Low-Rank Sparse Graph Hashing (LSGH) for Content-based Image Retrieval

Minghua Wan[1,2], Ying Zhao[1], Hai Tan[1], Jun Yin[2], Chengli Sun[3], Jie Xu[3], Mingwei Tang[1], Guowei Yang[4*]

[1]School of Computer Science (School of Intelligent Audit), Nanjing Audit University, Nanjing 211815, China
[2]College of Information Engineering, Shanghai Maritime University, Shanghai 201306, China
[3]School of Computer Science, Guangzhou Maritime University, Guangzhou 510700, China
[4]School of Information Engineering, Nantong Institute of Technology, Nantong 226001, China

Keywords: Content-based image retrieval; Hash learning; Low-rank constraint; Sparse representation; Graph embedding

Citation: Wan M.H., Zhao Y., Tan H., et al.: Low-Rank Sparse Graph Hashing (LSGH) for Content-based Image Retrieval. Data Intelligence, 2026, Vol. 8, Art. No.: 20250086. DOI: https://doi.org/10.3724/2096-7004.di.2025.0086

ABSTRACT

Nowadays, the supervised discrete hashing (SDH) algorithm with low computational complexity and low memory consumption is a hotspot in the field of content-based image retrieval (CBIR). However, when using binary hash codes to represent the original image, most of the image feature information will be lost, resulting in low precision of retrieval. Especially when dealing with data polluted by noise, the performance of the algorithm drops sharply. Therefore, to minimize the loss of feature information as much as possible to improve the precision and strengthen the robustness of the algorithm, this paper proposes a new low-rank sparse graph hashing (LSGH) algorithm. Firstly, based on the SDH algorithm, LSGH applies low-rank function to projection matrix, utilizing global structure of image features to ensure good performance of hash codes. Then, the local structure at the bottom of the data is captured to reduce noise and redundant information between features by using $L_{2,1}$ norm. Secondly, LSGH uses graph embedding to map data from the original dimension to Hamming space, while maintaining similar data structures, making it suitable for exploring complex structure of high dimensional images. Finally, results of experiments on four public datasets show that LSGH is superior to mainstream image hash algorithms under multiple evaluation metrics, and is more robust and accurate in image retrieval and feature extraction.

*Corresponding author: Guowei Yang (E-mail: ygw_ustb@163.com).

© 2026 Chinese Academy of Sciences. Published under a Creative Commons Attribution 4.0 International (CC BY 4.0) license.

1. INTRODUCTION

Content-based image retrieval (CBIR) technology [1-3] has developed rapidly in feature extraction and data storage in recent years, which enables users to identify the input image from large-scale datasets and then query a same or similar image for output quickly and accurately. However, the biggest problem it currently faces is that finding the right image takes a long time. This is mainly caused by two aspects: (1) A large amount of data needs to be retrieved; (2) The calculation of similarity between data points is complex. Related classic algorithms include: image retrieval based on word tree structure, image retrieval based on nearest neighbor graph using greedy algorithm, image retrieval based on vector quantization, which clusters vectors into different classes and approximates the vectors in each class using corresponding class centers, and image retrieval based on hash algorithm [4]. However, when processing high-dimensional images, the precision of the results returned by the first three methods is not high and the time performance is significantly lower. In contrast, by learning to construct a hash function, the hash algorithm projects features from original dimension to Hamming space, and then converts the mapped features into a series of compact binary hash codes [5]. Compared with floating point computation, the time to calculate the Hamming distance between feature vectors is significantly reduced. In addition, the use of hash codes also enormously reduces load pressure on computers and promotes efficiency of large-scale information storage [6].

For example, location sensitive hashing (LSH) algorithm [7] maps data to different hash buckets through randomly generated projection matrix. The higher the similarity of two data, the greater the probability of their projection to the same hash bucket. Spectral hashing (SH) [8] is an unsupervised hash learning algorithm that preserves the similarity distribution between data by constructing a similar matrix and a Laplacian matrix, and forces the ideal hash codes to be balanced and uncorrelated. Anchor graph hashing (AGH) [9] gets anchor points by clustering sample datasets, thus greatly reducing computational complexity while maintaining the manifold structure of data. Both SH and AGH ignore data labels in the process of hash learning, so supervised discrete hashing (SDH) [10] aims to extract and classify features by using label information, and handle the NP hard problem by relaxing binary constraints. The discrete cycle coordinate descent method (DCC) is used to solve and get well-generated hash codes [11], which greatly improves the coding performance. Bilinear iterative quantization (BITQ) [12] replaces traditional single projection matrix with bilinear projection for reducing dimensions of data. Moreover, iteration quantization is adopted to make quantization error be a minimum and generate effective hash codes. By introducing inter-class discriminant matrix and intra-class scattering matrix, weakly supervised discrete discriminant hashing (WDDH) [13] can maintain original neighbor relationship of samples belonging to a class after projection, instead of learning hash codes solely based on given label information, so as to obtain better image retrieval results. This algorithm improves the rate of hash code generation and avoids the computer memory being occupied for a long time. However, in the process of dimensionality reduction, information loss cannot be avoided. Therefore, using low-rank constraint is necessary to recover the loss more accurately. It can help maintain global structure of image features, thus learning binary codes with higher robustness.

Nowadays, more and more researchers pay attention to low-rank constraint [14-16]. Compared with the algorithm based on L_1 norm, it can find the low-rank structure from the high-dimensional data damaged by noise and outliers, and further strengthen the robustness of algorithm through constraints. Low-rank representation (LRR) is mainly applied to the partition of subspaces [17]. It assumes that the original space is composed of low-rank subspaces and sparse errors. That is to say, given a set of data, the linear combination of bases in a given dictionary can be used to represent this set of data, and the coefficient matrix is subject to overall constraints. Therefore, LRR can better represent the overall structure of data. In addition, robust principal component analysis (RPCA) [18] assumes that data errors obeyed Gaussian distribution and performs robust correction by introducing kernel norm. It restores low-rank subspace structure of the data damaged by noise or occlusion efficiently. However, when encountering untrained new samples, RPCA cannot handle them well. So the inductive robust principal component analysis (IRPCA) algorithm [19] was proposed by Bao et al. This algorithm has good generalization ability by introducing low-rank constraint. However, due to the large amount of computation, learning processes of the above algorithms are all slow and manifold structure of the data is not well maintained as well. Therefore, referring to the idea of the joint sparse hashing (JSH) algorithm [20], $L_{2,1}$ norm is applied to ensure the row sparsity of projection matrix, thus improving feature extraction capability of the proposed algorithm.

Sparse representation is also an important research direction in image retrieval. It can make data simpler and save important information in data at the same time. For example, sparse principal component analysis (SPCA) [21] algorithm reduces the dimension of the most important variables in the process of projection, analyzes the principal component of input data, and uses sparse projection to process noise for enhancing its robustness. Sparse embedded hashing (SEH) [22] not only uses sparse representation well in the process of hash learning, but also maintains the similarity difference between the original data. It integrates projection matrix, sparse representation and linear embedding, and uses matrix decomposition techniques to approximate the Euclidian structure of original data, which achieves better performance in both single-mode and cross-mode retrieval tasks. However, the above sparse algorithm does not fully use all the similarities between data and data. Therefore, joint sparse representation and local preserving projection (JSRLPP) algorithm [23] proposed by Zhang et al. integrates graph embedding (GE) technology into its objective function, so as to learn the graph structure that is more suitable for feature extraction. So in the sparse feature extraction process, the algorithm proposed in this paper adopts GE technology to map two adjacent data points into more similar hash codes for better maintaining local geometric relationship of features.

Since researchers have successfully found inherent characteristics in nonlinear manifolds, more and more extended algorithms based on GE technology have been proposed [24-26]. Firstly, He et al. proposed an algorithm based on one-dimensional vectors named local preserving projections (LPP) [27], but it always faces singular problems. Next, the researcher proposed two-dimension local preserving projections (2DLPP) [28], which uses the image matrix to replace the vector for operation, and more effectively performs the feature processing of image data. Another typical algorithm is locally linear embedding (LLE) [29]. This algorithm expands the three-dimensional Swiss Roll onto a two-dimensional plane to maintain the Euclidean distance between samples. The neighbourhood Preserving Embedding (NPE) [30] algorithm is a linear approximation of LLE algorithm. By constructing the nearest neighbor

graph between samples and solving the least squares problem under constraint conditions, the similarity matrix is calculated to maintain internal structure of data. The above methods have significant performance in the CBIR, not only making the data under the manifold smoother, but also further maintaining image features with local geometric relationships.

In recent years, with the development of deep learning technology, hash algorithms based on deep learning have made significant progress in the field of image retrieval [31]. Zhao et al. proposed a Multiscale context deep hashing for remote sensing image retrieval (MSCDH) [32] method. It is a study for large-scale remote sensing image retrieval tasks, aiming to improve retrieval efficiency and accuracy through multi-scale feature fusion and deep hashing techniques. Chen et al. proposed a TransHash: Transformer-based Hamming Hashing for Efficient Image Retrieval [33] method. It is a Transformer-based hash learning framework that is capable of generating discriminative global and local features through dual-stream multi-granularity feature learning. He et al. proposed a HybridHash: Hybrid Convolutional and Self-Attention Deep Hashing for Image Retrieval [34] method. It combines Convolutional Neural Networks (CNNs) and self-attention mechanisms to capture both local features and global dependencies of an image. Che et al. proposed a Deep Class-Guided Hashing for Multi-Label Cross-Modal Retrieval (DCGH) [35] method. It achieves excellent performance in multi-tag cross-modal retrieval tasks by considering intra-category aggregation and inter-category structural relationships. These methods utilize the powerful feature extraction capability of deep neural networks, which can automatically learn high-level feature representations of images to generate efficient hash codes. However, deep learning methods usually require large amounts of training data and computational resources, and may face overfitting and computational inefficiency when dealing with high-dimensional data.

To sum up, it is still a challenging problem to construct the underlying neighborhood structure of data and learn highly robust and unique hash codes for feature extraction and image retrieval due to random noise and information loss. Therefore, on the basis of the SDH algorithm, integrating low-rank constraint, sparse representation and graph embedding technology, this paper proposes a new efficient hash algorithm named low-rank sparse graph hashing (LSGH) for image retrieval. In the projection stage, low-rank constraint is used to enhance the robustness of LSGH algorithm. Moreover, $L_{2,1}$ norm constraint is introduced to carry out sparse representation for removing redundant information between features. Data is projected from high-dimensional original space to low-dimensional Hamming subspace by means of graph embedding, so as to improve the performance of hash codes. The LSGH algorithm exhibits unique advantages in several aspects compared to deep learning-based hashing algorithms. First, the LSGH algorithm recovers information lost due to noise and outliers in high-dimensional data by means of low-rank constraints, which is able to maintain the global consistency of the data more efficiently, thus improving the accuracy and efficiency of retrieval. Secondly, the LSGH algorithm introduces sparse representation, which further enhances the ability of feature extraction and can better maintain the local geometric relationships of the data. In addition, the LSGH algorithm incorporates graph embedding technology, which is able to project the data from the high-dimensional primitive space to the low-dimensional Hamming subspace, and further improves the quality of the hash code by maintaining the local neighborhood structure of the data. To further validate effectiveness of LSGH, we compared

it with eight mainstream hash methods: LSH, SH, SDH, JSH, BITQ, WDDH, MSCDH and DCGH. The experimental results on four public datasets, MNIST, CIFAR-10, Yale, and CIFAR-100, show that LSGH outperforms the other eight hash algorithms in content-based image retrieval tasks. The development route of the LSGH algorithm is shown in Figure 1 below.

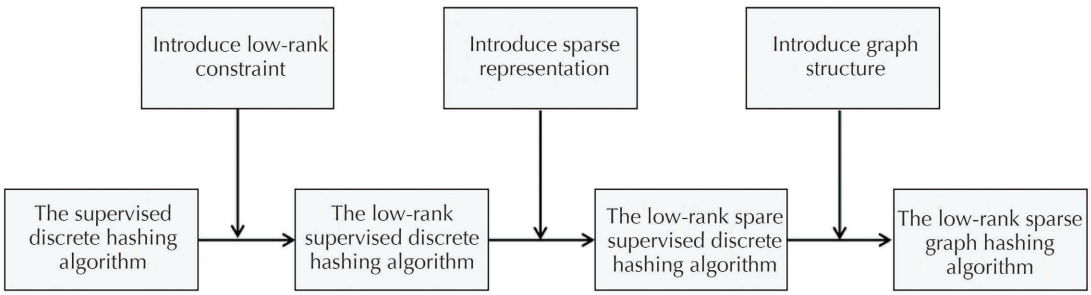


Figure 1. The development route of the LSGH algorithm.

The main innovations of the LSGH algorithm are as below:

1) Low-rank constraint is applied to the projection matrix to better preserve global structure of image features.
2) We utilize $L_{2,1}$ norm constrained projection matrix to denoise, remove redundant information, and capture the local structure of the data.
3) High-order relationships between data are maintained by graph embedding technology, which is suitable for high-dimensional images with complex feature structures.

The rest of this paper is structured as below: Section 2 introduces basic principles of three related algorithms. Then, in Section 3, the derivation and optimization of the LSGH algorithm proposed by us are described in detail. Next, the comparisons of experiment performance are discussed and analyzed in Section 4. Section 5 gives a comprehensive conclusion of LSGH algorithm at last.

2. RELATED ALGORITHMS

2.1 SDH Algorithm

Assuming that dataset $X = \{x_i\}_{i=1}^{n}$ has n samples, SDH aims to learn $H = \{h_i\}_{i=1}^{n} \in \{-1, 1\}^{L \times n}$ as a hash code matrix for maintaining semantic similarity between data, h_i represents the L-bit binary hash codes corresponding to x_i. To obtain the optimal hash codes, SDH uses multi-class classification equation $y = W^T h$ to represent the label vector of sample data, and constrains fitting error of hash codes by integrating regularization idea and nonlinear embedding $P(x)$. Therefore, its objective function is shown in (1).

$$\begin{aligned} &\min_{H,W,F} \left\|Y - W^T H\right\|^2 + \lambda \left\|W\right\|^2 + \gamma \left\|H - P(x)\right\|^2 \\ &s.t. \quad H \in \{-1,\ 1\}^{L \times n} \end{aligned} \tag{1}$$

Here $P(x)$ can be easily obtained by regression calculation, and W can be solved by regularized least squares method. Finally, the hash code matrix H can be learned by using DCC method. γ is a penalty parameter.

2.2 JSH Algorithm

JSH improves its ability to extract data features by enhancing the sparsity of the matrix, thus reducing lost when images are reduced from high dimension to low dimension. By adding $L_{2,1}$ norm constraint to the target function, it realizes feature extraction in the mapping process. The target function of JSH is as follows:

$$\begin{aligned} &\min_{\tilde{B},A,P} \sum_{j=1}^{n_2} \sum_{i=1}^{n} \left\|\tilde{b}_j - AP^T x_i\right\|_2^2 Z_{ij} + \lambda \left\|P\right\|_{2,1} \\ &s.t. \quad \tilde{b} \in \{-1,\ 1\}^{L \times n},\ \mathrm{A}^T\mathrm{A} = I_d \end{aligned} \tag{2}$$

where the projection matrix P is a row sparse matrix after introducing a sparse regularization term. The element Z_{ij} in the anchor graph Z is used to represent the similarity, and $\tilde{b}$ represents the hash code of Z_{ij}. λ is a regularization parameter. By alternating iterations and singular value decomposition, the optimal solutions for all variable quantity can be obtained.

2.3 WDDH Algorithm

The WDDH algorithm assigns a higher weight to adjacent points and reduces the weight of points that are not neighbouring but belong to the same label, so as to optimize the coarse-grained label problems existing in practical applications. Finally, using the regularization method in large-scale optimization, its objective function can be obtained as follows:

$$\begin{aligned} &\max_{B,U,F} \ \mathrm{tr}(U^T H S_w H^T U) - \mathrm{tr}(U^T H S_b H^T U) \\ &- \upsilon \left\|H - F(X)\right\|_F^2 \end{aligned} \tag{3}$$

where H is a hash code matrix. WDDH decomposed similarity measure matrix W into $W = UU^T$. The intra-class neighborhood graph S_w and inter-class neighborhood graph S_b are constructed respectively to preserve geometric structure of image. υ is the penalty parameter. In addition, $F(X)$ is used to simulate fitting error of H.

3. LSGH ALGORITHM

3.1 Problem Formulation and Symbols

SDH uses a series of compact hash codes to represent original images, which greatly improves the retrieval speed and reduces the pressure of computer storage, and is used in the CBIR field widely. However, because of the feature information loss and the interference of noise points in the projection process, its accuracy and robustness are both low. Therefore, we make improvements based on the SDH algorithm and then propose the LSGH algorithm with higher accuracy and robustness for image retrieval. The frame of the LSGH algorithm is shown as follow Figure 2.

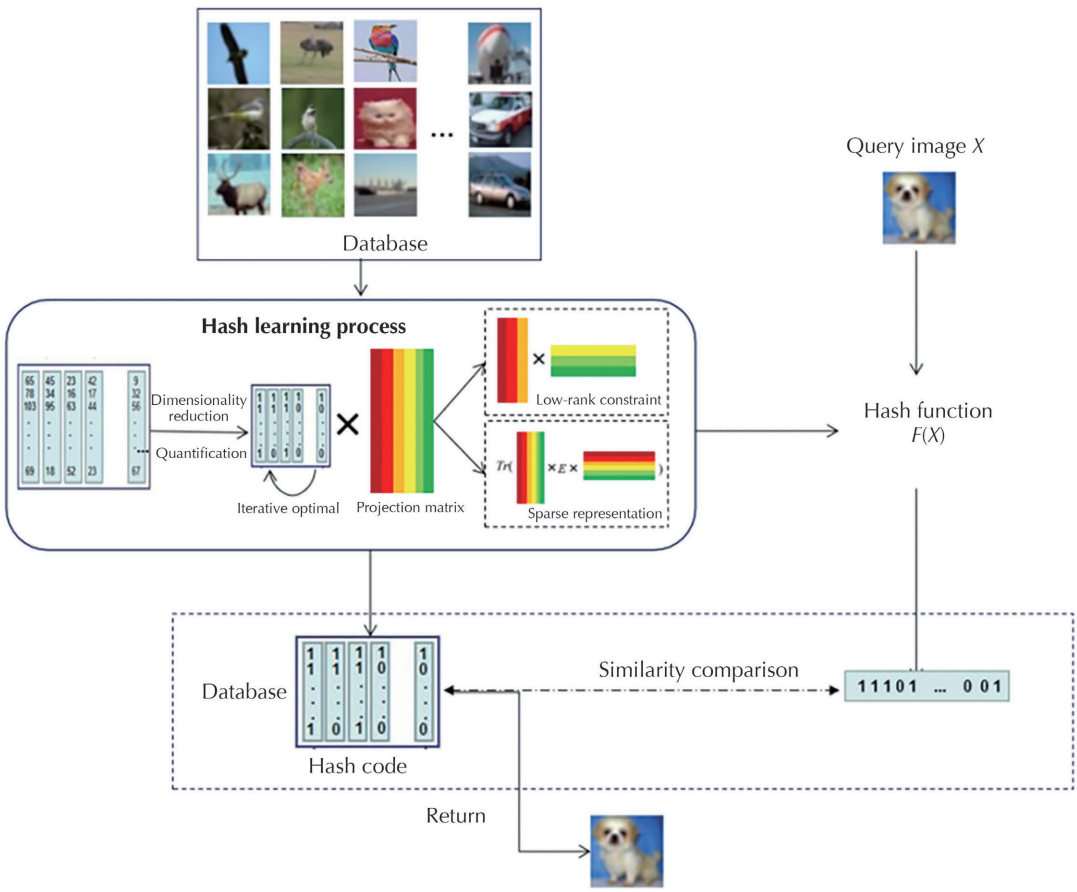


Figure 2. The frame of the LSGH algorithm.

Next, some symbols involved in the LSGH algorithm are explained. Suppose $O = \{o_i\}_{i=1}^{n}$ is a set of images and $X = \{x_1, x_2, \ldots, x_n\} \in \mathbb{R}^{n \times m}$ is its feature, where m is the dimension and n is image total. The

goal of the LSGH algorithm is to learn $H = \{h_1, h_2, \ldots, h_n\} \in \{-1, 1\}^{n \times l}$ as a hash code matrix and $HR(x) = sgn(R(x))$ as a hash function, which is used to maintain the similar structure between images. The matrix element $h_i \in \{-1, 1\}^{n \times l}$ represents the hash code of o_i, where l is used as the length of hash codes. $sgn(\cdot)$ is a sign function, $sgn(u) = \begin{cases} 1, & u \geq 0 \\ -1, & otherwise \end{cases}$.

3.2 The LSGH Algorithm

For joint learning classifiers, hash codes need to be optimal in order to better utilize label information. Therefore, a multi-class classification equation is adopted as follow:

$$y = W^T h \tag{4}$$

Here $y \in \mathbb{R}^{1 \times d}$ is a label vector, whose maximum value represents the specified class of x. d represents class number. And $W = \{w_1, w_2, \ldots, w_d\} \in \mathbb{R}^{l \times d}$ is a projection matrix, where $w_c \in \mathbb{R}^{l \times 1}$ is the projection vector of class $c \in \{1, 2, \ldots, d\}$. Therefore, the following problems need to be optimized:

$$\begin{aligned} &\min_{H, W} \left\| Y - W^T H \right\|_F^2 + \lambda_1 \left\| W \right\|_F^2 \\ &s.t.\ H \in \{-1, 1\}^{n \times l} \end{aligned} \tag{5}$$

where $Y = \{y_1, y_2, \ldots, y_n\} \in \mathbb{R}^{n \times d}$ is a ground truth label matrix, that is, when x_i belongs to class c, $y_{ic} = 1$, otherwise equal to 0. λ_1 is a regularization parameter, and the Frobenius norm is used to constrain the loss, which is convenient to minimize difference between label matrix Y and predicted value $W^T H$.

Low-rank constraint is used for capturing global structure of image features to keep data coefficients in the same space highly correlated. Moreover, it also alleviate impact of noise on data and make its regression more accurate. Therefore, the projection matrix $W = AC$ is decomposed into two low-rank matrices, that is $A \in \mathbb{R}^{l \times k}$, $C \in \mathbb{R}^{k \times d}$, $rank(W) = k \leq \min(l, d)$. Then, (5) can be further transformed into:

$$\begin{aligned} &\min_{H, A, C} \left\| Y - C^T A^T H \right\|_F^2 + \lambda_1 \left\| AC \right\|_{2,1} \\ &s.t.\ H \in \{-1, 1\}^{n \times l},\ AA^T = I \end{aligned} \tag{6}$$

Here I is a k dimensional identity matrix. In addition, to maintain local structure of data and reduce noise and redundant information between features, the $L_{2,1}$ norm is used to conduct row constraint on W. The constraint can be minimized when as many elements in its row as possible are zero. For maintaining the similarity structure of data, the closer two adjacent data are, the more similar their mapped hash codes should be. Therefore, the LSGH algorithm obtains the optimal hash codes by introducing Laplacian matrix:

$$\begin{aligned} &\min_{H, A, C} \left\| Y - C^T A^T H \right\|_F^2 + \lambda_1 \left\| AC \right\|_{2,1} \\ &+ \frac{\lambda_2}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} S_{ij} \left\| h_i - h_j \right\|_F^2 \\ &s.t.\ H \in \{-1, 1\}^{n \times l}, AA^T = I \end{aligned} \tag{7}$$

Equation (7) is equivalent to:

$$\begin{aligned}&\min_{H,\,A,\,C}\left\|Y-C^TA^TH\right\|_F^2\\&+\lambda_1\left\|AC\right\|_{2,\,1}+\lambda_2 tr(H^TLH)\\&s.t.\ H\in\{-1,\ 1\}^{n\times l},\ AA^T=I\end{aligned}\tag{8}$$

where λ_2 is a regularized parameter. L is a Laplace matrix, $L = D - S$. Matrix D is the natural measurement of the original image data. Larger the D_{ii} value of point i is, more important the corresponding x_i is. $D_{ii}=\sum_{j=1}^{n}S_{ij}$. S is the similarity matrix, which is used to record the similarity between data. Its matrix elements S_{ij} measure similarity relationship between point i and point j of class c, expressed as follows:

$$S_{ij}=\begin{cases}\exp\left(-\left\|x_i-x_j\right\|^2/2\sigma^2\right),\ x_i\in \mathrm{N}_c(x_j)\\0\qquad\qquad\qquad\qquad,\ otherwise\end{cases}\tag{9}$$

Here σ represents the neighborhood width of two points, and $x_i \in \mathrm{N}_c(x_j)$ represents that x_i is the near neighbor of x_j. Then the method of joint learning data graph is proposed. This is because the above derivation process assumes that the LSGH algorithm learns hash codes and the hash function on a data graph pre-constructed with label information, and this data graph remains unchanged during the learning process. However, in fact, due to incomplete label information, the data graph may be incompatible with the algorithm model, resulting in the algorithm is insufficient to complete the learning task.

Through joint learning, data graph can be compatible with learning model, further improving the recognition ability of hash codes. S as the graph similarity matrix needs to be consistent with S_0 which is the initial graph similarity matrix to avoid trivial solutions. In addition, the sum of similarity between one image and the others should be limited to 1, and $S_{ij} = 0$. Therefore, the ultimate objective function of the LSGH algorithm is shown in (10).

$$\begin{aligned}&\min_{H,\,A,\,C,\,S}\left\|Y-C^TA^TH\right\|_F^2+\lambda_1\left\|AC\right\|_{2,1}\\&+\ \lambda_2\ tr(H^TLH)+\alpha\left\|S-S_0\right\|_F^2\\&s.t.\ H\in\{-1,\ 1\}^{n\times l},\ AA^T=I,\ \sum_{j=1}^{n}S_{ij}=1,\ i=1,\ 2,\ \ldots,\ n\end{aligned}\tag{10}$$

where α is a positive equilibrium parameter. In the learning process, the Laplace matrix L is further updated to $L = D - (S + S^T)/2$. To sum up, the first term of LSGH objective function uses binary codes to learn classifiers, the second term minimizes the projection matrix constrained by $L_{2,\,1}$ norm to preserve its sparsity. The third term maintains inherent complex structure of data through the Laplace matrix L for obtaining high-quality hash codes. Hash learning, low-rank constraint, sparse representation and graph embedding are successfully integrated into a unified framework.

3.3 Optimization Algorithm

It is difficult to find its global solution due to the highly non convex objective function of LSGH, so we choose to solve each variable in the objective function iteratively one by one.

(1) Update *C*

Fix *A*, *H* and *S*, and update *C* by calculating (11):

$$\begin{aligned}&\min_{C}\left\|Y-C^{T}A^{T}H\right\|_{F}^{2}+\lambda_{1}\left\|AC\right\|_{2,1}\\&=-2tr(H^{T}ACY)+2tr(H^{T}ACC^{T}A^{T}H)\\&+\lambda_{1}\left\|AC\right\|_{2,1}\end{aligned}\tag{11}$$

Take the derivative of *C*, and set its derivative equal to zero, i.e.

$$\frac{\partial}{\partial C}(-2tr(H^{T}ACY)+2tr(H^{T}ACC^{T}A^{T}H)+\lambda_{1}\parallel AC\parallel_{2,1})=0\tag{12}$$

then we get:

$$C=(ACC^{T}A^{T}+\lambda_{1}AEA^{T})^{-1}ACY\tag{13}$$

The expression of diagonal matrix *E* is as follows:

$$E=\frac{1}{2\left\|(AC)^{i}\right\|_{2}}\tag{14}$$

where $(AC)^{i}$ denotes column *i* of *AC*.

(2) Update *A*

LSGH needs to fix *C*, *H* and *S* for updating *A*. But orthogonal constraints make it difficult to obtain the optimal solution of *A*, so the gradient descent and curve search methods are used to find a local optimal solution. First, *G* is defined as the gradient of *A*, whose expression is shown in (15).

$$\begin{aligned}&G=-2CYH^{T}+2CC^{T}A^{T}HH^{T}\\&+2\lambda_{1}EACC^{T}\end{aligned}\tag{15}$$

Next, the Crank Nicolson scheme is used for straight-line search along $J_t(\tau)$ path defined by (16), and the step size τ_t satisfying Wolfe criterion is calculated. *t* represents iterations.

$$J_{t}(\tau)=A-\frac{\tau_{t}}{2}(G^{T}A-A^{T}G)(A+J_{t}(\tau))\tag{16}$$

Referring to the idea of Cayley transformation, we can get the closed solution of $J_t(\tau)$ is:

$$J_t(\tau) = A\left(I - \frac{\tau_t}{2}\left(A^TG - G^TA\right)\right)\left(I + \frac{\tau_t}{2}\left(A^TG - G^TA\right)\right)^{-1} \tag{17}$$

(3) Update *H*

In order to update H, C, A and S need to be fixed, and (18) needs to be solved:

$$\begin{aligned} &\min_H F(H) = \left\|Y - C^TA^TH\right\|_F^2 + \lambda_2 tr(H^TLH) \\ &s.t.\ H \in \{-1,\ 1\}^{n \times l} \end{aligned} \tag{18}$$

Under influence of discrete constraints, (18) has no closed solution, so approximate point gradient method is used to optimize (18). Specifically, in the i^{th} iteration, the surrogate function $\hat{F}_i(H)$ is defined as the discrete approximation of $F(H)$ at point H_i. In this way, for a given H_i, the next discrete point H_{i+1} can be obtained through optimization. However, some elements in the gradient operator $\nabla F(H_i)$ may be zero, which will lead to the possibility of multiple solutions for H_{i+1}. For addressing the above problem, $G(m,\ n) = \begin{cases} m,\ m \neq 0 \\ n\ ,\ n \neq 0 \end{cases}$ as a new function is introduced to eliminate the zero element in $\nabla F(H_i)$, thus obtaining the expression of H_{i+1} as:

$$H_{i+1}\, sgn(G(\nabla F(H_i),\ H_i)) = sgn(G(-2YC^TA^T +\ 2HACC^TA^T + 2\lambda_2 LH,\ H_i)) \tag{19}$$

(4) Update *S*

In order to update S, it is necessary to fix C, A and H solve (20):

$$\begin{aligned} &\min_S \lambda_2 tr(H^TLH) + \alpha \left\|S - S_0\right\|_F^2 \\ &s.t.\ H \in \{-1,\ 1\}^{n \times l}, \sum_{j=1}^{n} S_{ij} = 1,\ i = 1,\ 2,\ \ldots,\ n \end{aligned} \tag{20}$$

Further rewritten (20) as follows:

$$\begin{aligned} &\min_S \sum_{i=1}^{n} \left\|s^i - s_0^i\right\|_2^2 + \frac{\lambda_2}{2\alpha} \sum_{i=1}^{n} \sum_{j=1}^{n} S_{ij} \left\|h^i - h^j\right\|_2^2 \\ &s.t.\ \ \sum_{j=1}^{n} S_{ij} = 1,\ i = 1,\ 2,\ \ldots,\ n,\ s^i \geq 0 \end{aligned} \tag{21}$$

Equation (21) can be divided into n independent sub-problems for solving. For more information, please refer to the solution of constrained quadratic programming problem in Reference [31].

3.4 Hash Function

Assuming there is a linear mapping between original space and Hamming space, what we need to do next is using the optimal hash codes learned after previous iterations to complete the projection of data in these two spaces. We define Q as the transformation matrix and measure the fitting error between data and hash codes by calculating (22).

$$\min_{Q} \|H - XQ\|_F^2 \tag{22}$$

So the hash function is defined as:

$$HR = sgn(xQ) \tag{23}$$

where $Q = (X^T X)^{-1} X^T H$.

Algorithm 1 gives the concrete steps of LSGH:

Algorithm 1. The LSGH Algorithm.

Input:

Feature matrix X, Label matrix Y,

Hash code length l;

Output:

Hash code matrix H and

hash function $HR = sgn(xQ)$.

1. Initialize C, A, H and S;
2. Initialize t, λ_1, λ_2 and α;
3. **Repeat**
4. Update C by calculating (13);
5. Update A by calculating (15);
6. Update H by calculating (19);
7. Update S by calculating (21);
8. Update $t = t + 1$;
9. **Until** convergence

3.5 Computational Complexity

During the optimization of our algorithm, we adopt a strategy of updating the hash function and the hash code synchronously, while introducing the technique of updating the hash code in a single step to reduce the time consumption of the overall optimization process. This approach allows us to update the hash code only once when optimizing the hash function, thus reducing the time complexity of each iteration and improving the operational efficiency of the algorithm. Through simple analysis, we can find that the time complexity of LSGH algorithm is $O(d^2 + d)$.

4. EVALUATION AND ANALYSIS OF EXPERIMENTAL RESULTS

Experimental comparisons were conducted with the mainstream image retrieval algorithms, including LSH [7], SH [8], SDH [10], JSH [20], BITQ [12], WDDH [13], MSCDH [32] and DCGH [35] on four datasets respectively to verify the effectiveness of LSGH. During experiments, the Hamming radius was set as 2, and hash code length was set as 8, 16, 32, 48, 64 and 128. Then the performance of nine algorithms was evaluated by five indexes: precision, recall, MAP value, training time and testing time. To ensure accuracy of the experimental results, the number of iterations was set as 10.

4.1 Introduction and Setting of Experimental Datasets

(1) The MNIST Dataset

The MNIST dataset is a grayscale image dataset of handwritten numbers {0, 1, …, 9}, containing 70,000 samples in total, with a size of 28 × 28. 6000 images were selected randomly from each class as a training set, and the rest 1000 pictures were set as a test set. The rank of projection matrix was set to 5. Partial examples of the MNIST dataset are shown in Figure 3:

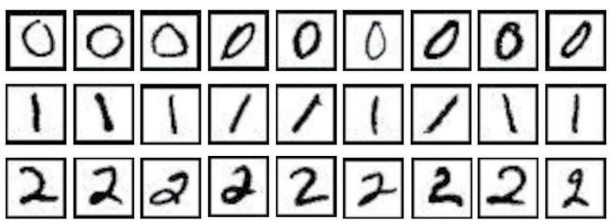


Figure 3. Partial examples of the MNIST dataset.

(2) The CIFAR-10 Dataset

The CIFAR-10 dataset contains 60,000 color images with a pixel size of 32 × 32, including 10 classes. In order to facilitate the experiment, we adjusted each image to a 512-dimensional Gist feature vector, randomly selected 50,000 of them as training samples and 10,000 as test samples, and set the rank of projection matrix to 5. Compared with the MNIST dataset, CIFAR-10 captures real objects in the real

world. The proportion and characteristics of objects are different and there is too much noise. They all bring great difficulties to feature extraction and image retrieval. Partial examples of CIFAR-10 dataset are shown in Figure 4.

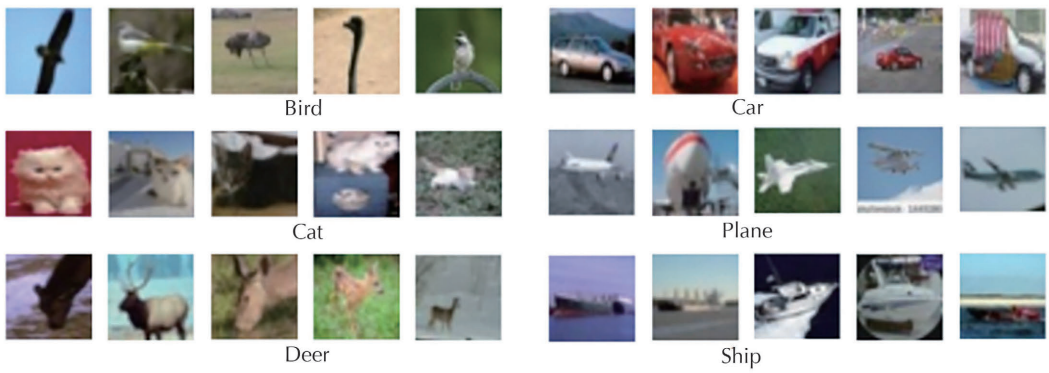


Figure 4. Partial examples of the CIFAR-10 dataset.

(3) The Yale Dataset

The Yale dataset includes 165 grayscale images in total. Fifteen experimental subjects need to adjust their expressions and take 11 pictures when the illumination changes significantly. Therefore, the Yale database can test the feature extraction ability of the LSGH algorithm. At the same time, higher requirements are put forward for the robustness of LSGH too. We divide the Yale dataset into 105 training samples and 60 test samples. The size of each sample is 50 × 50, and the rank of projection matrix is set to 10. Partial examples of Yale dataset are shown in Figure 5.



Figure 5. Partial examples of the Yale dataset.

(4) The CIFAR-100 Dataset

The CIFAR-100 dataset is similar to CIFAR-10 but contains 100 categories with 600 images per category. These images are divided into 500 training images and 100 test images, and set the rank of projection matrix to 5. The 100 categories in CIFAR-100 are further divided into 20 super categories. Each image has a "fine" label (indicating the specific category it belongs to) and a "coarse" label (indicating the super-category it belongs to).

The operations of four different datasets are summarized in Table 1 below.

Table 1. The operation of four different datasets.

Dataset	Class Number	Dimension	Train Set	Test Set	Rank of Projection Matrix
MNIST	10	784	60,000	10,000	5
CIFAR-10	10	512-dimensional Gist vector	50,000	10,000	5
Yale	15	2500	105	60	10
CIFAR-100	100	1024	50,000	10,000	5

4.2 Experimental Results on the MNIST Dataset

Figures 6 and 7 show the precision (%) and recall (%) of seven algorithms on the MNIST dataset respectively while the hash code length is set from 8 to 128. The performance of LSGH algorithm with 32 hash bits is significantly better than the others. Its precision is up to 95.6%, which is nearly 5% higher than the second highest WDDH precision, and is 1.2 times more accurate than SDH precision. The recall of LSGH algorithm is also higher than other algorithms. Especially when the length of hash codes is 8, it is up to 89.1%, which is about 4% higher than WDDH recall and about 10% higher than that of SDH.

From Table 2, we can clearly see that the average MAP values of LSGH with different code bits is about 0.946. In contrast, the average MAP value of WDDH with different hash code lengths is just 0.895. It is about 0.51 less than that of LSGH algorithm. When hash code length is 128, the highest MAP value of SDH is 0.904, which is still 0.55 lower than that of LSGH algorithm. Obviously, the LSGH algorithm performs well in processing high-quality label data. This is because LSGH integrates low-rank constraint, sparse representation, graph embedding and hash learning, which can better retain the manifold structure of image features.

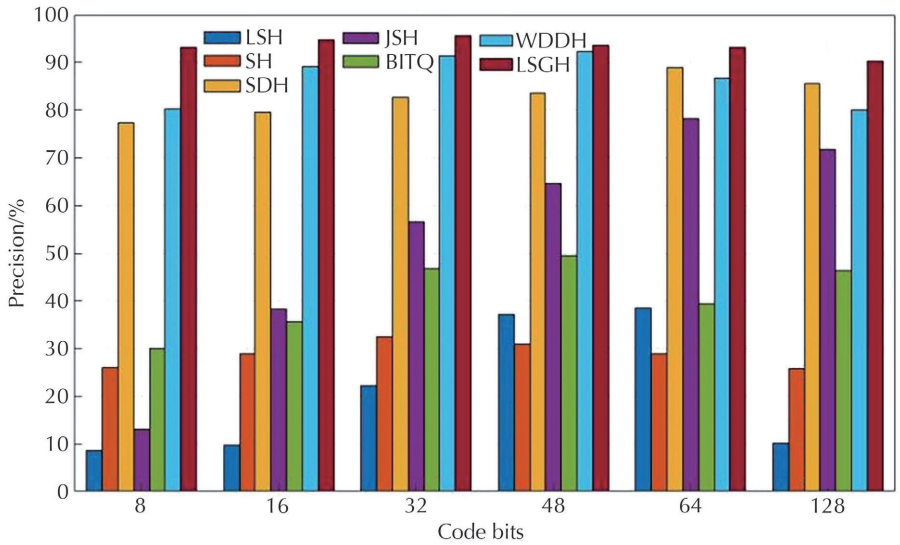


Figure 6. The precision of seven algorithms on the MNIST dataset with different code bits.

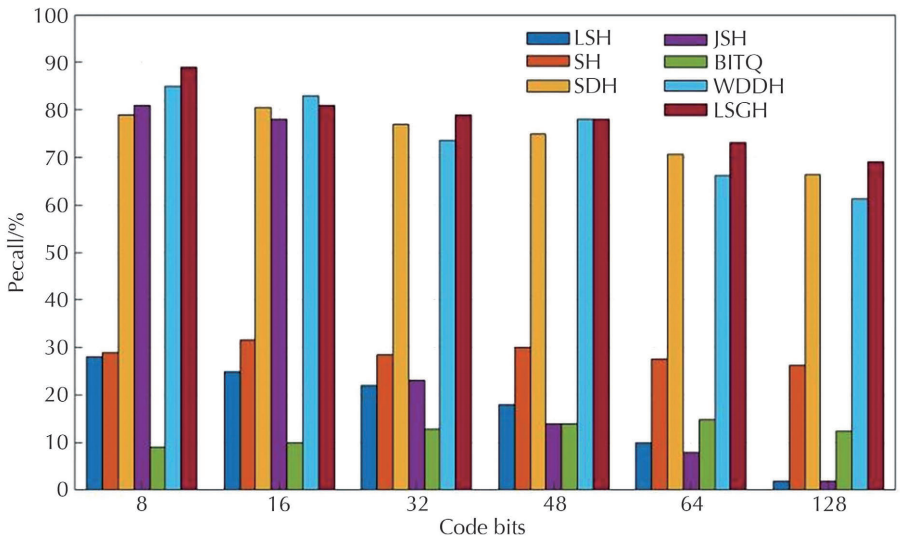


Figure 7. The recall of seven algorithms on the MNIST dataset with different code bits.

4.3 Experimental Results on the CIFAR-10 Dataset

The precision changes of LSGH and other comparison algorithms on the CIFAR-10 dataset are shown in Figure 8. Because images of the CIFAR-10 dataset come from different objects photographed in the real world. Their size and shape are various, and the existence of noise seriously affects image quality, which results that the precision of all algorithms on the CIFAR-10 dataset is far lower than on the MNIST dataset.

Table 2. The MAP values of different algorithms on the MNIST dataset with different code bits.

Code Length (bits)	LSH	SH	SDH	JSH	BITQ	WDDH	LSGH
8	0.204	0.304	0.737	0.333	0.382	0.869	0.912
16	0.227	0.243	0.781	0.449	0.388	0.881	0.938
32	0.260	0.356	0.889	0.489	0.692	0.901	0.953
48	0.321	0.378	0.886	0.537	0.753	0.913	0.955
64	0.342	0.367	0.888	0.549	0.778	0.892	0.958
128	0.381	0.351	0.904	0.580	0.830	0.914	0.959

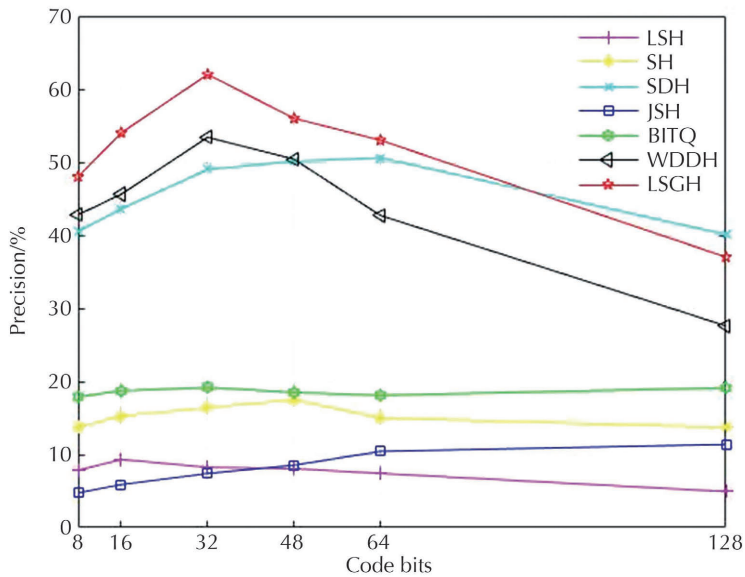


Figure 8. The precision (%) of seven algorithms on the CIFAR-10 dataset with different code bits.

Especially in the projection process, hash algorithms that ignore the neighborhood structure of data points and noise interference have poor performance, and their precision is less than 30%. In general, the precision of LSGH is still the best with different hash code bits, because reducing noise interference is an indispensable step to improve the precision of algorithm. When hash code length is 32, LSGH has the highest precision, reaching 62.3%, which is about 8.6% higher than that of WDDH. The SDH algorithm achieves the maximum precision when hash code length is 64, but it is still 5.4% lower than LSGH precision.

Figure 9 shows the recall comparison results of LSGH and related algorithms. LSGH can not only use the label information, but also reduce the noise of original images. It improves the retrieval speed and learn more robust binary codes. When binary code length is 8, LSGH has the highest recall, about 39.1%,

which is 1.2 times the SDH recall. With the same hash code length, the recall of WDDH is 36.4%, which is about 2.7% lower than that of LSGH. The experimental results show that the LSGH algorithm can obtain the optimal binary codes only after reducing the impact of noise and effectively reduces the error by constraining the projection matrix with low-rank constraint and then using the $L_{2,1}$ norm constraint.

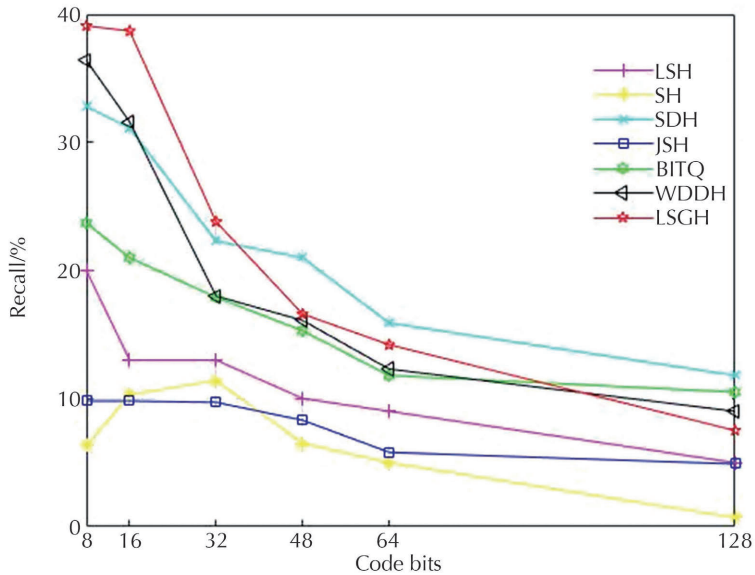


Figure 9. The recall of seven algorithms on the CIFAR-10 dataset with different code bits

The MAP values of seven algorithms with different binary code bits are shown in Table 3. The average MAP values of supervised hash algorithms are significantly higher than that of unsupervised hash algorithms. When hash code length is 128, the MAP value of LSGH is about 0.144 higher than that of WDDH, and is 1.33 times that of SDH. By observing the experimental results, we can conclude that the algorithm performance of LSGH is much higher than that of SDH in the image retrieval task. In summary,

Table 3. The MAP values of different algorithms on the CIFAR-10 dataset with different code bits.

Code Length (bits)	LSH	SH	SDH	JSH	BITQ	WDDH	LSGH
8	0.121	0.135	0.327	0.206	0.157	0.437	0.475
16	0.136	0.129	0.447	0.178	0.148	0.461	0.536
32	0.147	0.134	0.423	0.188	0.149	0.486	0.608
48	0.201	0.123	0.409	0.162	0.252	0.491	0.617
64	0.282	0.127	0.457	0.169	0.293	0.507	0.629
128	0.257	0.149	0.480	0.171	0.249	0.495	0.639

LSGH can effectively preserve the local structure of data points while maintaining the global structure of the image, and has strong robustness to resist noise interference.

4.4 Experimental Results on the Yale Dataset

The experimental performance of algorithm is influenced by external factors, such as the facial expression changes and different lighting conditions of each subject during the shooting process. Therefore, we choose to test whether LSGH is affected by random factors on the Yale dataset to evaluate its accuracy and robustness on small sample data. The accuracy and recall of each algorithm on the Yale dataset are shown in Figures 10 and 11. Because the sample size of Yale is very small, hash code length is set from 8 to 64. Although the accuracy of each experimental algorithm is not very high due to the challenge of Yale datasets themselves, the accuracy of LSGH is still slightly higher than other algorithms.

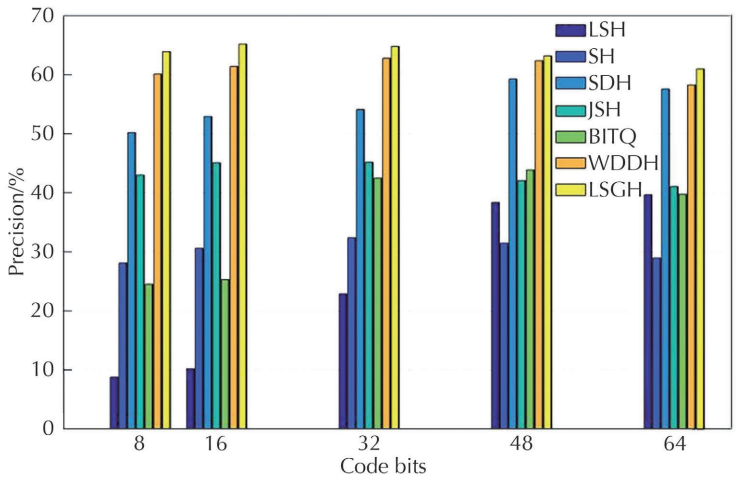


Figure 10. The precision (%) of seven algorithms on the Yale dataset with different code bits.

The precision of LSGH is 65.2% with 16 hash code bits, which is about 3.8% higher than WDDH precision and about 12.3% higher than SDH precision. In addition, the recall of LSGH is 58.2% with 8 hash code bits, which is about 6.7% higher than SDH recall and about 15.1% higher than WDDH recall. It is easily found that when doing experiments on the small dataset, the performance of using long hash codes to represent images are not always optimal, but the overall trends of LSGH precision and recall are both relatively flat, indicating that LSGH algorithm has strong robustness. In addition, Table 4 shows the MAP values of seven comparison algorithms on the Yale dataset. The average MAP of LSGH algorithm is about 0.769, which is significantly better than other algorithms. The average MAP value of SDH algorithm is about 0.66, which is lower than that of LSGH about 0.11.

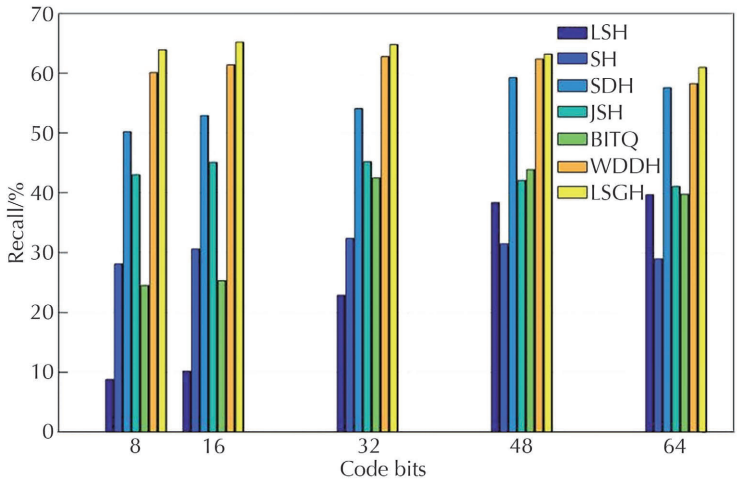


Figure 11. The recall (%) of seven algorithms on the Yale dataset with different code bits.

Table 4. The MAP values of different algorithms on the Yale dataset with different code bits.

Code Length (bits)	LSH	SH	SDH	JSH	BITQ	WDDH	LSGH
8	0.471	0.212	0.622	0.636	0.329	0.652	0.731
16	0.466	0.205	0.630	0.641	0.334	0.669	0.749
32	0.582	0.226	0.694	0.644	0.357	0.741	0.802
48	0.546	0.245	0.678	0.679	0.468	0.739	0.763
64	0.671	0.394	0.685	0.685	0.341	0.749	0.801

4.5 Experimental Results on the CIFAR-100 Dataset

We chose two deep hashing algorithms of recent years (MSCDH and DCGH), two unsupervised hashing algorithms (SDH and WDDH), and the LSGH algorithm, which has a relatively good MAP, to analyze the running time. Table 5 and Table 6 show their training time and test time. The analysis results show that the training time of SDH is relatively stable, ranging from 2.669 to 2.955 seconds from 8 bits to 128 bits. This indicates that the training time of SDH is not sensitive to changes in coding length, probably because of the relatively low complexity of its algorithm. Deep hashing algorithms (MSCDH and DCGH) are suitable for scenarios that require high quality of hash coding, but the training time is longer, especially when the coding length increases. For shorter coding lengths (e.g., 8-bit and 16-bit), the training time of the unsupervised algorithms (SDH and WDDH) is significantly shorter than that of the deep hashing algorithms (MSCDH and DCGH), which indicates that the unsupervised algorithms have an advantage in terms of training efficiency. The training time of LSGH is in-between that of the unsupervised algorithms and the deep hashing algorithms, and it performs well at all coding lengths.

Table 5. Training time on the CIFAR-100 database.

Code Length (bits)	MSCDH	DCGH	SDH	WDDH	LSGH
8	8.143	8.641	2.669	3.841	5.152
16	9.394	9.274	3.125	4.754	6.733
32	10.660	10.331	2.671	4.614	7.185
48	11.211	11.059	2.797	5.697	7.662
64	13.375	12.728	2.856	7.712	8.918
128	17.586	16.881	2.955	9.923	10.337

Table 6. Testing time on the CIFAR-100 database.

Code Length (bits)	MSCDH	DCGH	SDH	WDDH	LSGH
8	1.785	1.124	0.531	0.647	1.108
16	1.799	1.473	0.584	0.746	1.012
32	1.784	1.441	0.548	1.173	1.199
48	1.953	1.942	0.531	1.441	1.436
64	2.891	2.544	0.632	1.744	1.541
128	3.035	2.949	0.599	2.593	1.749

Combined with the testing time, the unsupervised algorithms (SDH and WDDH) also show good efficiency in the testing phase, with shorter testing time and less fluctuation with coding length. The deep hashing algorithms (MSCDH and DCGH), on the other hand, have slightly longer test times, but are still within reasonable limits overall. The LSGH algorithm also maintains a good performance in terms of test time, which is similar to that of the unsupervised algorithms, while approaching the deep hashing algorithms in terms of coding quality.

4.6 Parameter Sensitivity Analysis

There are three important parameters in the objective function of LSGH algorithm, which are λ_1, λ_2 and α respectively. Figures 12, 13, and 14 records the sensitiveness of parameter: λ_1, λ_2, α in terms of MAP value on the CIFAR-10 dataset when hash bits is set from 8 to 128. In the experiments, we set λ_1, λ_2 and α all belong to {0.0001, 0.001, 0.01, 0.1, 1, 10, 100}. If $\lambda_1 \in \{10, 100\}$, the MAP value of LSGH algorithm will slightly decrease. When $\lambda_2 \in \{0.01, 0.1\}$, the LSGH algorithm obtains higher MAP values, and the upward trend changes slowly. When $\alpha \in \{1, 10\}$, the performance of LSGH is good. The results of parameter sensitivity experiment show that λ_1, λ_2, and α allow slight control with a little parameter tuning in practice.

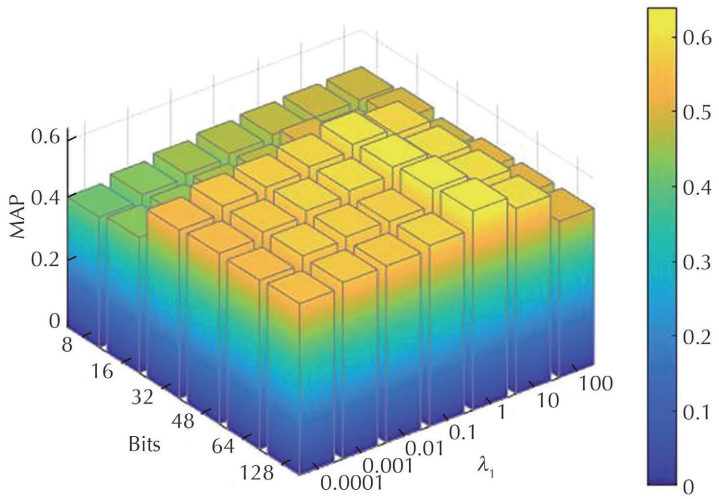


Figure 12. The sensitiveness of parameter λ_1 in terms of MAP value on the CIFAR-10 dataset.

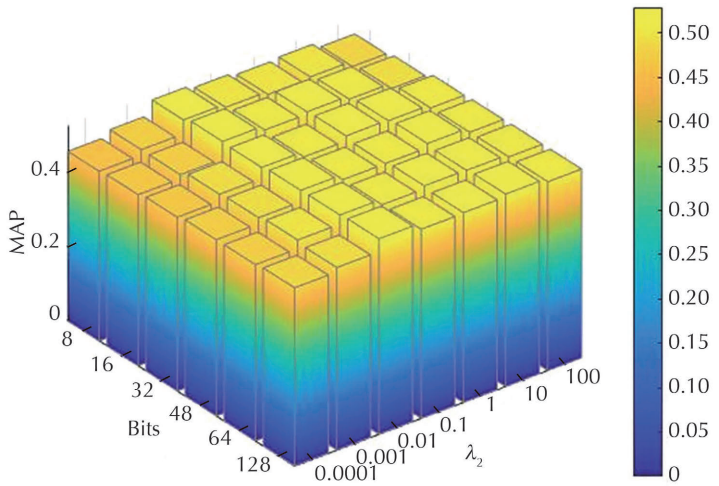


Figure 13. The sensitiveness of parameter λ_2 in terms of MAP value on the CIFAR-10 dataset.

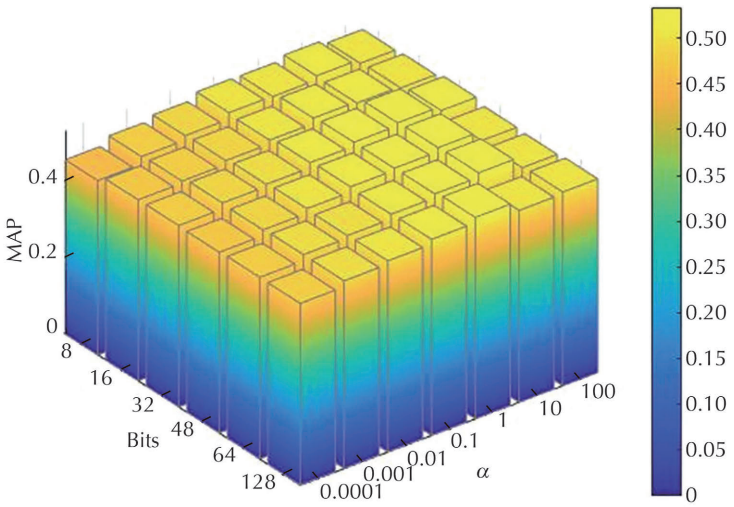


Figure 14. The sensitiveness of parameter α in terms of MAP value on the CIFAR-10 dataset.

5. CONCLUSION

To ensure the performance of hash codes in CBIR field, this paper proposes a new low-rank sparse graph hash (LSGH) algorithm based on the SDH algorithm after more in-depth research. LSGH solves the problem of feature redundancy when projecting data from original space to Hamming space by using a projection matrix constrained by low-rank constraints. The projection matrix adopts the $L_{2,1}$ norm constraint to make itself sparse, which can not only maintain the global structure of original images, but also has a strong anti-noise interference ability at the same time. The LSGH algorithm also focuses on the underlying manifold structure of data by introducing a graph embedding model to make two adjacent points remain close neighbors after projection to minimize the loss of information during the projection process. The experimental comparisons on four public datasets show that the LSGH algorithm can generate highly robust hash codes, improve retrieval efficiency and recognition performance of images. It is more suitable for content-based image retrieval than other mainstream image retrieval hash algorithms.

AUTHOR CONTRIBUTION

Minghua Wan: Conceptualization, Methodology; Ying Zhao: Writing-original draft; Hai Tan: Software; Jun Yin: Formal analysis; Chengli Sun: Formal analysis; Jie Xu: Data curation; Mingwei Tang: Data curation; Guowei Yang: Supervision, Funding acquisition.

ACKNOWLEDGMENT

This work is partially supported by Postgraduate Research & Practice Innovation Program of Jiangsu Province No. SJCX24_1162, the National Science Foundation of China under Grant Nos. 62172229, the Talent Support Program of Jiangxi (No. 20232BCJ22050), Ganzhou City's Open Competition Project: Research on Key Technologies of Customizable Speech Interaction outside Smart Car in Full Scenes, universities' scientific project of Guangzhou Municipal Education Bureau No. 2024312000, the Significant Project of Jiangsu College Philosophy and Social Sciences Research "Research on Knowledge Reasoning of Emergency Plan for Emergency Decision" (No. 2021SJZDA153), the Planning Fund Project of Humanities and Social Sciences Research of Ministry of Education "Research on the Knowledge Organization of Emergency Response Plans for Scenario Deduction" (No. 23YJA870009).

REFERENCES

[1] S. M. Abdullah and M. M. Jaber, "Deep learning for content-based image retrieval in fhe algorithms," *Journal of intelligent systems*, vol. 32, no. 1, p. 20220222, 2023.

[2] H. Rastegar and D. Giveki, "Designing a new deep convolutional neural network for content-based image retrieval with relevance feedback," *Computers and Electrical Engineering*, vol. 106, p.108593, 2023.

[3] K. Wu, D. Xiang, D. Cai, Y. Feng, Y. Xu, and Z. Jiang, "Extraction of incipient fault features of rolling bearings based on cwssmd and 1.5 d-edeo demodulation," *Measurement Science and Technology*, vol. 34, no. 4, p. 045011, 2023.

[4] A. Guan, L. Liu, X. Fu, and L. Liu, "Precision medical image hash retrieval by interpretability and feature fusion," *Computer Methods and Programs in Biomedicine*, vol. 222, p. 106945, 2022.

[5] Z. Zhang and C.-M. Pun, "Learning ordinal constraint binary codes for fast similarity search," *Information Processing & Management*, vol. 59, no. 3, p. 102919, 2022.

[6] X. Chen, M. Wan, H. Zheng, C. Xu, C. Sun, and Z. Fan, "A new bilinear supervised neighborhood discrete discriminant hashing," *Mathematics*, vol. 10, no. 12, p. 2110, 2022.

[7] M. Datar, N. Immorlica, P. Indyk, and V. S. Mirrokni, "Locality-sensitive hashing scheme based on p-stable distributions," in *Proceedings of the twentieth annual symposium on Computational geometry*, 2004, pp. 253262.

[8] Y. Weiss, A. Torralba, and R. Fergus, "Spectral hashing," *Advances in neural information processing systems*, vol. 21, 2008.

[9] M. Raginsky and S. Lazebnik, "Locality-sensitive binary codes from shift-invariant kernels," *Advances in neural information processing systems*, vol. 22, 2009.

[10] F. Shen, C. Shen, W. Liu, and H. Tao Shen, "Supervised discrete hashing," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 37–45.

[11] S. Sirajidin, H. Huo, and T. Fang, "Deep hashing network based on split channels for hybrid-source remote sensing image retrieval," in *Proceedings of the 2020 3rd International Conference on Artificial Intelligence and Pattern Recognition*, 2020, pp. 126–132.

[12] D. Saputra, "A preliminary study of bit-integer hash function for bilinear pairing identity-based cryptosystem," in *Journal of Physics: Conference Series*, vol. 1375, no. 1. IOP Publishing, 2019, p. 012071.

[13] M. Wan, X. Chen, C. Zhao, T. Zhan, and G. Yang, "A new weakly supervised discrete discriminant hashing for robust data representation," *Information Sciences*, vol. 611, pp. 335–348, 2022.

[14] Z. Tian, H. Zhang, Y. Chen, and D. Zhang, "Unsupervised hashing based on the recovery of subspace structures," *Pattern Recognition*, vol. 103, p. 107261, 2020.

[15] Z. Tang, Z. Yu, Z. Li, C. Yu, and X. Zhang, "Robust image hashing with low-rank representation and ring partition," *Wireless Communications and Mobile Computing*, vol. 2020, no. 1, p. 8870467, 2020.

[16] M. Wan, Y. Yao, T. Zhan, and G. Yang, "Supervised low-rank embedded regression (slrer) for robust subspace learning," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 32, no. 4, pp. 1917–1927, 2021.

[17] G. Liu, Z. Lin, and Y. Yu, "Robust subspace segmentation by low-rank representation," in *Proceedings of the 27th international conference on machine learning (ICML-10)*, 2010, pp. 663–670.

[18] E. J. Candès, X. Li, Y. Ma, and J. Wright, "Robust principal component analysis?" *Journal of the ACM (JACM)*, vol. 58, no. 3, pp. 1–37, 2011.

[19] B.-K. Bao, G. Liu, C. Xu, and S. Yan, "Inductive robust principal component analysis," *IEEE transactions on image processing*, vol. 21, no. 8, pp. 3794–3800, 2012.

[20] Z. Lai, Y. Chen, J. Wu, W. K. Wong, and F. Shen, "Jointly sparse hashing for image retrieval," *IEEE transactions on image processing*, vol. 27, no. 12, pp. 6147–6158, 2018.

[21] H. Zou, T. Hastie, and R. Tibshirani, "Sparse principal component analysis," *Journal of computational and graphical statistics*, vol. 15, no. 2, pp. 265–286, 2006.

[22] G. Ding, J. Zhou, Y. Guo, Z. Lin, S. Zhao, and J. Han, "Large-scale image retrieval with sparse embedded hashing," *Neurocomputing*, vol. 257, pp. 24–36, 2017.

[23] H. Liu, Z. Lai, and Y. Chen, "Joint sparse locality preserving projections," in *International Conference on Smart Computing and Communication*. Springer, 2017, pp. 125–133.

[24] Y. Liang, L. You, X. Lu, Z. He, and H. Wang, "Low-rank projection learning via graph embedding," *Neurocomputing*, vol. 348, pp. 97–106, 2019.

[25] P. Qin, W. Chen, M. Zhang, D. Li, and G. Feng, "Cc-gnn: A clustering contrastive learning network for graph semi-supervised learning," *IEEE Access*, vol. 12, pp. 71956–71969, 2024.

[26] J. Gao, J. Xu, X. Guo, Y. Deng, and J. Feng, "Local image descriptor developed from Fechner's law," *Journal of Electronic Imaging*, vol. 32, no. 1, pp. 013037–013037, 2023.

[27] X. He, S. Yan, Y. Hu, P. Niyogi, and H.-J. Zhang, "Face recognition using laplacianfaces," *IEEE transactions on pattern analysis and machine intelligence*, vol. 27, no. 3, pp. 328–340, 2005.

[28] S. Chen, H. Zhao, M. Kong, and B. Luo, "2d-lpp: A two-dimensional extension of locality preserving projections," *neurocomputing*, vol. 70, no. 4–6, pp. 912–921, 2007.

[29] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *science*, vol. 290, no. 5500, pp. 2323–2326, 2000.

[30] X. He, D. Cai, S. Yan, and H.-J. Zhang, "Neighborhood preserving embedding," in *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, vol. 2. IEEE, 2005, pp. 1208–1213.

[31] X. Luo, H. Wang, D. Wu, C. Chen, M. Deng, J. Huang, and X.-S. Hua, "A survey on deep hashing methods," *ACM Transactions on Knowledge Discovery from Data*, vol. 17, no. 1, pp. 1–50, 2023.

[32] D. Zhao, Y. Chen, and S. Xiong, "Multiscale context deep hashing for remote sensing image retrieval," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 16, pp. 7163–7172, 2023.

[33] Y. Chen, S. Zhang, F. Liu, Z. Chang, M. Ye, and Z. Qi, "Transhash: Transformer-based hamming hashing for efficient image retrieval," in *Proceedings of the 2022 international conference on multimedia retrieval*, 2022, pp. 127–136.

[34] C. He and H. Wei, "Hybridhash: Hybrid convolutional and self-attention deep hashing for image retrieval," in *Proceedings of the 2024 international conference on multimedia retrieval*, 2024, pp. 824–832.

[35] H. Chen, Z. Zou, Y. Liu, and X. Zhu, "Deep class-guided hashing for multi-label cross-modal retrieval," *Applied Sciences*, vol. 15, no. 6, p. 3068, 2025.

AUTHOR BIOGRAPHY



Minghua Wan received the B.S. degree in automated institute from the Nanchang University of Aviation in 2003, the M.S. and PhD degrees in pattern recognition and intelligent systems from the Nanjing University of Science and Technology (NUST) in 2007 and 2011, respectively. He is currently a professor at the Shanghai Maritime University, Shanghai, China. He is the author of more than 100 scientific papers in pattern recognition and computer vision. His research interests include face recognition and detection, and image processing.



Ying Zhao is currently pursuing the master's degree with School of Computer Science (School of Intelligent Audit), Nanjing Audit University, Nanjing, China. She current research interests include machine learning, hashing learning, multi-view learning and incomplete multi-view feature selection.



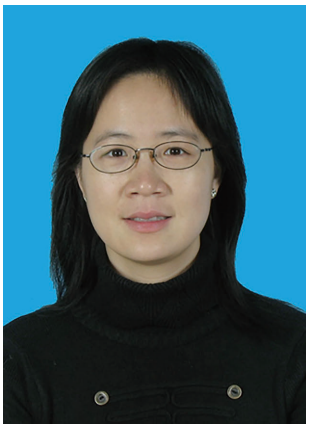
Hai Tan obtained his B.S. and M.S. degrees from Taiyuan University of Technology, Taiyuan, China in 2000 and 2003 respectively, and his Ph.D. degree from Beijing Institute of Technology, China in 2016. At present, he serves as a professor at the School of Computer Science (School of Intelligent Audit), Nanjing Audit University, Nanjing, China. His current research interests span pattern recognition, in-memory computing for large language models, and many-core architecture.



Jun Yin received the B.S. degree in Mathematics and the Ph.D. degree in Pattern Recognition and Intelligence System from Nanjing University of Science and Technology, Nanjing, China in 2006 and 2011 respectively. He is an associate professor at the College of Information Engineering, Shanghai Maritime University, and a postdoctoral fellow at the School of Computer Science and Technology, East China Normal University. From 2019 to 2020, he was a visiting scholar with the Department of Electrical and Computer Engineering, University of Pittsburgh, Pittsburgh, USA. His research interests include multi-view learning, dimension reduction, manifold learning, etc.



Chengli Sun received the B.S. degree in electronics engineering from Zhongbei University, in 1999, and the Ph.D. degree in signal and information processing from Beijing University of Posts and Telecommunication. He is currently a professor with Information Technology School, Nanchang Hangkong University, China. He current research interests include biomedical image analysis, image processing, speech separation and speech recognition.



Jie Xu received the B.S. degree in Mathematics from South China Normal University, the M.S. degree from SUN YAT-SEN University and the Ph.D. degree from Nanjing University of Science and Technology (NUST) China, in 2002, 2008 and 2013, respectively. She is currently a associate Professor of Guangzhou Maritime University, Guangzhou, China. She research interests include image processing and content-based image retrieval, pattern recognition.



Mingwei Tang is an associate professor at Nanjing Audit University. His research interests focus on application of deep learning, large language model, IoT and et al. He has published about 30 papers which are included in SCI, SSCI, EI and CSSCI.



Guowei Yang received the bachelor's and M.S. degrees in mathematics from Jiangxi Normal University, and the Ph.D. degree from the University of Science and Technology Beijing in 2004. He served as a Professor with the Normal College, Qingdao University; a Distinguished Professor with the School of Automation Engineering, Qingdao University; a Post-Doctoral Fellow with the Institute of Semiconductors, Chinese Academy of Sciences; and a Senior Visiting Scholar at the University of Illinois. He is currently a Junior Professor with Qingdao University. The main research regions are artificial intelligence, pattern recognition, big data analysis, intelligent information processing, and intelligent audit.

APPENDIX

A. The Important Symbols

Table VII summarizes all the important symbols used in the LSGH algorithm for facilitate reading.

TABLE VII. The important symbols used in the LSGH algorithm.

Notations	Size	Explanation
$O = \{o_i\}_{i=1}^{n}$	n	Data set
$X=\{x_1, x_2, \ldots, x_n\}$	$n \times m$	Feature matrix
m	–	Feature matrix dimension
n	–	Number of images
H	$n \times l$	Hash code matrix
l	–	Hash code length
h_i	$1 \times l$	Hash code
$HR(x)$	–	Hash function
$sgn(\cdot)$	–	Sign function
y	$1 \times d$	Label vector
d	–	Class number
W	$l \times d$	Projection matrix
w_c	$l \times 1$	Projection vector of class c
λ_1	–	Regularization parameter
$Y = \{y_1, y_2, \ldots, y_n\}$	$n \times d$	Ground truth label matrix
A	$l \times k$	Low-rank matrix
C	$k \times d$	Low-rank matrix
I	$k \times k$	Identity matrix.
λ_2	–	Regularization parameter
L	$n \times n$	Laplace matrix
D	$n \times n$	Natural measurement matrix
S	$n \times n$	Similarity matrix
σ	–	Neighborhood width of the data point
α	–	Positive equilibrium parameter
t	–	Iterations
i	n	The i^{th} sample in the dataset (real space)