An Image Retrieving Method Based on Strong Constrained Manifold-embedded Hashing

Hui Liang¹, Dongbing Pu¹*, Libiao Zhang¹ and Yingjuan Sun²

¹School of information Science and technology, Northeast Normal University, Changchun, China
²College of Computer Science and Technology, Changchun Normal University, Changchun, China

*Corresponding author email: pudb@nenu.edu.cn

Abstract. In recent years hashing methods have attracted considerable attention. The main procedure of manifold hashing methods is to convert the high dimension features into binary codes and then use manifold distance to approximate the original Euclidean distance. However, there still is a problem that how to directly preserve the manifold structure by hashing. Besides, because of information loss, the effect of using existing hashing methods to retrieve images is not ideal when the length of binary code is very short. In order to solve these problems, we proposed a strong constrained manifold-embedded hash (SCDMH), which can not only explore the nonlinear manifold structure of data, but also make full use of the semantic information in the data. Additionally, the objective function we proposed reconstructs the original features of the datasets and the hash representation mutually to preserve the semantic information. Our experimental results on three large benchmark datasets demonstrate that SCDMH outperforms other state-of-the-art baselines.

Keywords: Hashing; Manifold; Image retrieve; Reconstruction.

1. Introduction

In this big data era, data is growing exponentially. If people want to get the information they need, they have to store and process massive amounts of data. Compared with traditional methods of data processing that require a lot of time and space, hashing is a viable technique for representing data in the method of binary codes because of smaller storage space and efficient computation. Hashing has attracted considerable attention of many experts and scholars [1] [2] [3]. About image retrieval based on hashing, the image needs to be transformed into binary codes to find the nearest neighbour of image binary code to be retrieved.

However, how to reduce the semantic loss in the process of representing data as binary codes, maintain the structure of source data, and furthermore improve the effectiveness and potency of hashing methods continues to be a very important and difficult drawback today. Now available hashing methods can be categorized into data-dependent hashing and data-independent hashing. And for data-dependent hashing methods, there are two general ways for hashing, supervised hashing and unsupervised hashing. Supervised hashing is more effective than unsupervised hashing due to the using of labels, and also takes more time due to complex optimization algorithms. The supervised hashing uses the labels of original data to construct the hash optimization framework. The representative unsupervised hashing methods include Supervised discrete hash based on column sampling (COSDISH) [21], Supervised hashing with kernels (KSH) [11], Supervised discrete hashing
(SDH) [12], improved SDH called fast scalable supervised hashing (FSSH) [13] and strongly constrained discrete hashing (SCDH) [14]. There are also some deep learning methods. Such as discrete proximal linearization minimization (DPLM) [29] and deep supervised hashing with dynamic weighting scheme (DHDW) [30]. These in-depth approaches can be well executed, but in-depth learning requires training a lot of data to get hash functions and expensive computing resources.

In this article, our target is learning the fast shallow model of hashing, which is inexpensive to operate and useful for most applications in the world. It is still a hard problem that how to lessen the cost of hashing algorithm based on deep learning, we will study it later.

Furthermore, recent researches have incontestable that it's helpful for an image retrieval model to take advantage of the nonlinear manifold structure of datasets [22] [23]. Therefore, several ways utilize the manifold learning to come up with the binary hash codes. The classic approach is as follows, Spectral Hashing(SH) [5], Reversed Spectral Hashing (RSH) [6], Locally Linear Hashing(LLH) [9] and Discrete Locally Linear Embedding Hashing(DLLEH) [25]. However, these hashing methods are all unsupervised, and in the setting of most hashing methods, the schemes are indirect. These methods can be divided into two steps. Firstly, it builds the manifold structure in the original feature space. Secondly, it maps manifold structure to binary codes. The development of manifold structure and the learning of hash functions are separated, which can cause indirect optimization between both steps.

Not limited to the above hashing methods, some other hashing methods [15] [16] [17] [18] [19] [20] also face two problems: 1) due to considerable dimension reduction, representing a sample using a short length hash code leads to severe information loss. 2) In order to capture the manifold structures under cover, some hashing methods reconstruct the locally linear structures of manifolds in the hamming space. However, the direct learning of hash codes is an NP hard problem and always leads to information loss.

In order to work out these problems and make the utmost of supervised hashing, we proposed a method called “Strongly Constrained Discrete Manifold-Embedded Hashing (SCDMH)” combining double supervision, bit decorrelation, balance constraint, manifold reconstruction and discrete optimization. Its main characteristics are summarized as follows:

- SCDMH directly learns and preserves manifold structure in the hamming space. It directly reconstructs binary codes by the data points that are similar in the original feature space.
- In the proposed method, to achieve better performance, mutual regression and matrix factorization are seamlessly combined for hash learning to reduce the semantic loss.
- Extensive experiments are conducted over three public benchmark datasets. Our proposed methods are greatly better than other state-of-the-art hashing methods with higher retrieval accuracy.

2. Proposed Method

Here we describe the supervised hashing method SCDMH detailedly, including equation definition and construction.

2.1. Notations

Given N training instances $\mathbf{x} = [x_1, x_2, \ldots, x_N]^T$. $\varphi(x) = \exp\left(-\frac{\|x-a_i\|^2}{2\sigma^2}\right)$, where $\sigma$ is the Gaussian kernel parameter and $a_i$ is the anchor. This process can be calculated ahead. All of the $X$ that we mentioned in this paper are $\varphi(x)$. Let $l_i \in \{0, 1\}^C$ be the label of $x_i$. $c \in \{1, 2, \ldots, C\}$ denotes the number of classes in the dataset of images. $B \in \{-1, +1\}^{N \times K}$ is the to-be-learnt hash code matrix for the training data, $K$ is the length of the hash codes.

2.2. Mutual Reconstruction

Let $X = [x_1, x_2, \ldots, x_N]^T$. We to map the $X$ for binary code $B$. According to the linear hash function
\( P \in \mathbb{R}^{M \times K} \). It can be expressed as:

\[
\min_{B, P} \| \text{sgn}(XP) - B \|^2
\]

where \( \text{sgn}(\cdot) \) is a sign function.

The \( \text{sgn}(\cdot) \) function is not differentiable, so the above problem is difficult to solve, and we generally use \( XP \) instead of \( \text{sgn}(XP) \). According to the matrix decomposition theory, the potential semantic features of source data sets can be learned by matrix decomposition. \( B \) is the hash matrix represents the semantic information of the samples, it can be considered as the latent semantic feature of the data. However, existing hash methods generally use only one reconstruction method. In this article, the proposed method can preserve more semantic information by mutual reconstruction of the source dataset and the binary codes. As shown below:

\[
\min_{B, P} \|XP - B\|^2 + \|BU - X\|^2
\]

\( U \in \mathbb{R}^{K \times M} \) is the projection from the binary codes matrix to the original dataset.

### 2.3. Manifold Learning

The LLE mentioned in section two is a nonlinear dimensional reduction algorithm, it is able to maintain the original manifold structure of the data after dimension reduction. LLE keeps the difference of local manifold structure by looking for a low-dimensional feature space with real value, while our goal is to map the original feature space to discrete Hamming space. We want samples of similar locally popular structures to be projected into the similar hash codes and vice versa. Thus we define the KNN affinity matrix \( F \), and then find the optimal solution the same as LLE:

\[
F_{ij} = \sum_{k} G_{ik}^{-1} \frac{1}{\sum_{m} G_{lm}^{-1}}
\]

Where \( G \) is the local Gram matrix. The minimum reconstruction error can be expressed as:

\[
\min_{b} \left\| B - FB \right\|^2
\]

### 2.4. The Complete Optimization Problem

Through all the above function, the final objective function can be expressed as:

\[
\min_{b, p} \left\| K \cdot S - BB^T \right\|^2_F + \lambda \left\| XP - B \right\|^2_F + \alpha \left\| BU - X \right\|^2_F + \beta \left\| B - FZ \right\|^2_F + \gamma \left( \|P\|^2_F + \|U\|^2_F \right)
\]

s.t. \( b \in \{-1, +1\}^{N \times K}, B^T 1_N = 0_K, B^T B = N \cdot I_K \)

Which \( \gamma \left( \|P\|^2_F + \|U\|^2_F \right) \) is the regularization item, parameter \( \gamma \) is a regularization parameter, to prevent the fitting or irreversibility and improve the stability of regression [26]. This is a discrete optimization problem. To make the solution easier, an auxiliary variable \( Z \) is introduced and let \( Z \) be equal to \( B \). To solve this NP hard problem, we further remove the constraint that \( Z = B \), \( B \) and \( Z \) are no longer needed to be strictly identical. We relax \( Z \) to a real valued continuous variable and \( Z \) is approximate the discrete variable \( B \). By combining the equations above, the final objective function of our SCDMH is formulated as:

\[
\min_{b, p, z} \left\| K \cdot S - BZ^T \right\|^2_F + \lambda \left\| XP - B \right\|^2_F + \alpha \left\| BU - X \right\|^2_F + \beta \left\| B - FZ \right\|^2_F + \gamma \left( \|P\|^2_F + \|U\|^2_F \right)
\]
By solving the optimization problem, we can finally get the hash function \( P \) and hash codes \( B \) simultaneously. The same can be done for out of sample data such as test set images and query images. Suppose \( X_{oos} \) represents the images out of sample, we can map it to binary codes through the hash function \( P \) calculated from the data in sample:

\[
B_{oos} = \text{sgn}(X_{oos}P)
\]  

(7)

3. Optimization

In this part, we will show the solution of the SC DMH. Since equation (6) is non-convex and involves discrete constraint, which results in an NP-hard problem, it is difficult to optimize it directly. In the objective function, there are four variables \( U, P, B \) and \( Z \) that need to be solved. We try to use an iterative framework to obtain local minima. The steps are as follows:

3.1. Update \( P \) with \( U, B \) and \( Z \) Fixed

Projection \( P \) can be obtained with variables \( U, B \) and \( Z \) fixed, we can get. The problem in equation (6) becomes:

\[
\min_{P} O(P) = \lambda \|XP - B\|_F^2 + \gamma \|P\|_F^2 \tag{8}
\]

The closed-form solution of \( P \) is

\[
P = \left(X^TX + \frac{\gamma}{\lambda} I_m\right)^{-1}X^T B
\]  

(9)

3.2. Update \( U \) with \( P, B \) and \( Z \) Fixed

The resolution process is similar to \( P \), and you get the result

\[
U = \left(B^TB + \frac{\alpha}{\lambda} I_m\right)^{-1}B^TX
\]  

(10)

3.3. Update \( B \) with \( P, U \) and \( Z \) Fixed

The orthogonal real-valued representation \( B \) can be obtained with variables \( P, U \) and \( Z \) fixed. The problem in equation (6) becomes:

\[
\begin{align*}
\min_{B} O(B) = & \|K \cdot S - BZ^T\|_F^2 + \lambda \|XP - B\|_F^2 + \alpha \|BU - X\|_F^2 + \beta \|B - FZ\|_F^2 \\
\text{s.t.} & B \in \{-1,+1\}^{N \times K}, BB^T = I
\end{align*}
\]  

(11)

Derivative with respect to \( B \), the above function is equivalent to:

\[
\max_{B} \text{Tr} \left( B^T \left( K \cdot SZ + \lambda XP + \alpha XU^T + \beta FZ \right) \right)
\]  

(12)

The equation (12) can be solved by Theorem 1.

Theorem 1. Given a matrix \( C \in R^{N \times K} \), the optimization problem \( \max_{B} \text{tr} \left( BC^T \right) \) has the closed-form solution \( B = \text{sgn} \left( C \right) \).

Therefore, we can get the result of \( B \):
\[ B = \text{sgn}\left(K \cdot SZ + \lambda XP + \alpha XW^T + \beta FZ\right) \]  

(13)

3.4. Update Z with P, U and B Fixed

Learn the projection Z, with the other variables fixed. The objective function of Z can be written as:

\[
\min_Z \ O(Z) = \|K \cdot S - B Z^T\|_F + \beta \|B - F Z\|_F
\]

(14)

\[
s.t. \ Z \in \mathbb{R}^{N \times K}, \ Z^T 1_N = 0_K, \ Z^T Z = N \cdot I_K
\]

The equation (14) can be written as:

\[
\max_Z \ \text{tr}\left( Z^T \left( K \cdot SB + \beta F^T B \right) \right) = \max_Z \ \text{tr}\left( Z^T E \right)
\]

(15)

Let \( E = K \cdot SB + \beta FB \), and then we can get the result of equation (15) through the Theorem 2.

Theorem 2. The optimization problem

\[
\max_Z \ \text{tr}\left( Z^T E \right)
\]

has the closed-form solution:

\[
Z = \sqrt{N} \left[ U, U \right] \left[ V, V \right]^T
\]

(17)

Let \( J = I_N - \frac{1}{N} 1_N 1_N^T \), The matrices \( U = [u_1, u_2, \ldots, u_K] \) and \( V = [v_1, v_2, \ldots, v_K] \) can be obtained by Singular Value Decomposition (SVD) of \( JE \).

\[
JE = U \sum V^T = \sum_{k=1}^{K} \sigma_k v_k v_k^T
\]

(18)

Then, the matrices \( \bar{U} \in \mathbb{R}^{N \times (K-K')} \) and \( \bar{V} \in \mathbb{R}^{N \times (K-K')} \) are obtained via the Gram-Schmidt process such that \( \bar{U}^T \bar{U} = I_{K-K'}, \bar{U}^T 1_N = 0, \bar{V}^T \bar{V} = I_{K-K'}, \bar{V}^T 1_N = 0 \).

3.5. Computational Complexity

SCDMH algorithm is based on the above four optimization sub-problems. In each iteration of the algorithm, the four sub problems will produce the corresponding closed form solution.

For P sub problem, the most time-consuming part of the solution is the calculation of \( X^T X \) and its inverse matrix. Assuming that the number of samples in the data set is \( N \) and the dimension is \( M \), the time complexity is \( O\left(NM^2\right) \) and \( O(M^3) \). The total time complexity of the closed solution for the P sub problem is \( O\left(NM^2 + M^3 + NM^2 + KM^2\right) \), and \( K \) is the length of the resulting hash code. In general, \( M \) and \( K \) are much less than \( N \), so the time complexity of this last sub problem is actually linear. The same is true for U sub problem.

For the B sub problem, the longest time running part is the calculation of \( SZ \) and \( FZ \), the time complexity is \( O\left(KN^2\right) \), the total time complexity is \( O\left(KN^2 + KN\right) \).

For the Z sub problem, the most important step is the SVD decomposition operation of \( JE \), whose time complexity is \( O\left(NK^2\right) \). The time complexity of other operations in this sub problem is so low
that it can be ignored. In general, the time complexity of the Z sub problem is also linear with respect
to N.

In general, only one of the sub problems has a time complexity that is not linear. Therefore, the
proposed SCDMH algorithm is relatively efficient.

4. Experiments

Several large-scale image datasets and evaluation indicators are used to evaluate SCDMH’s retrieval
performance with 128GB RAM PC.

4.1. Datasets

In the experiment, three widely using datasets are used, including Cifar10, MNIST handwritten dataset
and Caltech256 dataset.

Cifar10 contains 60,000 color images. The size of the image is 32*32. There are altogether 10 types
of single-label images, and we extract 512-dimensional GIST features from them. In 60,000 images,
training set includes 54,000 images that are randomly selected. The other 6,000 images are the test set.

Mnist contains 70,000 handwritten digital images from 0 to 9. They all can be represented by a vector
of 784 dimensions. In this dataset, training set includes 69,000 images that are randomly selected. The
other 1,000 images are the test set.

Caltech256 is a dataset collected by the California Institute of Technology. The dataset is selected
from the Google Image data set and manually removes images that do not fit its category. In this data
set, the images are divided into 256 categories, with more than 80 images in each category. Each
image transformed to a 1,024-dimension CNN feature vector. Training set includes 2,9500 images that
are randomly selected. The other 280 images are the test set.

4.2. Evaluation

When hashing learning was finished, We get P and B. And then we need to find the images similar to
the query image. In this article, our main indicator is hamming distance. The smaller the hamming
distance is, the more similar the picture is. Our goal is to find the picture that has smallest hamming
distance. The relevant results refer that at least one label of images are the same as the query image.
In this article, we used mean average precision (mAP) [27], precision rate with the number of retrieved
sample and precision-recall curves to evaluate the image retrieval performance.

4.3. Parameter Sensitivity Analysis

We set the parameters of Algorithm 1 advanced. We set maxIter = 10 and $\varepsilon = 10^{-10}$. For the hyper
parameters $\alpha$, $\beta$, $\gamma$ and $\lambda$, we run the test with changing each variable from $10^{-9}$ to $10^9$. The
combination of these parameters is $(\lambda=10, \alpha=10, \beta=10, \gamma=0.001)$ after several experiments. SCDMH
would employ the Gaussian (RBF) kernel with $\sigma = 0.4$ and anchors=2,000. Keeping all the other
parameters fixed, we vary the number of anchors from 50 to 8000 and plot the retrieval performance
of SCDMH in Figure 1. We can see that the performance will become better and better as the number
of anchors increasing. This is reasonable because it will represent more information about dataset with the anchors increasing. However, the time assuming is becoming longer with the numbers of anchor increasing. So Q is set to 2,000 in our experiments.

We vary the kernel bandwidth $\sigma$ from 0.1 to 100 with keeping all the other parameters fixed. And then we plot the retrieval performance of SCDMH in Figure 2. It can be seen that SCDMH performs well on all the datasets when $\sigma$ is vary from 0.3 to 1.0. So we set $\sigma$ as 0.4 in our experiments.

![Figure 1. The mAP scores of SCDH (Gaussian kernel) w.r.t. the number of anchors Q.](image1.png)

Figure 1. The mAP scores of SCDH (Gaussian kernel) w.r.t. the number of anchors Q.

![Figure 2. The mAP scores of SCDH (Gaussian kernel) w.r.t. the kernel bandwidth $\sigma$.](image2.png)

Figure 2. The mAP scores of SCDH (Gaussian kernel) w.r.t. the kernel bandwidth $\sigma$.

![Figure 3. Convergence curves of the proposed SCDMH for hash codes for three datasets. In each curve, the loss of the proposed SCDMH for the first iteration is considered to be 1.](image3.png)

Figure 3. Convergence curves of the proposed SCDMH for hash codes for three datasets. In each curve, the loss of the proposed SCDMH for the first iteration is considered to be 1.

4.4. Convergence Analysis

We can see clearly from Figure 3 for SCDMH that the result of the objective function is decreasing with the number of iteration increases until it is stabilized. Since during the execution of the algorithm, the objective function is a monotone decreasing function and greater than 0. Theoretically speaking, the algorithm for SCDMH is converging. Figure 3 depicts the changes in the objective values achieved by the SCDMH for three datasets, where normalized objective function values represent CIFAR-10, MNIST and CALTECH256, respectively. The first image and the second image indicate hash codes with lengths of 32 and 64, respectively. As the number of iterations increased, the objective values become small and stable, indicating that the SCDMH appeared to reach convergence rapidly during training, thereby considerably reducing the time required for training.

4.5. Experimental Results and Analysis

We compared the proposed SCDMH with the following methods: locality-sensitive hashing (LSH) [4], principle component analysis hashing (PCAH) [28], principle component analysis (PCA)-iterative quantization (PCA-ITQ) [7], Discrete graph hashing (DGH) [8], Scalable Graph Hashing (SGH) [10], supervised discrete hashing (SDH) [12], column sampling based discrete supervised hashing (COSDISH) [21], and Fast Scalable Supervised Hashing (FSSH) [13], and Strongly Constrained
Discrete Hashing (SCDH) [14]. All parameters are set according to the original article. We set the number of anchors 2000 in all methods. LSH, PCAH, PCA-ITQ, DGH and SGH are unsupervised hashing methods, whereas SDH, COSDISH, FSDH, and SCDH are supervised ones. All the comparison algorithms are non-deep because the proposed method is a method based on linear model. The best results for mAP are shown in bold, the second-best are underlined.

The table 1 shows the mAP values for each method, for CIFAR-10. We compare the mAP from the perspective of the short hash codes and the long hash codes. We approximately set the short-length not greater than 10. According to the mAP of table 1, The mAP performance of the SCDMH was considerably better than those of the other methods for these two benchmark datasets, with short-length and long-length hash codes. Specifically, the SCDMH exhibited more than 10% improvement with an extremely short-length hash code (4-bit in CIFAR-10). In addition, the proposed SCDMH demonstrated considerable improvement with long length hash codes.

Table 1. Performance in terms of mAP score on Cifar10.

| Methods/Length | 4 bits | 6 bits | 8 bits | 10 bits | 16 bits | 32 bits | 48 bits | 64 bits |
|----------------|--------|--------|--------|---------|---------|---------|---------|---------|
| LSH            | 0.0153 | 0.0288 | 0.0264 | 0.0514  | 0.0712  | 0.0899  | 0.1689  | 0.2040  |
| PCAH           | 0.0299 | 0.0363 | 0.0375 | 0.0456  | 0.0451  | 0.0304  | 0.0401  | 0.0343  |
| ITQ            | 0.0401 | 0.0707 | 0.1049 | 0.1373  | 0.2106  | 0.2728  | 0.3028  | 0.3321  |
| DSH            | 0.0277 | 0.0438 | 0.0499 | 0.0579  | 0.123   | 0.1261  | 0.1877  | 0.2422  |
| SGH            | 0.0381 | 0.0563 | 0.0717 | 0.0657  | 0.0874  | 0.1497  | 0.1692  | 0.2115  |
| COSDISH        | 0.2348 | 0.2518 | 0.2598 | 0.2224  | 0.3032  | 0.3674  | 0.4416  | 0.4648  |
| SDH            | 0.2526 | 0.2763 | 0.3263 | 0.3403  | 0.3713  | 0.4189  | 0.4275  | 0.4320  |
| FSSH           | 0.4896 | 0.5715 | 0.5718 | 0.6021  | 0.6298  | 0.6732  | 0.6845  | 0.6952  |
| SCDH           | 0.4636 | 0.5754 | 0.6059 | 0.6202  | 0.6521  | 0.6714  | 0.6854  | 0.6901  |
| SCDMH          | 0.5555 | 0.6057 | 0.6464 | 0.6610  | 0.6853  | 0.7092  | 0.7172  | 0.7204  |

Table 2. Performance in terms of mAP score on Mnist.

| Methods/Length | 4 bits | 6 bits | 8 bits | 10 bits | 16 bits | 32 bits | 48 bits | 64 bits |
|----------------|--------|--------|--------|---------|---------|---------|---------|---------|
| LSH            | 0.0198 | 0.0359 | 0.0607 | 0.0381  | 0.1544  | 0.2676  | 0.4772  | 0.5355  |
| PCAH           | 0.0480 | 0.0832 | 0.1473 | 0.1740  | 0.3506  | 0.4091  | 0.4093  | 0.3896  |
| ITQ            | 0.0586 | 0.1148 | 0.2425 | 0.2874  | 0.5411  | 0.7176  | 0.8130  | 0.8468  |
| DSH            | 0.0392 | 0.0446 | 0.0639 | 0.1104  | 0.1624  | 0.4277  | 0.6097  | 0.6353  |
| SGH            | 0.0613 | 0.1013 | 0.1944 | 0.2375  | 0.3668  | 0.5433  | 0.6478  | 0.6541  |
| COSDISH        | 0.4275 | 0.7601 | 0.8053 | 0.8346  | 0.8499  | 0.8980  | 0.9007  | 0.9065  |
| SDH            | 0.4527 | 0.5423 | 0.5850 | 0.6500  | 0.6788  | 0.7503  | 0.7662  | 0.7672  |
| FSSH           | 0.3893 | 0.7828 | 0.7046 | 0.8759  | 0.9580  | 0.9613  | 0.9660  | 0.9678  |
| SCDH           | 0.7709 | 0.9340 | 0.9434 | 0.9499  | 0.9553  | 0.9596  | 0.9603  | 0.9626  |
| SCDMH          | 0.8397 | 0.9371 | 0.9503 | 0.9532  | 0.9613  | 0.9637  | 0.9747  | 0.9759  |
Table 3. Performance in terms of mAP score on Caltech256.

| Methods/Length | 4 bits | 6 bits | 8 bits | 10 bits | 16 bits | 32 bits | 48 bits | 64 bits |
|---------------|--------|--------|--------|--------|--------|--------|--------|--------|
| LSH           | 0.009  | 0.0094 | 0.0161 | 0.0163 | 0.0414 | 0.2244 | 0.2855 | 0.3778 |
| PCAH          | 0.0201 | 0.0396 | 0.0611 | 0.0904 | 0.1610 | 0.3216 | 0.3901 | 0.4228 |
| ITQ           | 0.0263 | 0.0592 | 0.1140 | 0.2017 | 0.4343 | 0.6811 | 0.7289 | 0.7739 |
| DSH           | 0.0120 | 0.0132 | 0.0382 | 0.0315 | 0.0567 | 0.1539 | 0.2730 | 0.2723 |
| SGH           | 0.0211 | 0.0390 | 0.0711 | 0.1149 | 0.3730 | 0.6132 | 0.6574 | 0.7085 |
| COSDISH       | 0.0143 | 0.0427 | 0.1068 | 0.2199 | 0.4143 | 0.5829 | 0.6577 | 0.7005 |
| SDH           | 0.0410 | 0.0624 | 0.1467 | 0.3131 | 0.5315 | 0.6067 | 0.6421 | 0.6728 |
| FSSH          | 0.0365 | 0.0769 | 0.237 | 0.3715 | 0.4328 | 0.651 | 0.6927 | 0.7051 |
| SCDH          | 0.0574 | 0.1639 | 0.3393 | 0.4564 | 0.5748 | 0.6687 | 0.7088 | 0.7290 |
| SCDMH         | 0.0580 | 0.1810 | 0.3545 | 0.4648 | 0.5984 | 0.6810 | 0.7120 | 0.7396 |

Figure 4. Performance in terms of the precision score based on four benchmark datasets.

5. Conclusion

In this paper, we proposed a method called strong constraint discrete manifold-embedding hashing (SCDMH), where the information loss was reduced by mutual reconstruction of original feature and binary codes. At the same time, our method directly preserves manifold structure in the Hamming space. The proposed method achieved more stable and precise performances in image retrieval. From the experiment we can see that SCDMH outperforms other state-of-the-art supervised learning at hashing methods for largescale image retrieval. Experiments conducted on three benchmark datasets indicated that the proposed method exhibits superior performance, compared to the other existing methods. In future, we will attempt to reduce the time costs of it.

References

[1] Ruslan Salakhutdinov and Geoffrey E. Hinton, Semantic Hashing. In Int. J. Approx. Reasoning, 969–978, 2009. 1
[2] Aristides Gionis, Piotr Indyk and Rajeev Motwani, Similarity Search in High Dimensions via Hashing. In VLDB, 518–529, 1999. 1, 6
[3] Yunchao Gong, Svetlana Lazebnik, Albert Gordo and Florent Perronnin, Iterative Quantization: A Procrustean Approach to Learning Binary Codes for Large-Scale Image Retrieval. In IEEE Trans. on Pattern Anal. Mach.Intell., 2916–2929, 2013. 1, 6
[4] M. Datar, N. Immorlica, P. Indyk, and V. Mirrokni, "Locality-sensitive hashing scheme based on p-stable distributions", in Symposium on Computational Geometry, 2004.
[5] Yair Weiss, Antonio Torralba and Robert Fergus, Spectral Hashing. In NIPS, 1753–1760, 2008.
[6] Q. Liu, G. Liu, L. Li, X. Yuan, M. Wang and W. Liu, "Reversed Spectral Hashing," in IEEE Transactions on Neural Networks and Learning Systems, vol. 29, no. 6, pp. 2441-2449, June 2018, doi: 10.1109/TNNLS.2017.2696053.
[7] Y. Gong, S. Lazebnik, A. Gordo, and F. Perronnin, "Iterative Quantization: A Procrustean Approach to Learning Binary Codes for Large-Scale Image Retrieval", in IEEE Trans. Pattern Anal. Mach. Intell, 2013.
[8] W. Liu, C. Mu, S. Kumar, and S. Chang, "Discrete Graph Hashing", in Proceeding of NIPS, 2014.
[9] G. Irie, Z. Li, X. Wu, and S. Chang, "Locally Linear Hashing for Extracting Non-linear Manifolds", in Proceeding of CVPR, 2014.
[10] Q. Jiang, and W. Li, "Scalable Graph Hashing with Feature Transformation", in Proceeding of IJCAI, 2015.
[11] Wei Liu, Jun Wang, Rongrong Ji, Yu-Gang Jiang and Shih-Fu Chang, Supervised Hashing with Kernels. In CVPR, 2074–2081, 2012. 1, 2, 4, 6, 9
[12] Fumin Shen, Chunhua Shen, Wei Liu and Heng Tao Shen, Supervised Discrete Hashing. In CVPR, 37–45, 2015. 2, 6, 12
[13] Xin Luo, Liqiang Nie, Xiangnan He, Ye Wu, Zhen-Duo Chen and Xin-Shun Xu, Fast Scalable Supervised Hashing. In SIGIR, 735–744, 2018.2, 6, 9, 12
[14] Y. Chen, Z. Tian, H. Zhang, J. Wang and D. Zhang, "Strongly Constrained Discrete Hashing," in IEEE Transactions on Image Processing, vol. 29, pp. 3596-3611, 2020.
[15] G. Lin, C. Shen, and A. van den Hengel, “Supervised hashing using graph cuts and boosted decision trees,” IEEE transactions on pattern analysis and machine intelligence, vol. 37, no. 11, pp. 2317–2331, 2015.
[16] J. Güt, T. Liu, Z. Sun, D. Tao, and T. Tan, “Supervised discrete hashing with relaxation,” IEEE transactions on neural networks and learning systems, 2016.
[17] Q. Liu and H. Lu, “Natural supervised hashing,” in International Joint Conferences on Artificial Intelligence, pp. 1788–1794, 2016.
[18] Z. Lai, Y. Chen, J. Wu, W. K. Wong, and F. Shen, ”Jointly sparse hashing for image retrieval,” IEEE Transactions on Image Processing, 2018.
[19] X. Liu, X. Nie, Q. Zhou, X. Xi, L. Zhu, and Y. Yin, “Supervised short-length hashing,” in 28th International Joint Conference on Artificial Intelligence, pp. 3031–3037, 2019.
[20] X. Liu, X. Nie, Q. Zhou, and Y. Yin, “Supervised discrete hashing with mutual linear regression,” in ACM Multimedia, pp. 1561–1568, 2019.
[21] Wang-Cheng Kang, Wu-Jun Li and Zhi-Hua Zhou, Column Sampling Based Discrete Supervised Hashing. In AAAI, 1230–1236, 2016. 2, 3, 6, 12
[22] Yi Yang, Yueting Zhuang, Fei Wu, and Yunhe Pan. Harmonizing hierarchical manifolds for multimedia document semantics understanding and cross-media retrieval. TMM, 10(3):437–446, 2008.
[23] Lei Zhang, Yongdong Zhang, Richang Hong, and Qi Tian. Full-space local topology extraction for cross-modal retrieval. TIP, 24(7):2212–2224, 2015.
[24] Fumin Shen, Chunhua Shen, Qinfeng Shi, Anton van den Hengel, Zhenmin Tang, and Heng Tao Shen. Hashing on nonlinear manifolds. TIP, 24(6):1839–1851, 2015.
[25] Rongrong Ji, Hong Liu, Liujuan Cao, Di Liu, Yongjian Wu, and Feiyue Huang. Toward optimal manifold hashing via discrete locally linear embedding. TIP, 26(11):5411–5420, 2017.
[26] Arthur E. Hoerl and Robert W. Kennard. Ridge regression: Applications to nonorthogonal problems. Technometrics, pages 69–82, 1970.
[27] F. McSherry and M. Najork, “Computing information retrieval performance measures efficiently in the presence of tied scores,” in European conference on information retrieval, Conference Proceedings, pp. 414–421.
[28] Yunchao Gong and Svetlana Lazebnik, Iterative Quantization: A Procrustean Approach to Learning Binary Codes. In CVPR, 817–824, 2011.
[29] Fumin Shen, Xiang Zhou, Yang Yang, Jingkuan Song, Heng Tao Shen and Dacheng Tao, A Fast Optimization Method for General Binary Code Learning. In IEEE Trans. on Image Processing, 5610–5621, 2016. 2, 3, 10
[30] Y. Sun and S. Yu, "Deep Supervised Hashing with Dynamic Weighting Scheme," 2020 5th IEEE International Conference on Big Data Analytics (ICBDA), 2020, pp. 57-62, doi: 10.1109/ICBDA49040.2020.910274.