree to the public cloud server.

As shown in Figure 2, it is assumed that the number of document classifications is 3, the number of documents of each type is 2, 3, and 2, respectively, and the dimension of keyword dictionary and mark vector is 15. The second category is the document set about cloud computing. The keywords extracted from it may include cloud, computing, encrypted, and search. These words will be arranged at specific positions in the dictionary in sequence, such as the last part of the dictionary, so that the generated index mark 1 will also concentrate on the last part. When the user input includes a plurality of related query keywords such as cloud and search, the matching degree between the query mark b and the index marks B_{21}, B_{22}, and B_{23} of the second type document set is higher, and only the documents with the highest correlation before need to be returned in the end, so the documents in the first and third types with lower matching degree can be filtered without having great influence on the results, thus avoiding unnecessary score calculation for all documents and improving the search efficiency.

4.3. Dimensionality Reduction

Step 1. Generating the initial index matrix. Similar to the process of creating marks, the i-th document in any class will be represented as the vector, where $D_i = (d_{i1}, d_{i2} \cdots d_{in})^T$ and d_{ij} is the score of the j-th keyword in the i-th document. Then c index vectors corresponding to the filtered index marks are expressed as matrix $\tilde{D}$, where $\tilde{D} = \{\tilde{D}_1, \tilde{D}_2 \cdots \tilde{D}_c\}$.

B_{11}	:	1	0	1	1	1	0	0	0	0	1	0	0	0	0	0
B_{12}	:	1	1	1	0	1	0	0	0	0	0	0	0	0	1	0
B_{21}	:	0	0	1	0	0	0	0	0	0	0	1	1	1	1	1
B_{22}	:	0	0	0	0	0	0	0	0	0	0	1	0	1	1	1
B_{23}	:	0	0	0	1	0	0	1	0	0	0	1	1	1	1	1
B_{31}	:	0	0	0	0	0	1	1	1	0	1	0	0	0	0	1
B_{32}	:	0	1	0	0	0	1	1	1	1	1	0	0	0	0	0
b	:	0	0	0	0	0	0	0	1	0	0	1	1	0	1	1

Figure 2: Matching process of query mark and index mark.

Step 2. Using the SVD to reduce the dimension. Then the reduced-dimensional matrix D' is obtained, where $D' = \{D'_1, D'_2 \cdots D'_c\}$ and $D'_i = (d'_{i1}, d'_{i2} \cdots d'_{ik})^T$.

4.4. Index Grouping. For each index vector D'_i, its dimension is extended from k to $k + u + 1$, where u is the number of virtual keywords, the values of $(k+1)$-th dimension to $(k+u)$-th dimension are set to arbitrary random numbers $\varepsilon_i (i = 1, 2, \cdots u)$, and they obey the same uniform distribution, and the value of $(k+u+1)$-th dimension is set to a constant 1, and the expanded vector is expressed as $P_i = (D'^T_i, \varepsilon_1, \varepsilon_2 \cdots \varepsilon_u, 1)^T$. Finally, the $k+u+1$ elements of vector P_i are divided into g group and expressed as $P_i = (P_{i1}, P_{i2} \cdots P_{ig})^T$; if g can be divided by $k+u+1$, then the dimension of vector $P_{ij} (j = 1, 2, \cdots g)$ is $e_1 = (k+u+1)/g$; otherwise, the dimension of the first $g-1$ group vector $P_{ij} (j = 1, 2, \cdots g-1)$ is e_1, and the dimension of the g-th group vector P_{ig} is $e_2 = (k+u+1)\%g$.

4.5. Generating Key. The data owner randomly generates $2g$ random invertible group key matrices $M_1 = \{M_{11}, M_{12} \cdots M_{1g}\}$ and $M_2 = \{M_{21}, M_{22} \cdots M_{2g}\}$ and g group division indicator vector $\{S_1, S_2 \cdots S_g\}$ according to the grouping number of the index. If $g | (k+u+1)$, then the dimension of each group is e_1; otherwise, the dimension of the first $g-1$ group vector is e_1, and the dimension of the g-th group vector is e_2. The group keys M_{1c} and $M_{2c} (c = 1, 2, \cdots, g-1)$ are random reversible group matrices, the dimensions of which are $e_1 \times e_1$; M_{1g} and M_{2g} are random invertible group matrices, the dimensions of which are $e_2 \times e_2$; and $S_i \in \{0,1\}^{e_1} (i = 1, 2, \cdots, g-1)$, and $S_g \in \{0,1\}^{e_2}$.

4.6. Creating Index

Step 1: Random split. According to each group indicator vector S_j, the corresponding group vector $P_{ij} (i = 1, 2, \cdots, g)$ of the index vector P_i is randomly divided into P'_{ij} and P''_{ij}. For any position r, the rules for segmentation are as shown in the following equation:

$$\begin{cases} P'_{ij}[r] = P''_{ij}[r] = P_{ij}[r] \quad \text{for} & S_j[r] = 0 \\ P'_{ij}[r] + P''_{ij}[r] = P_{ij}[r], P'_{ij}[r] \neq P''_{ij}[r] \quad \text{for} & S_j[r] = 1 \end{cases}. \tag{7}$$

Step 2: Index encryption. The group keys M_{1c} and $M_{2c}\,(c = 1, 2 \cdots, g)$ are used to encrypt the divided group index P'_{ij} and $P''_{ij}\,(i = 1, 2 \cdots m.\, j = 1, 2 \cdots g)$, respectively, and the encrypt process of the i-th document is, respectively, expressed as $P'_i M_1 = \{P'_{i1} M_{11}, P'_{i2} M_{12} \cdots P'_{ig} M_{1g}\}$ and $P''_i M_2 = \{P''_{i1} M_{21}, P''_{i2} M_{22} \cdots P''_{ig} M_{2g}\}$; then the entire encryption process of a document is $\{P'_i M_1, P''_i M_2\}$.

Step 3: The data owner uploads the encrypted documents and their encrypted indexes to the cloud server.

4.7. Creating Trapdoor

Step 1: Creating query vector. First, the user enters query keywords. Then, they are compared with each keyword of the keyword dictionary W; if one of them is the same as the i-th keyword of dictionary W, then $q_i = 1$ is set; otherwise, $q_i = 0$ is set. Finally, query vector $q = (q_1, q_2 \cdots q_n)^T$ is generated.

Step 2: Dimensionality reduction and expansion. The projection matrix V_{nk} is used to reduce the dimension of the query vector q to obtain the reduced-dimensional vector $q' = (q'_1, q'_2 \cdots q'_k)^T$; then the dimension of q' will be expanded from k to $k + u + 1$. Among them, the v dimension is arbitrarily selected from $k + 1$ to $k + u$ and set to 1; then the rest are set to 0. At last, the values of the $(k + u)$-th dimensions will be multiplied by a random number r and the value of the $(k + u + 1)$-th dimension is set to a random number t. The expanded final query vector is expressed as Q.

Step 3: Query vector grouping. The query vector Q is divided into g group vectors $Q = \{Q_1, Q_2, \cdots Q_g\}$. If $g|(k + u + 1)$, the dimension of each group is e_1; otherwise, the dimension of the first $g - 1$ group vector is e_1, and the dimension of the g-th group vector is e_2.

Step 4: Random segmentation. According to each group indicator vector S_j of the indicator vector S, the corresponding group vector $Q_j\,(j = 1, 2 \cdots, g)$ of the query vector Q is randomly divided and expressed as $Q' = \{Q'_1, Q'_2 \cdots Q'_g\}$ and $Q'' = \{Q''_1, Q''_2 \cdots Q''_g\}$. The rules of division are as shown in the following equation:

$$\begin{cases} Q'_j[r] = Q''_j[r] = Q_j[r] \quad \text{for} & S_j[r] = 1 \\ Q'_j[r] + Q''_j[r] = Q_j[r], Q'_j[r] \neq Q''_j[r] \quad \text{for} & S_j[r] = 0 \end{cases}. \tag{8}$$

Step 5: Generate a trapdoor. The group keys M_{1c}^{-1} and $M_{2c}^{-1}\,(c = 1, 2 \cdots, g)$ are used to encrypt the divided query group index Q'_j and $Q''_j\,(j = 1, 2 \cdots, g)$, respectively. The encrypted index is $M_1^{-1}Q' = \{M_{11}^{-1}Q'_1, M_{12}^{-1}Q'_2 \cdots M_{1g}^{-1}Q'_g\}$ and $M_2^{-1}Q'' = \{M_{21}^{-1}Q''_1, M_{22}^{-1}Q''_2 \cdots M_{2g}^{-1}Q''_g\}$, and finally a trapdoor is generated which is expressed as $T = \{M_1^{-1}Q', M_2^{-1}Q''\}$.

4.8. Query. The user uploads trapdoor T to the cloud server. After receiving it, the cloud server calculates the inner product scores of the index and the trapdoor and then sorts them in descending order according to the inner product. Finally, the k encrypted documents with higher scores are returned to the data consumer. The calculation process of the inner product is as shown in the following equation:

$$\begin{aligned} I_i \cdot T &= \{P'_i M_1, P''_i M_2\}\{M_1^{-1}Q', M_2^{-1}Q''\} = P'_i Q' + P''_i Q'' = P_i Q \\ &= r\left(D'_i q' + \sum_{i \in v} \varepsilon_i\right) + t = r\left(\text{Score}(F_i, q) + \sum_{i \in v} \varepsilon_i\right) + t. \end{aligned} \tag{9}$$

5. Performance Analysis

5.1. Complexity Analysis. As the number of documents increases, the number of keywords also increases, and then the keyword dictionary becomes larger. This leads to larger original index matrix data and more redundant information. In the MRSE scheme, the dimension of the encryption matrix directly depends on the size of the keyword dictionary; thus the encryption time complexity is $O(m(\mathrm{n} + u + 1)^2)$. In order to reduce the dimensionality of the encryption matrix, we may first construct marks for all index vectors and query vectors according to the different categories of their elements. By matching mark vectors, we filter a large number of documents with low correlation, thus reducing the time complexity of encryption to $O(c(\mathrm{n} + u + 1)^2)$. Then, dimension reduction is realized by SVD to obtain n singular values in descending order. In most cases, most of the information is concentrated in the first k singular values, so the new matrix after dimensionality reduction may be approximate to the original matrix, and the encryption time complexity will be reduced to $O(\mathrm{c}(\mathrm{k} + u + 1)^2)$.

The value of k is different; thus the new matrix obtained will be different. The larger the value of k is, the smaller the magnitude of dimensionality reduction is and the more original information is retained. On the contrary, the smaller the value of k is, the greater the degree of dimensionality reduction is but the less original information is retained. The selection of k is directly related to information integrity and query efficiency; thus we can define a threshold θ to control the value of k. The specific expression is as follows:

$$\frac{\sum_{\mathrm{i}=1}^{\mathrm{k}} \alpha_{\mathrm{i}}^{\,2}}{\sum_{\mathrm{i}=1}^{\mathrm{n}} \alpha_{\mathrm{i}}^{\,2}} \leq \theta. \tag{10}$$

In the above equation, α_i represents the i-th singular value of a matrix.

In order to ensure that the information is not lost as much as possible, the value of θ may not be too low; thus the selection of k should not be too small, which leads to limited dimensionality reduction; thus the time complexity $O(c(\mathrm{k}+u+1)^2)$ is still relatively high.

In order to further reduce the time complexity, we divide all index vectors and query vectors into g groups; thus the time complexity is reduced to $O(c(\mathrm{k}+u+1)^2/g)$ at this time.

5.2. Privacy Analysis. Ensuring the security of data information is very important for the searchable encryption process. Under the known background attack model, this paper conducts security analysis from several aspects of the key security, the keyword information, the query information, and the trapdoor nonrelevance protection.

Key Security. Under the attack model with known background, the cloud server knows the index encryption process $\{P_i' M_1, P_i'' M_2\}$. For the i-th document, the encryption process of the j-th $(i = 1, 2 \cdots g)$ group vector is expressed as $\{P_{ij}' M_{1j}, P_{ij}'' M_{2j}\}$, the dimension of each group vector is set as k_j, and its value is e_1 or e_2. The cloud server does not know the specific process of dimensionality reduction, grouping, and segmentation. For the encrypted group vectors $\widetilde{P}_{ij}{}'$ and $\widetilde{P}_{ij}{}''$, the following may only be established:

$$\begin{cases} P_{ij}' M_{1j} = \widetilde{P}_{ij}{}' \\ P_{ij}'' M_{2j} = \widetilde{P}_{ij}{}'' \end{cases}. \tag{11}$$

M_{1j} and M_{2j} have $2k_j{}^2$ unknown variables, and $\widetilde{P}_{ij}{}'$ and $\widetilde{P}_{ij}{}''$ have $2k_j$ unknown variables, but the number of equations is $2k_j$, so the cloud server may not derive the secret key.

Keyword Information Protection. The introduction of random numbers is to prevent information leakage, where $\varepsilon_i (i = 1, 2, \cdots \mu)$ satisfies the normal distribution $N(\mu, \sigma^2)$, in which the standard deviation σ is used as a compromise parameter. When the standard deviation σ is smaller, the search accuracy is higher, but the relative confusion is less, and the security is reduced; thus security may be ensured by setting the value of σ. In the known background model, in order to allow the introduction of random number ε_i to effectively improve security, the system parameter w is also introduced to ensure that the index vector has at least 2^w different $\sum_{i \in V} \varepsilon_i$, so that the probability of two having the same value $\sum_{i \in V} \varepsilon_i$ is less than $1/2^w$, the number C_u^v of different $\sum_{i \in V} \varepsilon_i$ is not greater than u/v, and $u/v = 2$ reaches the maximum value. Considering $C_u^v \geq (u/v)^v$, $u = 2w$ and $v = w$ need to be set. ε_i also needs to satisfy a uniform distribution $N(\mu' - c, \mu' + c)$, its mean is μ', and the variance is $\sigma^2 = c^2/3$. To ensure that ε_i conforms to the normal distribution $N(\mu, \sigma^2)$, $\mu' = \mu/w$ and $c = \sqrt{3/w}\sigma$ should be set.

In addition, the cloud server does not know the number of keyword dictionaries; due to the fact that threshold θ is flexible and variable during dimensionality reduction, the matrix after dimensionality reduction is also changeable, thereby further improving data security.

Query Information Protection. In order to prevent the cloud server from inferring the user's query information from the trapdoor, the proposed method performs dimensionality reduction, grouping, expansion, random segmentation, and encryption processing on the query vector, so that the query keyword information will not appear in the query trapdoor, thereby protecting query information. In addition, due to the introduction of random number r and t, different or even same query requests will have different scores, thereby protecting the nonrelevance of trapdoors.

5.3. Experiment Analysis. The RFC (Request for Comments) [25] data is selected as the experimental data set. The experimental system is implemented in Java and runs on a Windows 7 server with an Intel Core i5 (2.5 GHz) processor and 8G memory.

The main factor affecting the efficiency of the experiment is the number of documents. The more documents there are, the larger the keywords dictionary will be generated. The query vector and document vector dimensions constructed by the vector space model will be higher, and the time complexity of encryption will also be higher. For this reason, we control the vector dimension and the encryption dimension by adjusting threshold θ and the number of groups g, respectively, and analyze the time impact on the MGSM scheme for comparison with the MRSE scheme.

The first experiment is about threshold θ selection of MGSM. This experiment examines how the generating trapdoor and query time change as threshold θ changes when the number of documents is 2000.

The experimental results are shown in Figure 3. As threshold θ becomes smaller, the time to execute queries and generate trapdoors gradually decreases. When it is reduced from 1 to 0.98, the time of trapdoor generation is reduced from 1.2 s to 0.38 s, which is a reduction of 68.3%. When it is reduced from 0.98 to 0.9, the time of trapdoor generation gradually decreases. For the query time, when threshold θ is reduced from 1 to 0.98, the time is reduced from 1.6 s to 0.85 s, which is a reduction of 46.9%. When it is reduced from 0.98 to 0.9, the rate of decrease in query time slows down. Since the value of the threshold will also affect the query accuracy, we will take $\theta = 0.98$ in the following experiments.

In the second experiment, the MRSE is chosen for comparison with the MGSM to test the time change of trapdoors generating and query when the number of documents changes.

As Figure 4 shows, when the number of files increases from 1000 to 6000, the trapdoor generation times of the MRSE scheme and the MGSM scheme are gradually increased. The trapdoor generation time of the MRSE scheme increases from 0.8 s to 4.6 s, while that of the MGSM scheme increases from 0.25 s to 0.8 s. This is because both

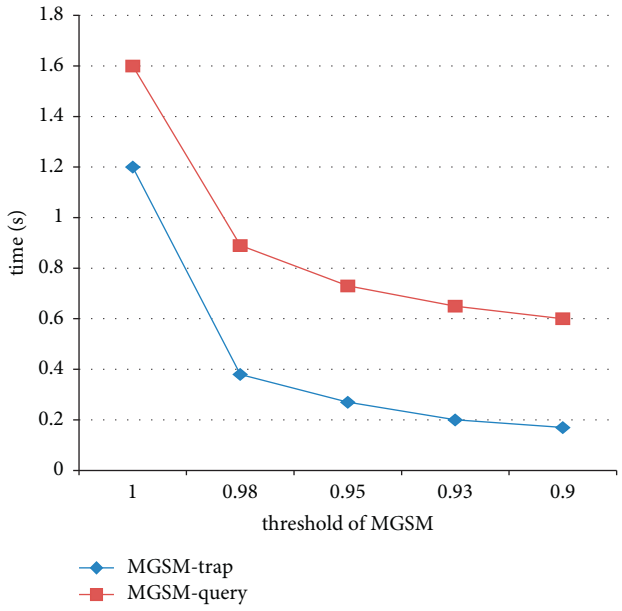

Figure 3: Time cost of generating trapdoor and query with different choice of threshold θ at 2000 documents.

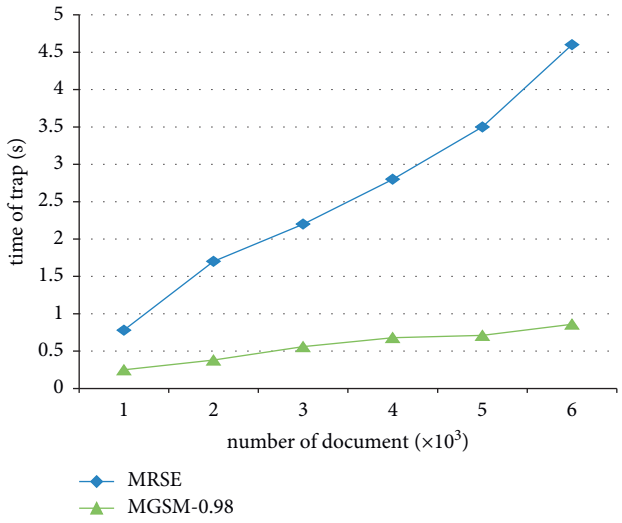

Figure 4: Comparison of two algorithms in the time of generating trapdoor with different number of documents.

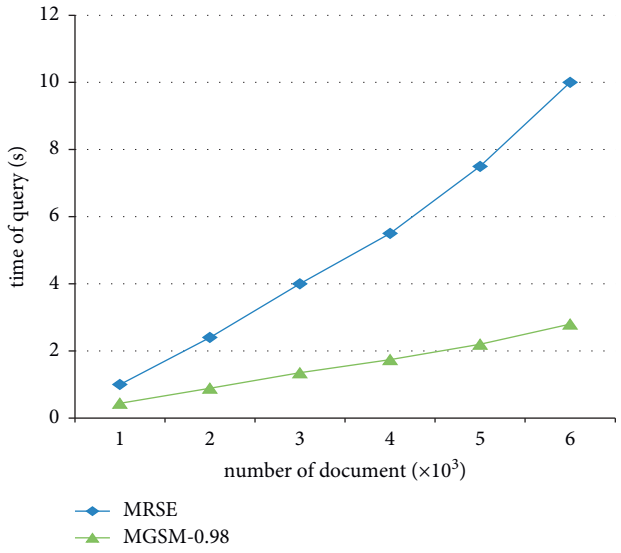

Figure 5: Comparison of two algorithms in the time of query with different number of documents.

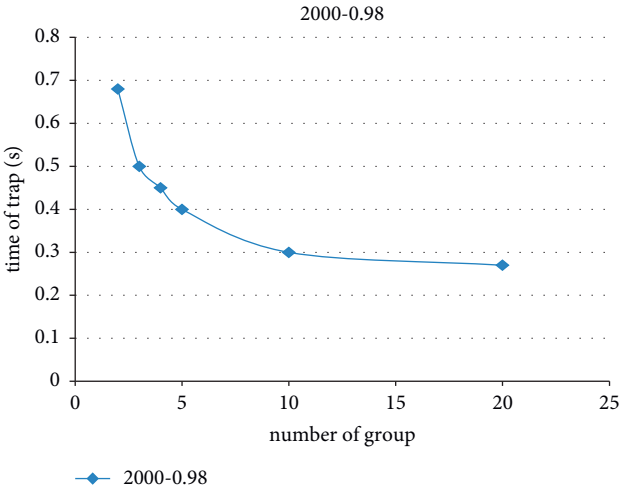

Figure 6: Time cost of generating trapdoor with different number of groups g when threshold $\theta = 0.98$ and the number of documents is 2000.

the index vector and encryption key dimensions for generating trapdoors have increased. When the numbers of documents are equal, the MGSM scheme always takes less trapdoor generation time than the MRSE scheme , and the rate of increase in the trapdoor generation time taken by the MGSM scheme is smaller than that of the MRSE scheme; thus, the MGSM scheme is more efficient than the MRSE scheme.

From Figure 5, we can see that the high-dimensional index determines the query time. The MRSE scheme takes time to increase from 1 s to 10 s, and the MGSM scheme from 0.5 s to 3 s. However, due to the fact that the MGSM scheme reduces its dimensionality, in the case of the same number of documents, the MGSM scheme takes less time than the MRSE scheme. In terms of query, the former is more efficient than the latter.

The third experiment explores the effect of the number of groups on the trapdoors generation time in the MGSM scheme. In the case of threshold, the number of documents is 2000, and the experimental results are shown in Figure 6. When the number of groups g increases from 2 to 10, the trapdoor generation time is significantly reduced, from 0.7 s to 0.3 s, a decrease of 57%. This is because the group vector and secret key dimensions after grouping are reduced, while as g further increases, the time change will gradually stabilize; when g increases from 10 to 20, the decrease is only 7%. Since the value of the appropriate grouping number g will affect the query time, we will use g = 10 in the following experiments.

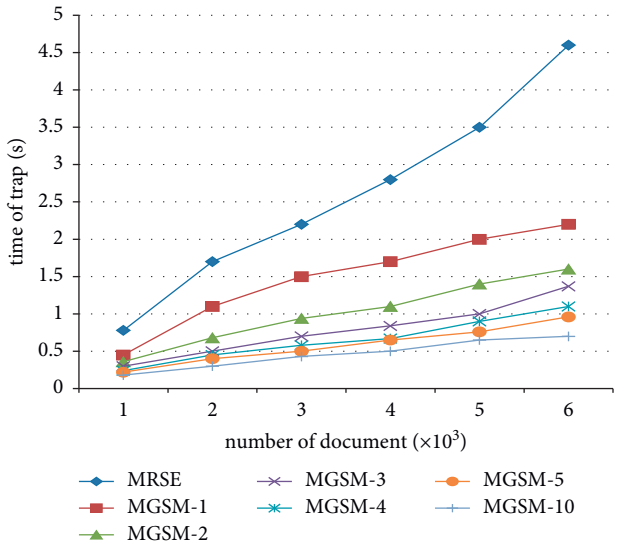

Figure 7: Time cost of generating trapdoor with different numbers of groups and documents.

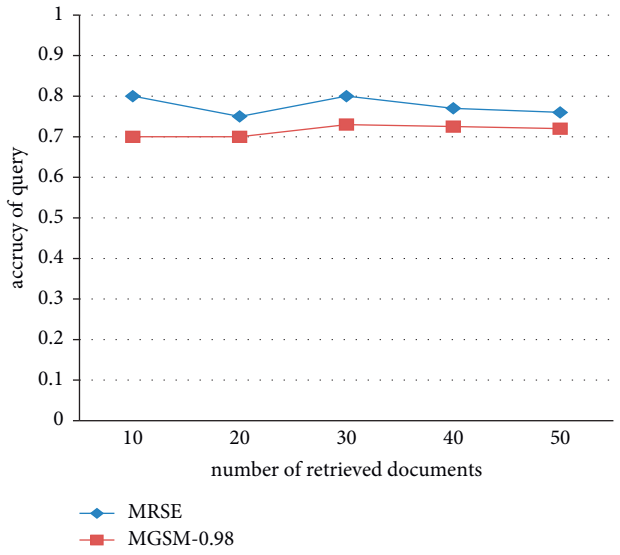

Figure 8: Query accuracy with different number of returned documents.

The fourth experiment is about the influence of the number of groups and the number of documents on the generation time of the trapdoor. In the case of threshold $\theta = 0.98$, the number of files increases from 1000 to 6000, and the experimental results are shown in Figure 7. As the number of documents increases, the trapdoor times of the MRSE and MGSM schemes are gradually increased. The time of the MRSE scheme increases from 0.8 s to 4.6 s, while the number of groups g = 10, and the query time only increases from 0.23 s to 0.7 s; the rate of increase in the trapdoor generation time taken by the MGSM scheme is much smaller than that of the MRSE scheme; thus, the MGSM scheme is more efficient than the MRSE scheme.

In the fifth experiment, we compare the proposed MGSM with the MRSE scheme on the query accuracy. Here, we define query accuracy λ as follows:

$$\lambda = \frac{\alpha}{\beta}. \tag{12}$$

In the above equation, β is the number of documents returned and α represents documents containing query keywords when the user inputs the querying words to query the documents.

When threshold $\theta = 0.98$, the number of documents is 2000, and the number of groups g = 10, the experimental results are shown in Figure 8. When the number of returned documents is increased from 10 to 50, the query accuracy of the MRSE scheme is between 0.76 and 0.8. In contrast, due to the dimensionality reduction operation, the accuracy of the MGSM scheme is between 0.7 and 0.72; although the accuracy has decreased slightly, it is still relatively stable.

6. Conclusions

This paper proposes an encrypted cloud data mark and group search method (MGSM) based on singular value decomposition. Firstly, documents are clustered, and dictionaries are constructed according to classes. Then, all documents and query keywords are marked with binary numbers through vector space model, so that the mark value 1 of the mark vector formed is relatively concentrated. Finally, documents with low correlation can be filtered by matching index marks and query marks, thus reducing the score calculation time of documents. In addition, the remaining document index vectors after filtering form a matrix, and singular value decomposition algorithm is used to reduce its dimension. After that, the index vector after dimension reduction is encrypted in groups, and the high-dimensional secret key is divided into multiple low-dimensional secret keys. Finally, the following conclusions can be drawn:

(1) The retrieval time is inversely proportional to the threshold value. When the threshold value decreases continuously, the reduction of retrieval time will gradually slow down.

(2) Trap generation time and retrieval time have a linear coefficient relationship with the number of documents, and both increase with the increase of the number of documents.

(3) The retrieval time is inversely proportional to the number of groups. The more groups, the less retrieval time, but, with the further increase of the number of groups, the decrease of retrieval time will slow down.

Although MGSM improves the efficiency of encryption and search to a certain extent, it still has some shortcomings. For example, the result of mark matching depends on the clustering effect of keywords and documents to a great extent, which may make some marks scattered, resulting in the wrong filtering of the corresponding documents. In addition, the dimensionality reduction of vectors improves the retrieval efficiency, but it will also bring about the problem of reduced query accuracy, so it is more important

to choose a suitable threshold to adjust the degree of dimensionality reduction. Theoretical analysis and experimental results show that the proposed method is more feasible and more effective than the compared schemes. In the following research, further improving query efficiency and accuracy will be the main direction.

Data Availability

The data are available at https://www.rfc-editor.org/rfc-index-100a.html.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the scientific research project of Zhejiang Provincial Department of Education (Project no. 21030074-F).

References

[1] N. Cao, S. Yu, Z. Y. Yang, W. Lou, and Y. T. Hou, "LT codes-based secure and reliable cloud storage service," in *Proceedings of the IEEE INFOCOM*, pp. 693–701, Orlando, FL, USA, March 2012.

[2] C. Wang, K. Ren, S. C. Yu, and K. M. R. Urs, "Achieving useable and privacy-assured similarity search over outsourced cloud data," in *Proceedings of the IEEE INFOCOM*, pp. 451–459, Orlando, FL, USA, March 2012.

[3] D. Boneh, G. Di Crescenzo, R. Ostrovsky, and G. Persiano, "Public key encryption with keyword search, Advances in Cryptology - EUROCRYPT 2004," in *Proc. International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 506–522, Interlaken, Switzerland, 2004.

[4] P. Golle, J. Staddon, and B. Waters, "Secure conjunctive keyword search over encrypted data," in *Proceedings of the Applied Cryptography and Network Security: Second International Conference, ACNS*, pp. 31–45, Yellow Mountain, China, June, 2004.

[5] C. Guo, R. Zhuang, C.-C. Chang, and Q. Yuan, "Dynamic multi-keyword ranked search based on Bloom filter over encrypted cloud data," *IEEE Access*, vol. 7, no. 10, pp. 35826–35837, 2019.

[6] M. Zhang, Y. Chen, and J. Huang, "SE-PPFM: a searchable encryption scheme supporting privacy-preserving fuzzy multikeyword in cloud systems," *IEEE Systems Journal*, vol. 15, no. 2, pp. 2980–2988, 2021.

[7] L. Ballard, S. Kamara, and F. Monrose, "Achieving efficient conjunctive keyword searches over encrypted data," in *Information and Communications Security. ICICS 2005*, S. Qing, W. Mao, J. López, and G. Wang, Eds., vol. 3783, pp. 27–39, Springer, Berlin, 2005.

[8] J. Li, J. Ma, Y. Miao, Y. Ruikang, X. Liu, and K.-K. R. Choo, "Practical multi-keyword ranked search with access control over encrypted cloud data," *IEEE Transactions on Cloud Computing, to be published*, p. 1, 2020.

[9] C. Zhang, S. Fu, and W. Ao, "A blockchain based searchable encryption scheme for multiple cloud storage," in *Cyberspace Safety and Security. CSS 2019*, J. Vaidya, X. Zhang, and J. Li, Eds., vol. 11982, pp. 585–600, Springer, Cham, 2019.

[10] D. Kalman, "A singularly valuable decomposition: the SVD of a matrix," *The College Mathematics Journal*, vol. 27, no. 1, pp. 2–23, 1996.

[11] D. X. D. Song, D. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," in *Proceedings of the IEEE Symposium on Security and Privacy*, pp. 44–55, Berkeley, CA, USA, May 2000.

[12] E. J. Goh, "Building secure indexes for searching efficiently on encrypted compressed data,"Cryptology ePrint Archive, Report 2003/216 2003, http://eprint.iacr.org/2003/216/.

[13] Q. Chai and G. Gong, "Verifiable symmetric searchable encryption for semi-honest-but-curious cloud servers," in *Proceedings of the IEEE International Conference on Communications*, pp. 917–922, Ottawa, ON, Canada, June 2012.

[14] N. Mahajan and V. Barkade, "Clustering based efficient privacy preserving multi keyword search over encrypted data," in *Proceedings of the 2018 Fourth International Conference on Computing Communication Control and Automation*, pp. 1–6, Pune, India, August 2018.

[15] N. Cao, C. Wang, M. Li, K. Ren, and W. Lou, "Privacy-preserving multi-keyword ranked search over encrypted cloud data," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 1, pp. 222–233, 2014.

[16] C. Wang, N. Cao, J. Li, K. Ren, and W. Lou, "Secure ranked keyword search over encrypted cloud data," in *Proceedings of the International Conference on Distributed Computing Systems*, pp. 253–262, Genova, Italy, June 2010.

[17] V. Saini, R. K. Challa, and N. S. Khan, "An efficient multi-keyword synonym-based fuzzy ranked search over outsourced encrypted cloud data, Advanced Computing and Communication Technologies," in *Advanced Computing and Communication Technologies*, R. Choudhary, J. Mandal, N. Auluck, and H. Nagarajaram, Eds., vol. 452, pp. 433–441, Springer, Singapore, 2016.

[18] A. Ahmed, S. Ramachandram, and K. U. R. Khan, "Privacy preserving dynamically indexed multi-phrase search over encrypted data," in *Proceedings of the 2018 International Conference on Advances in Computing, Communications and Informatics*, pp. 350–356, Bangalore, India, September 2018.

[19] C. R. Krishna and S. A. Mittal, "Privacy preserving synonym based fuzzy multi-keyword ranked search over encrypted cloud data," in *Proceedings of the 2016 International Conference on Computing, Communication and Automation*, pp. 1187–1194, Greater Noida, India, April 2016.

[20] H. Pang and K. Mouratidis, "Authenticating the query results of text search engines," in *Proceedings of the Vldb Endowment*, vol. 1, no. 1, pp. 126–137, Auckland, New Zealand, August 2008.

[21] T. Peng, Y. Lin, X. Yao, and W. Zhang, "An efficient ranked multi-keyword search for multiple data owners over encrypted cloud data," *IEEE Access*, vol. 6, pp. 21924–21933, 2018.

[22] L. X. Chen, L. B. Qiu, K. C. Li, W. Shi, and N. Zhang, "DMRS: an efficient dynamic multi-keyword ranked search over encrypted cloud data," *Soft Computing*, vol. 21, no. 1, pp. 1–13, 2017.

[23] J. Zobel and A. Moffat, "Exploring the similarity space," *Acm Sigir Forum*, vol. 32, no. 1, pp. 18–34, 1998.

[24] M. Steinbach, G. Karypis, and V. Kumar, "A comparison of document clustering techniques," in *Proceedings of the World Text Mining Conference Workshop*, pp. 525-526, Boston, MA, USA, August 2000.

[25] Rfc Index, https://www.rfc-editor.org/rfc-index-100a.html, 2022.