Research Article

Secure and Efficient Image Retrieval over Encrypted Cloud Data

Haihua Liang ^{1,2}, Xinpeng Zhang,^{1,3} Hang Cheng,⁴ and Qiuhan Wei¹

¹School of Communication and Information Engineering, Shanghai University, Shanghai, China

²Department of Computer Science and Engineering, Changshu Institute of Technology, Suzhou, China

³Key Laboratory of Specialty Fiber Optics and Optical Access Networks, Joint International Research Laboratory of Specialty Fiber Optics and Advanced Communication, Shanghai Institute for Advanced Communication and Data Science, Shanghai University, Shanghai, China

⁴College of Mathematics and Computer Science, Fuzhou University, Fuzhou, China

Correspondence should be addressed to Haihua Liang; lh@csig.edu.cn

Received 28 September 2017; Revised 7 April 2018; Accepted 6 June 2018; Published 8 July 2018

Academic Editor: Zheng Yan

Copyright © 2018 Haihua Liang et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

This paper proposes a novel image retrieval scheme over encrypted cloud data, which achieves high efficiency and confidentiality. For the purpose of improving search efficiency, an index tree is often deployed in the image retrieval scheme. Meanwhile, the confidentiality of the sensitive cloud data, such as outsourced images, index tree, and query request, is also a key issue. Firstly, a balanced binary clustering algorithm is exploited over the integrated image features composed of basic features, such as HSV histogram and DCT histogram, yielding a balanced binary tree (BBT). In particular, due to the adoption of a balanced index tree, our scheme can achieve logarithmic search time. Secondly, the secure inner product is employed to encrypt the index vector and query feature. Finally, to resist the statistical attack of the frequency distribution of the retrieved results, we copy the database and merge the subtree of encrypted BBT to blind the search results. Security analysis and experimental results show that the proposed scheme is secure and efficient.

1. Introduction

With the arrival of the cloud computing, secure image retrieval on the cloud has attracted more and more attention in recent years [1]. Due to high convenience and economic savings, both enterprises and individuals tend to employ the cloud to store and manage their sensitive data, such as photo albums and personal health documents. To ensure the confidentiality of outsourced data, the common approach is to encrypt the data before it is outsourced to the cloud. However, the traditional encryption may make the basic data operations unfeasible, for example, the information retrieval of encrypted data. In the ciphertext scenario, how to achieve an efficient retrieval while protecting the privacy of customers becomes a challenging problem.

At present, there are three main issues that restrict the development of information retrieval in the encrypted domain. The first issue is to realize searchable functionality on encrypted data and achieve the same precision as

plaintext data. Certainly, a naive approach is to download all the ciphertext, decrypt them, and search locally in the plaintext. However, it will incur heavy cost of bandwidth and computation. To address this issue, cryptographic techniques, such as homomorphic encryption [2] and multiparty computation, can be employed to encrypt the plaintext data and support search operation in the ciphertext. However, the above methods are concerned more with data confidentiality than retrieval efficiency, and the cost is expensive in practical applications. In the contrary, some efficient techniques, such as order-preserving encryption (OPE) [3, 4], randomized hash functions [5–7], and asymmetric scalar-product-preserving encryption (ASPE) [8], are widely adopted. The reason is that they take both the data confidentiality and retrieval efficiency into account. The second issue is that although plaintext content of the encrypted data is not leaked in the above schemes, some statistical information, such as the request frequency of encrypted query (*i.e.*, query pattern) or the access frequency of encrypted result (*i.e.*,

access pattern), may leak the privacy of query user. Oblivious RAMs [9] is a solution to protect the access pattern, but not practical enough. The third issue is that a retrieval scheme of linear efficiency is not desirable, because the search time will increase as the dataset becomes larger. In fact, secure information retrieval is commonly used for images or documents stored in a cloud server. We detail them as follows.

On the one hand, abundant works have been put forward to achieve secure retrieval in the encrypted documents. For example, Boolean search based on the single keyword is separately proposed in symmetric key setting [11] and public key setting [12]. Since similarity search is more practical than Boolean search, multi-keyword ranked search [13] is studied to enrich search functionality and improve result accuracy, where each document is associated with an index vector. Each element of the vector indicates whether a keyword exists or represents its “term frequency (TF) $\times$ inverse document frequency (IDF).” Then, the k -nearest neighbor is found by comparing the cosine similarity between the query vector and all index vectors, which is linear efficiency. In order to improve retrieval efficiency, a few works based on index tree are proposed. For example, Sun et al. [14] present a tree-based search scheme that constructs index vectors of all documents as an MDB-tree. It achieves sublinear search efficiency via setting the prediction threshold for each level of the index tree. Although a tighter prediction value can obtain logarithmic search efficiency, the result accuracy is sacrificed at the same time. Moreover, Xia et al. [15] build a KBB-tree from a bottom-up approach. In KBB-tree, the element of internal node vector is the maximum value of the corresponding position of its child node vectors. A “Greedy-Depth-First Search” algorithm is executed to find k most relevant leaf nodes, which are stored in an RList. If the correlation score between the query vector and the internal node vector is smaller than the minimum score in the RList, the subtree of the internal node does not need to be searched. Thus, this scheme can also achieve sublinear efficiency.

On the other hand, some works have been proposed for encrypted image retrieval. In [16], a privacy-enhanced face recognition is realized with a help of Paillier homomorphic encryption (HE). The drawback is that the adoption of HE incurs heavy cost of computation and communication. To be practical, Lu et al. propose a secure content-based image retrieval (CBIR) scheme based on feature/index randomization [17] or min-Hash [5]. Meanwhile, the performance comparison between homomorphic encryption and distance-preserving randomization is studied in [18]. Due to the one-way and binary property of hash code, secure CBIR that exploits the hash function to encrypt features is effective and efficient in the large-scale database [6]. However, the access pattern is leaked in the above schemes. To solve this issue, Weng et al. [7] omit certain bits of the hash code of query image. As a result, the cloud returns all possible candidates to the user. Therefore, the query pattern and access pattern are protected. But the user is involved to compare the features of candidates and obtain an accurate result. Also, it is difficult to generate hash codes that uniformly distributed in the feature space. In further, under the vector space model, there is only a handful of works that support efficient index structure.

For example, Xia et al. [19] use local-sensitive hash (LSH) to construct a prefilter table, but a refinement of the candidate results is also a linear comparison. Thus, it just achieves sublinear search efficiency. Yuan et al. [20] employ k -means to build an index tree. Since k -means is not a balanced clustering algorithm, it is inevitable to generate an index tree of skewed hierarchies. Subsequently, due to uneven depth in different parts of index tree, the search efficiency tends to be sublinear. In short, under the vector space model, the requirement for secure and efficient image retrieval mechanisms remains open up to date.

Besides the above schemes that based on the encrypted feature, the secure retrieval scheme directly based on encrypted images has been proposed. It is designed for the widely used JPEG image. For example, Zhang and Cheng first use DC histogram preserved in the encrypted image as a feature [21] and then exploit AC histogram [22] and a new block feature [23] for retrieval. However, the efficiency of linear comparison or block-wise comparison is unacceptable. Therefore, the construction of secure index tree is necessary.

In this paper, we propose an efficient image retrieval scheme over encrypted cloud data, which based on a balanced index tree, *i.e.*, balanced binary tree (BBT). We introduce a novel balanced binary clustering (BBC) algorithm and use integrated features to build the BBT. Owing to the fact that this paper realizes similarity search, we tend to utilize global feature (such as color, shape) instead of accurate local feature (such as SIFT, BRIEF). Hence, our integrated feature is composed of widely used HSV histogram and DCT histogram [5, 21, 22, 24]. We focus on the construction of index tree and protection of query pattern and access pattern. As for the image encryption, we adopt partial image encryption technique [23, 25, 26] for JPEG format image and mature cryptographic ciphers (such as AES) for the other format images. Note that the key distribution is an independent issue and out of the scope of this paper. Our main contributions lie in three aspects:

(1) We combine HSV histogram in the color space and DCT histogram in the transform domain as an integrated feature. To improve the search efficiency, we construct a BBT for the image database. The BBT is built from a top-down approach. Since the hierarchical index tree is approximately balanced, the scheme can achieve logarithmic search time.

(2) By incorporating secure inner product (*i.e.*, ASPE), the BBT is encrypted as secure BBT (SBBT) by a secret key, and query feature is encrypted. The database image is encrypted by another secret key. After that, the sensitive data in the cloud is protected; meanwhile, the searchable functionality of the encrypted feature is still valid. Further, the random number extended in the query feature can blind the query pattern.

(3) To blind the search result, we copy the database and merge the subtree to rebuild SBBT. The reduced SBBT and the weighted factor of the integrated feature make the search result of one query image not unique. Therefore, the query privacy about access pattern is protected.

The rest of this paper is organized as follows. In Section 2, we introduce system model, threat model, and design goals. Also, common notations are defined. In Section 3, we propose

FIGURE 1: A secure and efficient image retrieval scheme over encrypted data.

BBC algorithm, use BBC to build a BBT, and adopt ASPE to encrypted BBT. Section 4 presents and discusses experimental results. Finally, we conclude the paper in Section 5.

2. Problem Formulation

2.1. System Model. The system contains three parties: the data owner, the cloud server, and authorized users, as illustrated in Figure 1.

The owner first extracts feature vectors, such as HSV histogram and DCT histogram, from each image in his database. They are concatenated into an integrated feature. After the feature extraction, the plaintext image is encrypted as the ciphertext by a secret key. For improving search efficiency, the owner builds an index tree based on the integrated features. Then, he utilizes another secret key to encrypt the index tree. The leaf node of the secure index tree is associated with the encrypted image. Afterward, the owner outsources the encrypted database images and the secure index tree to the cloud. In addition, the owner sends secret keys to each authorized user through a secure channel.

The authorized user extracts feature vectors from the query image. To fetch encrypted images in the secure index tree, he encrypts the integrated feature of a query image and submits it to the cloud.

The cloud stores the encrypted database images and secure index tree from the owner. After receiving an encrypted query feature, the cloud executes the search operation in the secure index tree. After that, he returns encrypted images that associated with retrieved leaf node to the user.

2.2. Threat Model. We assume that three parties follow a semihonest model, in which any two parties do not collude to share information. However, even if the cloud does not

know the content of the encrypted query, it may count the query frequency of same encrypted request and the access frequency of the same returned result. Thus, depending on the available information to the cloud, we consider two threat models here.

Known Ciphertext Model. The cloud only knows encrypted database, secure index tree, and the encrypted query.

Known Background Model. In this stronger model, the cloud knows background knowledge about the frequency distribution of the query images. For example, Trump's photos are very popular in the presidential campaign. The cloud can conduct a statistical attack to identify certain image via the access frequency of the returned images.

2.3. Design Goals. To enable secure image retrieval over encrypted cloud data under the above models, our scheme should achieve the following design goals.

Effectiveness. The scheme is designed to support search over encrypted data. The result is as accurate as the search in the plaintext domain.

Efficiency. Under vector space model, we structure the feature vectors of image database as an SBBT. Instead of linear or sublinear search efficiency, our scheme will achieve logarithmic search time.

Privacy. The goal is to protect the privacy of owner and user, which is summarized as follows. (1) The database and its index tree confidentiality: namely, the image and its index tree exist in the ciphertext form in the cloud. (2) The query privacy: it is not possible to determine whether two requests are from the same query image. Note that to ease the overload of computation and communication, our goals are not to fully protect access pattern, *i.e.*, the search path in the SBBT.

2.4. Notations. Assume that the total number of images in the database is N . We define common notations as follows:

- (i) $\mathbf{D}$: the database, namely, the set of images
- (ii) $\mathbf{C}$: the ciphertext form of $\mathbf{D}$ stored in the cloud
- (iii) $\mathbf{p}_i^1$: the normalized feature, such as HSV histogram from the i th image, $1 \leq i \leq N$, it is a d_1 -dimensional row vector
- (iv) $\mathbf{p}_i^2$: the normalized feature, such as DCT histogram from the i th image, $1 \leq i \leq N$, it is a d_2 -dimensional row vector
- (v) $\vec{\mathbf{p}}_i$: the integrated feature, $\vec{\mathbf{p}}_i = [\alpha \mathbf{p}_i^1, (1 - \alpha) \mathbf{p}_i^2]$, a weighted factor $\alpha \in [0, 1]$
- (vi) $\mathbf{v}$: the index vector in each node of BBT, it is generated from $\vec{\mathbf{p}}_i$
- (vii) $\mathbf{I}$: the H -height BBT for the whole image database $\mathbf{D}$, $H \approx \lceil \log_2^N \rceil$
- (viii) $\mathbf{V}$: the ciphertext index which corresponds to plaintext $\mathbf{v}$
- (ix) $\mathbf{T}$: the ciphertext form of $\mathbf{I}$
- (x) $\mathbf{q}^1$: the normalized feature of the query image, such as HSV histogram
- (xi) $\mathbf{q}^2$: the normalized feature of the query image, such as DCT histogram
- (xii) $\vec{\mathbf{q}}$: the query vector, $\vec{\mathbf{q}} = [\beta \mathbf{q}^1, (1 - \beta) \mathbf{q}^2]$, a weighted factor $\beta \in [0, 1]$
- (xiii) $\mathbf{M}$: the encryption key $\mathbf{M} \in \mathbb{R}^{(d+1) \times (d+1)}$ is a random invertible matrix, $d = d_1 + d_2$
- (xiv) $\mathbf{Q}$: the ciphertext query, corresponding to plaintext $\vec{\mathbf{q}}$

To find the k -nearest neighbor of the query image, we adopt “inner product similarity” to evaluate the distance among the features. Specifically, $\vec{\mathbf{p}}_i$ is an integrated feature of the database image, including HSV histogram and DCT histogram. $\vec{\mathbf{q}}$ is the integrated feature of the query image. The final similarity score is obtained by $\vec{\mathbf{p}}_i \cdot \vec{\mathbf{q}}$. Based on the score, the most relevant images in the database are returned to the user.

3. Proposed Scheme

In order to achieve logarithmic search efficiency, we first build a balanced index tree for the plaintext image database, in which a novel balanced binary clustering algorithm is proposed. Then, to ensure the confidentiality of index vector $\mathbf{v}$ and query vector $\vec{\mathbf{q}}$, the secure inner product computation is employed to encrypt the vector $\mathbf{v}$ and $\vec{\mathbf{q}}$. Also, we adopt a partial image encryption to encrypt database images. Finally, we improve the scheme to enhance privacy protection in the known background model.

3.1. BBC Algorithm: Balanced Binary Clustering. To build the balanced index tree, we need a balanced binary clustering (BBC) algorithm to find a separating vector $\mathbf{v}$ which divides each dataset into two equal (or nearly equal) clusters. However, traditional k -means aims at finding cluster centers so that the sum of the distance between each point and its nearest center is maximized, rather than the equal size of clusters. Thus, frequency sensitive competitive learning (FSCL) [27] is introduced to improve k -means. FSCL is a conscience type algorithm that takes the size of clusters as a weight factor which makes the bigger cluster less likely to win more points in next iteration. It can converge to a local minimum [28]. Although the difference of cluster centers can be used as a separating vector $\mathbf{v}$, FSCL does not ensure that the k clusters are equally sized. In fact, it is experienced as very unstable in high-dimensional vector spaces [29]. On the contrast, balanced k -means [30] can split a dataset into two equal-sized clusters by finding a perfect matching in a bipartite graph. But it does not generate the separating vector $\mathbf{v}$. Thus, we propose a novel BBC to find a separating vector that makes the size of two clusters equal. To our best knowledge, this work is the first endeavor to combine FSCL and balanced k -means to achieve balanced clustering. The detail of BBC algorithm is elaborated in the following.

Given n points $\{\vec{\mathbf{p}}_i\}_{i=1}^n \in \mathbb{R}^d$, at the t th iteration, the size of cluster is $\tau_k^{(t)} \geq 0$ and the cluster center is $\mathbf{C}_k^{(t)}$, $k = 1, 2$. The BBC algorithm works as follows.

Initialization. We adopt k -means++ to choose two points from $\{\vec{\mathbf{p}}_i\}_{i=1}^n$ as initial centers $\mathbf{C}_k^{(1)}$, $k = 1, 2$.

First, uniformly choose an initial center $\mathbf{C}_1^{(1)}$ from $\{\vec{\mathbf{p}}_i\}_{i=1}^n$.

Second, choose the next initial center $\mathbf{C}_2^{(1)}$ from $\{\vec{\mathbf{p}}_i, \vec{\mathbf{p}}_i \neq \mathbf{C}_1^{(1)}\}_{i=1}^n$ with probability $D(\vec{\mathbf{p}}_i, \mathbf{C}_1^{(1)}) / \sum D(\vec{\mathbf{p}}_i, \mathbf{C}_1^{(1)})$, $D(\vec{\mathbf{p}}_i, \mathbf{C}_1^{(1)}) = 1 - (\vec{\mathbf{p}}_i \cdot \mathbf{C}_1^{(1)}) / \|\vec{\mathbf{p}}_i\| \|\mathbf{C}_1^{(1)}\|$.

Finally, according to the distance to the initial centers, other points are divided into two categories CL_k , the size of which is $\tau_k^{(1)}$, $k = 1, 2$.

Cluster Assignment. Let new $T_{i,k}^{(t)}$ be the solution of optimization problem as follows.

$$\min J(T_{i,k}^{(t)}) = \sum_{k=1}^2 \left(\tau_k^{(t)} \cdot \sum_{i=1}^n T_{i,k}^{(t)} D(\vec{\mathbf{p}}_i, \mathbf{C}_k^{(t)}) \right) \quad (1)$$

$$\text{s.t.} \quad \begin{cases} T_{i,k}^{(t)} \in \{0, 1\}, & i = 1, \dots, n, k = 1, 2 \\ \sum_{k=1}^2 T_{i,k}^{(t)} = 1, & i = 1, \dots, n \\ \sum_{i=1}^n T_{i,k}^{(t)} \geq \left\lfloor \frac{n}{2} \right\rfloor, & k = 1, 2 \end{cases} \quad (2)$$

where $D(\vec{\mathbf{p}}_i, \mathbf{C}_k^{(t)}) = 1 - (\vec{\mathbf{p}}_i \cdot \mathbf{C}_k^{(t)}) / \|\vec{\mathbf{p}}_i\| \|\mathbf{C}_k^{(t)}\|$. A smaller value of $D(\vec{\mathbf{p}}_i, \mathbf{C}_k^{(t)})$ indicates that the two vectors $\vec{\mathbf{p}}_i, \mathbf{C}_k^{(t)}$ are more similar.

Inspired by balanced k-means, we first turn (1) as a linear assignment problem (LAP) [31]. The cost matrix is defined as

$$CE = \begin{bmatrix} D\langle \vec{\mathbf{p}}_1, \mathbf{C}_k^{(1)} \rangle & \cdots & D\langle \vec{\mathbf{p}}_n, \mathbf{C}_k^{(1)} \rangle \\ D\langle \vec{\mathbf{p}}_1, \mathbf{C}_k^{(2)} \rangle & \cdots & D\langle \vec{\mathbf{p}}_n, \mathbf{C}_k^{(2)} \rangle \\ \vdots & \vdots & \vdots \\ D\langle \vec{\mathbf{p}}_1, \mathbf{C}_k^{(1)} \rangle & \cdots & D\langle \vec{\mathbf{p}}_n, \mathbf{C}_k^{(1)} \rangle \\ D\langle \vec{\mathbf{p}}_1, \mathbf{C}_k^{(2)} \rangle & \cdots & D\langle \vec{\mathbf{p}}_n, \mathbf{C}_k^{(2)} \rangle \end{bmatrix}, \quad (3)$$

where $CE \in \mathbb{R}^{2\lceil n/2 \rceil \times n}$. If it is not a square matrix, a column of zeros is added to make it square. Then, we adopt Volgenant-Jonker (VJ) algorithm [32, 33] to find a perfect matching in a bipartite graph. Finally, if the vector $\vec{\mathbf{p}}_i$ is assigned to center $\mathbf{C}_k^{(t)}$, $T_{i,k}^{(t)} = 1$ and 0 otherwise.

Cluster Update. Update the cluster centers $\mathbf{C}_k^{(t+1)}$, the “selection” variables $T_{i,k}^{(t+1)}$, and the size of clusters $\tau_k^{(t+1)}$ as follows:

$$\begin{aligned} \mathbf{C}_k^{(t+1)} &= \frac{\sum_{i=1}^n T_{i,k}^{(t)} \vec{\mathbf{p}}_i}{\sum_{i=1}^n T_{i,k}^{(t)}}, \quad k = 1, 2 \\ T_{i,1}^{(t+1)} &= 1, \\ T_{i,2}^{(t+1)} &= 0, \\ &\quad \text{if } D\langle \vec{\mathbf{p}}_i, \mathbf{C}_1^{(t+1)} - \mathbf{C}_2^{(t+1)} \rangle \leq 0 \\ T_{i,1}^{(t+1)} &= 0, \\ T_{i,2}^{(t+1)} &= 1, \\ &\quad \text{if } D\langle \vec{\mathbf{p}}_i, \mathbf{C}_1^{(t+1)} - \mathbf{C}_2^{(t+1)} \rangle > 0 \\ \tau_k^{(t+1)} &= \sum_{i=1}^n T_{i,k}^{(t+1)}, \quad k = 1, 2. \end{aligned} \quad (4)$$

Cluster assignment and cluster update are repeated until $\tau_k^{(t+1)} \leq 2^{\lceil \log_2 n \rceil - 1}$ or $\mathbf{C}_k^{(t+1)} = \mathbf{C}_k^{(t)}$, $k = 1, 2$.

Output. The above three steps run several times with different initial centers $\mathbf{C}_1^{(1)}, \mathbf{C}_2^{(1)}$.

Owing to the fact that different initial centers may generate different results, the result that the sizes of two clusters are most similar is chosen as a final output. The size similarity of two clusters is defined as

$$s(\tau_k^{(t+1)}) = \begin{cases} 0, & \prod_{k=1,2} \tau_k^{(t+1)} = 0 \\ \frac{\min\{\tau_k^{(t+1)}, k=1,2\}}{\max\{\tau_k^{(t+1)}, k=1,2\}}, & \prod_{k=1,2} \tau_k^{(t+1)} \neq 0. \end{cases} \quad (5)$$

Finally, the output is

$$\begin{aligned} \vec{\mathbf{p}}_i \in CL_k &\quad \text{if } T_{i,k}^{(t+1)} = 1, \quad i = 1, \dots, n, \quad k = 1, 2 \\ \mathbf{v} &= \left[\frac{\mathbf{C}_1^{(t)}}{\|\mathbf{C}_1^{(t)}\|} - \frac{\mathbf{C}_2^{(t)}}{\|\mathbf{C}_2^{(t)}\|} \right]. \end{aligned} \quad (6)$$

The dataset of n points can be divided into two equal (nearly equal) clusters by the separating vector $\mathbf{v}$.

3.2. UBBT: Unencrypted Balanced Binary Tree. In this subsection, we build the BBT for the whole database. Firstly, we generate an integrated feature $\vec{\mathbf{p}}_i$ for each image in the database. Then, an index vector $\mathbf{v}$ of the root node or internal node of the BBT is generated by a top-down approach. BBC algorithm generates $\mathbf{v}$ to separate image features into two clusters of nearly equal size, i.e., CL_1, CL_2 . Namely,

$$\begin{aligned} \vec{\mathbf{p}}_i \cdot \mathbf{v} &\geq 0 & \text{if } \vec{\mathbf{p}}_i \in CL_1 \\ \vec{\mathbf{p}}_i \cdot \mathbf{v} &< 0 & \text{if } \vec{\mathbf{p}}_i \in CL_2. \end{aligned} \quad (7)$$

Algorithm 1 recursively divides feature vectors of the database images into two clusters until no cluster has more than two feature vectors. The hierarchical index tree is shown in Figure 2. The level of the root node is 0 and the level of the leaf node is H . Thus, the height of entire index tree is H . The subtree that the root node is at h th level of the index tree has the height of $H - h$. The root and internal node u in our BBT are defined as follows:

$$u = \langle ID, \mathbf{v}, uL, uR \rangle \quad (8)$$

where ID is the identity of node, $\mathbf{v}$ is the index vector dividing images on the leaf nodes into two clusters CL_1 and CL_2 , and uL and uR are the pointers to the left and right child node. If u is a leaf node, it is defined as

$$u = \langle ID, iL, \dots, iL \rangle. \quad (9)$$

where iL points to the images.

The search process starts from the root node of UBBT. If $\vec{\mathbf{q}} \cdot \mathbf{v} \geq 0$, we execute iterative retrieval upon the left subtree, otherwise upon the right subtree until obtaining the leaf node.

3.3. SBBT: Secure Balanced Binary Tree. In UBBT scheme, the leaf node is retrieved by the sign of the inner product $\vec{\mathbf{q}} \cdot \mathbf{v}$. In order to protect the sensitive data $\vec{\mathbf{q}}$ and $\mathbf{v}$, we use the secure inner product (i.e., ASPE) [8] to encrypt the vectors and obtain the same result as the plaintext.

First of all, the owner generates a random invertible matrix $\mathbf{M} \in \mathbb{R}^{d \times d}$ to encrypt $\mathbf{v}$. The user also use it to encrypt $\vec{\mathbf{q}}$, i.e.,

$$\begin{aligned} \mathbf{V} &= \mathbf{vM}^{-1} \\ \mathbf{Q} &= \mathbf{M}\vec{\mathbf{q}}^T, \end{aligned} \quad (10)$$

where $\mathbf{V}$ is the ciphertext index and $\mathbf{Q}$ is the ciphertext query. In addition, the JPEG images are encrypted via a partial image encryption [22, 25]. Therefore, index tree $\mathbf{I}$ is encrypted as $\mathbf{T}$. The leaf node of $\mathbf{T}$ is associated with the encrypted images. Then, they are outsourced to the cloud for secure image retrieval. When retrieving, the user submits $\mathbf{Q}$ of the query image to the cloud. Since $\mathbf{VQ} = \vec{\mathbf{q}} \cdot \mathbf{v}$, the cloud can

Input: $\{\vec{p}_i\}$, $1 \leq i \leq N$; Balanced Binary Clustering, *i.e.*, BBC.

Output: Balanced Binary Tree **I**.

- (1) $h = 0$ //the height of node
- (2) $ChildrenGen(\{\vec{p}_i\}_{1 \leq i \leq N}) \rightarrow$ root node
- (3) **function** $ChildrenGen(\text{vectors } \{\vec{p}_i\})$
- (4) $BBC(\{\vec{p}_i\}) \rightarrow$ two clusters CL_1, CL_2 ,
separating vector as the index v
- (5) **if** the size of $CL_1 \geq 2$ **then**
- (6) $ChildrenGen(\{\vec{p}_i\}_{i \in CL_1}) \rightarrow$ internal node
- (7) **else**
- (8) $\{i\text{th image}\}_{i \in CL_1} \rightarrow$ leaf node
- (9) **end if**
- (10) **if** the size of $CL_2 \geq 2$ **then**
- (11) $ChildrenGen(\{\vec{p}_i\}_{i \in CL_2}) \rightarrow$ internal node
- (12) **else**
- (13) $\{i\text{th image}\}_{i \in CL_2} \rightarrow$ leaf node
- (14) **end if**
- (15) $h = h + 1$
- (16) **return** v, CL_1, CL_2
- (17) **end function**

ALGORITHM 1: Building balanced binary tree.

FIGURE 2: The hierarchical structure of unencrypted BBT.

use Q to execute an effective search on secure index tree T . Therefore, the retrieved result and efficiency in the ciphertext would always be in line with that in the plaintext. Note that the more details of ASPE are described in [8].

Unfortunately, the relationship between $\vec{q}$ and Q is deterministic. According to the same Q or VQ , the cloud can count the query frequency of $\vec{q}$. Further, according to the same result, the cloud can also count the access frequency of $\vec{q}$. Therefore, if the query image and $\vec{q}$ is a one-to-one relationship, the proposed scheme cannot resist the statistical attack in the known background model.

To break the one-to-one relationship between the query image and ciphertext query Q , we extend $\vec{q}$ and v as follows:

$$\begin{aligned} \vec{q} &= \theta [\gamma, \vec{q}] \\ v &= [0, v], \end{aligned} \quad (11)$$

where the random numbers $\theta > 0$, $\gamma \gg 1$. Thus, the same query image can generate different ciphertext query Q by setting θ and γ . Meanwhile, $VQ = \theta(\vec{q} \cdot v)$ is blinded by positive θ . The cloud cannot get available statistical

FIGURE 3: (a) a query image, (b) one encrypted version, and (c) another encrypted version.

FIGURE 4: In Corel database [10], except for (a),(e) a query image, the most similar image by the retrieval based on (b),(f) HSV histogram, (c),(g) DCT histogram, or (d),(h) an integrated feature.

information from the ciphertext query $\mathbf{Q}$ and the result $\mathbf{VQ}$. However, the sign of $\mathbf{VQ}$ remains the same even if γ and θ are different, so the retrieved result and search path remain the same. The request frequency of a query image can be inferred from the access frequency of the same return image.

To overcome the issue, we first copy the image database. The owner uses partial image encryption [23] to protect plaintext image. According to different encryption keys, same plaintext image can be encrypted as different encrypted images, as shown in Figure 3. Meanwhile, the owner uses the embedding key to hide which encryption key was used [25]. Then, the owner shares encryption keys and the embedding key with authorized users through a secret channel. After obtaining an encrypted image, the user uses the embedding key to extract hidden data and select true encryption key to recover the plaintext image.

Second, different encrypted images derived from the same plaintext image correspond to different integrated features. For the diversity of the integrated feature, we use HSV histogram and DCT histogram as basic features. The reason is that HSV histogram in color space and DCT histogram in transform domain are image features from different perspectives. As shown in Figure 4, different features

of the query image can still retrieve a similar image. For ease of description, we define the new ciphertext database as $\mathbf{C}'$. Then, we set $\mathbf{C}$ corresponding to the feature $\vec{\mathbf{p}}_i = [\alpha_1 \mathbf{p}_i^1, (1 - \alpha_1) \mathbf{p}_i^2]$ and $\mathbf{C}'$ corresponds to the feature $\vec{\mathbf{p}}_i' = [\alpha_2 \mathbf{p}_i^1, (1 - \alpha_2) \mathbf{p}_i^2]$. Finally, we use proposed BBC algorithm to build an index tree $\mathbf{I}$ from $\{\vec{\mathbf{p}}_i, \vec{\mathbf{p}}_i', \mathbf{C}, \mathbf{C}'\}$ and employ ASPE to encrypt the plaintext tree as a ciphertext tree $\mathbf{T}$.

To further enhance query privacy, we reduce the height of the index tree $\mathbf{T}$ and return multiple encrypted images to the query user at the one time. Specifically, we set the internal node to be a new leaf node. Then, the new leaf node contains all the encrypted images in the subtree that the internal node of $\mathbf{T}$ is a root node. Due to the fact that the index tree is balanced, leaf nodes of reduced index tree can always point to multiple encrypted images. Please note that some images in a subtree are similar, which is a compromise with effective retrieval.

3.4. Security Analysis. We analyze SBBT concerning the privacy requirements in the design goals:

(1) The database and its index are confidential. In SBBT, index vectors $\mathbf{v}$ of BBT are encrypted as $\mathbf{V}$ by the high-dimensional random matrix which is kept confidential to the

FIGURE 5: The average correlation of HSV histogram (left) and DCT histogram (right) under same category and different categories.

cloud. For images in the database, the partial encryption can protect their confidentiality [23, 25].

(2) Since the feature vector $\vec{q}$ of the query image is extended by random numbers γ and θ , it can generate different ciphertext queries $\mathbf{Q}$. The cloud cannot identify the same query image from its different ciphertext queries. The query pattern is protected.

(3) According to different β , different features of same query image can retrieve different images. When β is equal to α_1 or α_2 , different results can be recovered to the same query image. Thus, the retrieved result and path of the same query image are not unique but the retrieval is still effective. The cloud cannot link the different returned image to the same query image. The access pattern is protected.

(4) In addition, since the height of SBBT is reduced, one leaf node will point to multiple encrypted images. It further breaks the one-to-one relationship between the query image and the return images. The access pattern is further protected.

In short, the SBBT is secure in the known ciphertext model and can resist the statistical attack in the known background model.

4. Performance Analysis

In this section, we show the experimental results and efficiency analysis of the proposed scheme. As we know that JPEG is the most commonly used image format and accounting for up to 95% of images on the web [34], so we adopt the Corel database [10] for the similarity search. The database contains 1000 JPEG images which belong to 10 categories: African, Beach, Architecture, Buses, Dinosaurs, Elephants, Flowers, Horses, Mountain, and Food. Each of them includes 100 images. As [5], we split an image into 256 blocks and extract hue, saturation, and value of brightness

of each block. Then, we perform hierarchical clustering of block features to construct the “bag of word” model. The usage frequency of the visual word is the HSV histogram of the image. As [22], DC/AC coefficients of image blocks are quantized into different bins and the DCT histogram is the usage frequency of quantized DC/AC coefficients. Thus, the color feature $\mathbf{p}_i^1$ and texture feature $\mathbf{p}_i^2$ are set as the 991-dimensional HSV histogram and 780-dimensional DCT histogram, respectively.

4.1. Parameters Setting. First of all, we should set parameters α_1 of $\vec{\mathbf{p}}_i$ and α_2 of $\vec{\mathbf{p}}_i'$ to construct an index tree. Because the integrated features compose of HSV histograms and DCT histogram, we first evaluate average histogram similarity of the images in the same category and different categories (denoted as corr-intra and corr-inter, respectively), as shown in Figure 5. We can see that the value of corr-intra is larger than corr-inter; thus, images of different categories can be distinguished by each histogram. According to the difference between corr-intra and corr-inter (*i.e.*, corr-diff in Figure 5), we set $\alpha_1 = 0.1$ for the integrated feature $\vec{\mathbf{p}}_i$ of the database $\mathbf{C}$. The purpose is to make the HSV histogram and DCT histogram have the same contribution to the search results.

For the parameter α_2 of $\vec{\mathbf{p}}_i'$, we hope that the index tree of database $\{\mathbf{C}, \mathbf{C}'\}$ can better protect the access pattern. We use two different metrics to measure the privacy protection achieved by different α_2 . The two metrics are the number of different image categories in each subtree and the category entropy of each subtree. Figure 6 shows the average number of image categories contained in the subtree of different levels. Because the index tree of the database based on different α_2 has same height of 11 (*i.e.*, $\lceil \log_2^{2000} \rceil$), the proposed BBC algorithm performs well. However, we can see from

TABLE 1: Average category entropy of each level.

Parameter	Level of the Index Tree										
α_2	1	2	3	4	5	6	7	8	9	10	11
0.2	2.99	2.49	2.1	1.75	1.38	1.07	0.81	0.62	0.44	0.21	0
0.3	3.2	2.82	2.07	1.81	1.43	1.18	0.91	0.7	0.48	0.24	0
0.4	3.32	2.88	2.28	1.8	1.53	1.28	1.02	0.78	0.54	0.28	0
0.5	3.32	3.03	2.31	1.83	1.56	1.27	1.03	0.79	0.55	0.3	0
0.6	3.32	2.91	2.37	1.96	1.65	1.41	1.11	0.89	0.61	0.32	0
0.7	3.32	2.6	2.1	1.71	1.46	1.24	1.03	0.81	0.57	0.29	0
0.8	3.32	2.92	2.32	1.85	1.52	1.26	1.04	0.81	0.57	0.3	0
0.9	3.32	2.92	2.25	1.78	1.49	1.27	1.03	0.83	0.58	0.3	0

FIGURE 6: The average number of image categories.

Figure 6 that the subtree near the leaf node contains a similar number of image categories. We will use category entropy to measure the privacy. The category entropy, which is defined as $-\sum_i (c_i / \sum_i c_i) \log_2 (c_i / \sum_i c_i)$ (c_i is the number of existing i category), is used to measure the randomness of image categories in the subtree. From Table 1, we know that the index tree of $\alpha_2 = 0.6$ has the highest average entropy at the 3th-10th level, even its average category number at 3th-6th level is not the biggest in Figure 6. Thus, we set the weighted factor $\alpha_2 = 0.6$ for the integrated feature $\vec{p}_i'$ of database C' .

4.2. Index Tree Construction. In order to prove that our algorithm can classify feature vectors in a balanced manner, we use average similarity of the cluster size to compare the clustering performance of proposed BBC algorithm, balanced k-means (*i.e.*, Bkmeans), k-means++, and FSCL. The average similarity based on (5) is defined as $S = s(\tau_k^{(t+1)})$.

FIGURE 7: The average size similarity of two clusters.

From Figure 7, we can see that the index trees generated by the proposed BBC algorithm, Bkmeans, k-means++, and FSCL have the height of 11, 12, 13, and 16, respectively. The smaller the height of the index tree indicates that the proposed algorithm can classify feature vectors into two clusters more evenly at each level of the index tree. Figure 7 shows that the cluster size of FSCL is the most unbalanced at 10th-11th levels. There are two reasons: the first is that the change of the weight factor affects the stability of the clustering result, and the second is that the value of weight factor impacts the distance between the cluster center and other feature vectors. On contrast, due to the use of conscience weight factor, the proposed algorithm performs better than Bkmeans and k-means++, and the result is much better than FSCL. The reason is that the bipartite graph matching mechanism constrains a balance of the two clusters and restricts the influence of weight factor. In short, the weight factor is still effective in improve the balance of clustering. In fact, above result is also similar in the database of different α_1, α_2 .

TABLE 2: Minimal Dunn index of each level.

Algorithm	Level of the Index Tree															
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
BBC	0.47	0.01	0.01	0.02	0.01	0.01	0.01	0.01	0.01	0.02	∞	-	-	-	-	-
Bkmeans	0.47	0.01	0.02	0.01	0.02	0.02	0.02	0.03	0.04	0.24	0.73	∞	-	-	-	-
Kmeans++	0.21	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.02	0.03	0.22	1.05	∞	-	-	-
FSCL	0.47	0.04	0.05	0.06	0.07	0.1	0.15	0.19	0.23	0.38	0.28	0.58	0.62	0.66	0.95	∞

TABLE 3: Time cost of index tree construction.

Algorithm	BBC	Bkmeans	Kmeans++[20]	FSCL
Time (s)	111	2316	1749	254

In further, we employ Dunn index (DI) [35] to evaluate mentioned clustering algorithms. The Dunn index is defined as

$$DI = \frac{\min_{\forall x \in CL_1, \forall y \in CL_2} \{D\langle x, y \rangle\}}{\max_{k=1,2} \{\max_{\forall x, y \in CL_k} \{D\langle x, y \rangle\}\}}, \quad (12)$$

where the numerator is the minimal distance of vectors in different clusters and the denominator is the largest within-cluster distance. Thus, we can know that when separating two points into two clusters, the denominator is zero and DI is infinite. A high value of DI means good compactness. Table 2 shows the minimal DI of clustering results in each level. In the same level of the index tree, the most of BBC's DI is smaller than other algorithms (i.e., Bkmeans, K-means++, and FSCL). Meanwhile, BBC's DI of the last level is much smaller than other algorithms. Obviously, for the proposed BBC algorithm, the cluster-size similarity is with higher priority than clustering compactness.

The above algorithms are implemented using MATLAB on a PC with an Intel Core i5 3.2 GHz CPU and 16 GB RAM. Time consumption for index tree construction is shown in Table 3. In particular, we execute 30 times clustering with random initial centers and select the result of highest score of cluster-size similarity. Bkmeans is most time consuming because it is not designed to find a vector that splits a dataset evenly. In contrast, proposed BBC costs the least time.

4.3. Retrieval Performance. To evaluate the retrieval performance, each image in the Corel database is selected as a query image. The retrieval performance is evaluated by average precision-recall (PR) curve, i.e.,

$$\begin{aligned} \text{Precision} &= \frac{\# \text{ of returned positive images}}{\# \text{ of all returned images}}, \\ \text{Recall} &= \frac{\# \text{ of returned positive images}}{\# \text{ of all positive images}}. \end{aligned} \quad (13)$$

Figure 8(a) compares the PR curves of the query feature based on different β values. It can be seen that the performance of different β achieves high precision when recall belongs to (0, 0.05). Besides, when only one image is returned, i.e., recall=1%, the precision of $\beta = 0.1$ or 0.6 is 100%. The reason

is that we set β as α_1 or α_2 . Although two integrated features of a query image are different, the retrieved results are different encrypted versions of the same plaintext image. Since the query features, the corresponding results, and search paths are different, the statistical information of query image is protected.

Figure 8(b) shows the retrieval performance of the reduced index trees with different heights. It can be seen that the precision increases with the height of index tree, even if β is set as 0 or 1. Further, Figure 9 shows the average maximum height of a reduced index tree, in which the returned leaf node contains the exact query image. It can be seen that the height of reduced index tree is 11 when β is set as 0.1 or 0.6. If β is set closer to 0.1 or 0.6, the height of reduced index tree is nearer to 11. When the height is 9, the most of exact query images are retrievable in the reduced index tree and one search returns four (2^2) images. Fortunately, as shown in Figure 8(b), the precision is about 70% at the 9th level of the index tree. Thus, the retrieval is still effective.

4.4. Privacy Protection. We consider the privacy protection from two aspects: query pattern and access pattern. Since the query feature is an integrated histogram, i.e., $\vec{q} = [\beta q^1, (1 - \beta)q^2]$. Different β correspond to different query features. In addition, even if β is determined, the feature will be extended by random numbers θ, γ as $\vec{q} = \theta[\gamma, \vec{q}]$. Thus, the encrypted feature is $\mathbf{Q} = \mathbf{M}(\theta[\gamma, \beta q^1, (1 - \beta)q^2])^T$. Obviously, the encrypted feature of a query image is not unique. The query pattern of each query image is protected.

To protect access pattern, the retrieved result of a query image should be not unique. We have built an index tree based on features of original database ($\alpha_1 = 0.1$) and features of the copied database ($\alpha_2 = 0.6$). Then, the integrated features of different β are used to retrieve in the index tree. We know the fact that the returned result is a leaf node of the index tree and different leaf nodes indicate different access paths. So, we define a path privacy to evaluate the protection level of access pattern

$$\text{Privacy} = \frac{\# \text{ of different search paths}}{\# \text{ of all search paths}}. \quad (14)$$

Different search path can be obtained by using the query feature of different β . Figures 10(a) and 10(b) show the path

FIGURE 8: (a) Precision-recall curves of original index tree. (b) Retrieval precision of reduced index trees.

FIGURE 9: Under different β , the average maximum height of a reduced index tree that the query image can retrieve itself.

privacy of different β , where $\beta = 0.1$ and 0.6 are baselines, respectively. It can be seen that the path privacy is 100% when β is set as 0.6 in Figure 10(a) and 0.1 in Figure 10(b). The reason is that retrieval is separately carried out in the left or right subtree of the root node. Figure 10(a) also shows that search paths are in the same subtree of the root node when β is set to 0.6 , 0.8 , or 1 . But the search path of $\beta = 0.8$ or 1 is still different with $\beta = 0.6$ as shown in Figure 10(b). Similarly, when β is set to 0 , 0.0025 , 0.05 , 0.075 , 0.1 , 0.3 , or 0.35 , the search path is in the same subtree of the root node, but still different. Besides, when β is set to 0.4 , the search traverses in both subtrees of the root node. In addition, the path privacy increases with the height of the tree. In short, the

access pattern can be protected by setting different β . Please note that usage frequency of different β can be optimized based on the frequency distribution of the query images.

In summary, Table 4 compares the proposed scheme with other schemes in the terms of search efficiency, query pattern protection, and access pattern protection. We compare the search efficiency of different schemes by theoretical analysis instead of experiments. The reason is that encrypted image retrieval schemes are based on different image databases according to the application. Also, some schemes utilize encrypted image feature, but other schemes employ the hash code of the image feature.

In the schemes under vector space model, such as Yuan [20], a skew index tree leads to $O(\log N \sim N)$ search time. Xia [19] uses local-sensitive hash (LSH) to construct a prefilter table. Then, with corresponding image feature, the cloud carries out a refinement of candidate results. Thus, it is a sublinear scheme. Cheng [22, 23] does not build an index tree, so the retrieval efficiency is $O(N)$. Weng [6, 7] uses a hash table to improve the search efficiency (*i.e.*, $O(1 \sim N)$). In short, the index tree or hash table is a trade-off between space and efficiency. Although Weng's scheme [7] and proposed scheme both protect query pattern and access pattern, the hash code and feature vector are, respectively, used. Specifically, Weng [7] omits certain bits of the hash code of query image and returns multiple candidates to the user for refinement. The user is involved in completing retrieval and obtaining the accurate result. In our scheme, the user can set β as α_1 or α_2 to retrieve exact query image through different paths. Unlike [7], our scheme does not need the user to participate in the feature comparison. However, to protect

FIGURE 10: Path privacy of different β : (a) $\beta = 0.1$ is the baseline and (b) $\beta = 0.6$ is the baseline.

TABLE 4: Comparison of encrypted image retrieval schemes.

Goals	Yuan [20]	Xia [19]	Weng [6]	Weng [7]	Cheng [22, 23]	Proposed
Efficiency	$O(\log N \sim N)$		$O(1 \sim N)$		$O(N)$	$O(\log 2N)$
Query Pattern	✓	✓	×	✓	×	✓
Access Pattern	×	×	×	✓	×	✓

access pattern and alleviate computation burden on the user, we copy the image database and cost more storage space in the cloud. Fortunately, the retrieval efficiency is $O(\log 2N)$.

5. Conclusion

This paper introduces a secure and efficient image retrieval scheme over encrypted cloud data. In this scheme, we propose a novel clustering algorithm, *i.e.*, BBC, to build a balanced index tree, namely, BBT. Thus, our scheme can achieve logarithmic search time. Firstly, to support effective image retrieval in the ciphertext, we employ ASPE to encrypt the index of the image database and feature of the query image. In this case, the proposed SBBT is secure in the known ciphertext model. Secondly, to resist the statistical attack on known background model, we copy the database and build a new SBBT of the original database and the copied database. After that, adjusting the weighted factor of integrated feature, the search results of a query image are different leaf nodes. Finally, through merging the subtree of SBBT, the leaf node of new SBBT contains multiple encrypted images. Therefore, the one-to-one relationship between the query image and the

return image is broken. In short, the SBBT can protect query pattern and access pattern. Furthermore, proposed BBC and SBBT can be used as independent tools for encrypted document retrieval.

In future, we will improve the ASPE to support multiuser scenario, where a dishonest user may try to reveal the encrypted query of other users. It is also a meaningful work to generate a generic, high-semantic feature.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (U1636206, 61602295, 61525203, and 61472235), the Shanghai Dawn Scholar Plan (14SG36), the Shanghai Excellent Academic Leader Plan (16XD1401200), the Natural Science Foundation of Fujian Province (2017J01502), and the Scientific Research Foundation of Fuzhou University (510483).

References

- [1] K. Ren, C. Wang, and Q. Wang, "Security challenges for the public cloud," *IEEE Internet Computing*, vol. 16, no. 1, pp. 69–73, 2012.
- [2] C. Gentry, "Fully homomorphic encryption using ideal lattices," in *Proceedings of the 41st annual ACM symposium on Theory of Computing (STOC '09)*, pp. 169–178, ACM, Bethesda, Md, USA, 2009.
- [3] C. Wang, N. Cao, K. Ren, and W. Lou, "Enabling secure and efficient ranked keyword search over outsourced cloud data," *IEEE Transactions on Parallel and Distributed Systems*, vol. 23, no. 8, pp. 1467–1479, 2012.
- [4] K. Li, W. Zhang, C. Yang, and N. Yu, "Security Analysis on One-to-Many Order Preserving Encryption-Based Cloud Data Search," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 9, pp. 1918–1926, 2015.
- [5] W. Lu, A. Swaminathan, A. L. Varna, M. Wu et al., "Enabling search over encrypted multimedia databases," *Media Forensics and Security*, vol. 7254, 2009.
- [6] L. Weng, L. Amsaleg, A. Morton, and S. Marchand-Maillet, "A privacy-preserving framework for large-scale content-based information retrieval," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 1, pp. 152–167, 2015.
- [7] L. Weng, L. Amsaleg, and T. Furon, "Privacy-preserving outsourced media search," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 10, pp. 2738–2751, 2016.
- [8] W. K. Wong, D. W. Cheung, B. Kao, and N. Mamoulis, "Secure kNN computation on encrypted databases," in *Proceedings of the International Conference on Management of Data and 28th Symposium on Principles of Database Systems (SIGMOD-PODS '09)*, pp. 139–152, Providence, RI, USA, July 2009.
- [9] P. Williams, R. Sion, and B. Carbunar, "Building castles out of mud: practical access pattern privacy and correctness on untrusted storage," in *Proceedings of the 15th ACM Conference on Computer and Communications Security, CCS'08*, pp. 139–148, October 2008.
- [10] <http://wang.ist.psu.edu/docs/related/>.
- [11] D. X. Song, D. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," in *Security and Privacy, 2000. S&P 2000. Proceedings. 2000 IEEE Symposium on IEEE*, pp. 44–55, IEEE, 2000.
- [12] D. Boneh, G. Di Crescenzo, R. Ostrovsky, and G. Persiano, "Public key encryption with keyword search," in *Advances in cryptology—EUROCRYPT 2004*, vol. 3027 of *Lecture Notes in Comput. Sci.*, pp. 506–522, Springer, Berlin, 2004.
- [13] 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.
- [14] W. Sun, B. Wang, N. Cao et al., "Verifiable privacy-preserving multi-keyword text search in the cloud supporting similarity-based ranking," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 11, pp. 3025–3035, 2014.
- [15] Z. Xia, X. Wang, X. Sun, and Q. Wang, "A Secure and Dynamic Multi-Keyword Ranked Search Scheme over Encrypted Cloud Data," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 2, pp. 340–352, 2016.
- [16] Z. Erkin, M. Franz, J. Guajardo, S. Katzenbeisser, I. Lagendijk, and T. Toft, "Privacy-Preserving Face Recognition," in *Privacy Enhancing Technologies*, vol. 5672 of *Lecture Notes in Computer Science*, pp. 235–253, Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.
- [17] W. Lu, A. L. Varna, A. Swaminathan, and M. Wu, "Secure image retrieval through feature protection," in *Proceedings of the ICASSP 2009 - 2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 1533–1536, Taipei, Taiwan, April 2009.
- [18] W. Lu, A. L. Varna, and M. Wu, "Confidentiality-preserving image search: A comparative study between homomorphic encryption and distance-preserving randomization," *IEEE Access*, vol. 2, pp. 125–141, 2014.
- [19] Z. Xia, N. N. Xiong, A. V. Vasilakos, and X. Sun, "EPCBIR: An efficient and privacy-preserving content-based image retrieval scheme in cloud computing," *Information Sciences*, vol. 387, pp. 195–204, 2017.
- [20] J. Yuan, S. Yu, and L. Guo, "SEISA: Secure and efficient encrypted image search with access control," in *Proceedings of the IEEE INFOCOM 2015 - IEEE Conference on Computer Communications*, pp. 2083–2091, Kowloon, Hong Kong, April 2015.
- [21] X. Zhang and H. Cheng, "Histogram-based retrieval for encrypted jpeg images," in *Proceedings of the in Signal and Information Processing (ChinaSIP, 2014 IEEE China Summit International Conference on. IEEE*, pp. 446–449, 2014.
- [22] H. Cheng, X. Zhang, and J. Yu, "AC-coefficient histogram-based retrieval for encrypted JPEG images," *Multimedia Tools and Applications*, vol. 75, no. 21, pp. 13791–13803, 2016.
- [23] H. Cheng, X. Zhang, J. Yu, and Y. Zhang, "Encrypted JPEG image retrieval using block-wise feature comparison," *Journal of Visual Communication and Image Representation*, vol. 40, pp. 111–117, 2016.
- [24] Y. Xu, J. Gong, L. Xiong, Z. Xu, J. Wang, and Y. Q. Shi, "A privacy-preserving content-based image retrieval method in cloud environment," *Journal of Visual Communication Image Representation*, vol. 43, no. C, pp. 164–172, 2017.
- [25] Z. Qian, H. Zhou, X. Zhang, and W. Zhang, "Separable reversible data hiding in encrypted jpeg bitstreams," *IEEE Transactions on Dependable Secure Computing*, no. 99, pp. 1–1, 2016.
- [26] Z. Qian, H. Xu, X. Luo, and X. Zhang, "framework of reversible data hiding in encrypted jpeg bitstreams," *IEEE Transactions on Circuits Systems for Video Technology*, no. 99, pp. 1–1, 2018.
- [27] A. Banerjee and J. Ghosh, "Frequency-sensitive competitive learning for scalable balanced clustering on high-dimensional hyperspheres," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 15, no. 3, pp. 702–719, 2004.
- [28] A. S. Galanopoulos, R. L. Moses, and S. C. Ahalt, "Diffusion approximation of frequency sensitive competitive learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 8, no. 5, pp. 1026–1030, 1997.
- [29] C. T. Althoff, A. Ulges, and A. Dengel, "Balanced clustering for content-based image browsing," in *GI-Informatiktage, 2011., GI-Informatiktage (Informatik-2011)*, Gesellschaft für Informatik, Bonn, Germany.
- [30] M. I. Malinen and P. Fränti, "Balanced K-means for clustering," in *Structural, Syntactic, and Statistical Pattern Recognition*, P. Fränti, G. Brown, M. Loog, F. Escolano, and M. Pelillo, Eds., vol. 8621 of *Lecture Notes in Computer Science*, pp. 32–41, Springer, Berlin, Germany, 2014.
- [31] https://en.wikipedia.org/wiki/Assignment_problem.

- [32] R. Jonker and A. Volgenant, "A shortest augmenting path algorithm for dense and sparse linear assignment problems," *Computing: Archives for Scientific Computing*, vol. 38, no. 4, pp. 325–340, 1987.
- [33] W. Jones, A. Chawdhary, and A. King, "Optimising the Volgenant–Jonker algorithm for approximating graph edit distance," *Pattern Recognition Letters*, vol. 87, pp. 47–54, 2017.
- [34] G. Schaefer, "Fast JPEG compressed domain image retrieval," in *Proceedings of the 5th International Conference on Multimedia Computing and Systems, ICMCS 2016*, pp. 148–150, mar, October 2016.
- [35] https://en.wikipedia.org/wiki/Dunn_index.