sensors

MDPI

Article

An Online Hashing Algorithm for Image Retrieval Based on Optical-Sensor Network

Xiao Chen [1], Yanlong Li [1,2,*] and Chen Chen [1]

[1] Department of Information and Communication, Guilin University of Electronic Technology, Guilin 541004, China
[2] Ministry of Education Key Laboratory of Cognitive Radio and Information Processing, Guilin University of Electronic Technology, Guilin 541004, China
* Correspondence: lylong@guet.edu.cn

Abstract: Online hashing is a valid storage and online retrieval scheme, which is meeting the rapid increase in data in the optical-sensor network and the real-time processing needs of users in the era of big data. Existing online-hashing algorithms rely on data tags excessively to construct the hash function, and ignore the mining of the structural features of the data itself, resulting in a serious loss of the image-streaming features and the reduction in retrieval accuracy. In this paper, an online hashing model that fuses global and local dual semantics is proposed. First, to preserve the local features of the streaming data, an anchor hash model, which is based on the idea of manifold learning, is constructed. Second, a global similarity matrix, which is used to constrain hash codes is built by the balanced similarity between the newly arrived data and previous data, which makes hash codes retain global data features as much as possible. Then, under a unified framework, an online hash model that integrates global and local dual semantics is learned, and an effective discrete binary-optimization solution is proposed. A large number of experiments on three datasets, including CIFAR10, MNIST and Places205, show that our proposed algorithm improves the efficiency of image retrieval effectively, compared with several existing advanced online-hashing algorithms.

Keywords: optical-sensor network; manifold learning; balanced similarity; discrete binary optimization; image retrieval

Citation: Chen, X.; Li, Y.; Chen, C. An Online Hashing Algorithm for Image Retrieval Based on Optical-Sensor Network. *Sensors* **2023**, *23*, 2576. https://doi.org/10.3390/s23052576

Academic Editors: Syed Agha Hassnain Mohsan, Wali Ullah Khan and Dinh-Thuan Do

Received: 27 December 2022
Revised: 20 February 2023
Accepted: 24 February 2023
Published: 25 February 2023

1. Introduction

With the popularization of optical-sensor networks and the wide use of intelligent interconnected devices, data in various fields are increasing at an unbelievable speed. People realize that intelligent processing and analysis of data is necessary [1–3] and it is of great significance to store high-dimensional data effectively and retrieve data rapidly. The traditional indexing methods involving text-based image retrieval (TBIR) and content-based image retrieval (CBIR) [4,5] encounter the curse of dimensionality in high-dimensional situations, and their query performance is even worse than linear query. An approximate nearest-neighbor query based on the hash method is an efficient method to solve the above issue [6,7]. Specifically, in image retrieval systems, image hashing means mapping a high-dimensional real-valued image to a compact binary code, which can preserve the relationship between different high-dimensional data and the Hamming space [8–11].

According to the dependency between the hash model and the sample data, hashing algorithms include data-independent algorithms and data-dependent algorithms. The representative data-independent algorithms include locality-sensitive hashing (LSH) [12,13], and its variants such as ℓ_p—stable hashing [14], min-hash [15], and kernel LSH (KLSH) [16]. Data-dependent hashing algorithms include unsupervised hashing algorithms [17–19] and supervised hashing algorithms [20–23]. Since methods using data distribution or class labels perform better in the quick search field, more effort is being put into data-dependent methods.

In fact, data samples always arrive sequentially, as time goes on, and thus the previously existing data is often out of date [24,25]. When the discrepancy between newly arrived data and previously existing data is large, the hashing function often loses efficiency on newly arrived data. Therefore, it is very important to present an online hashing model that is suitable for streaming data. Unlike the offline hashing methods, which correct the training error on the fixed dataset through multiple training rounds, online algorithms use multiple batches of streaming data to update hash functions, which are more realistic and have a strong application background [26–31].

Several classic online hashing methods have emerged in recent years, and they are all data-dependent. Representative works include Online Hashing (OKH) [24], Adaptive Hashing (AdaptHash) [32], Online Supervised Hashing (OSH) [33], Online Hashing with Mutual Information (MIHash) [34], Balanced Similarity for Online Discrete Hashing (BSODH) [35], Supervised Online Hashing via Hadamard Codebook Learning (HCOH) [36], Hadamard Matrix Guided Online Hashing (HMOH) [37], etc.

Most of the mentioned online-hashing algorithms consider the adaptability, pairwise similarity, or independence of hash codes to build a constrained hashing model, but the optimization needs relaxation learning, which brings quantization errors to a certain extent and reduces the retrieval accuracy. In addition, there exist unsupervised online hashing methods, which are mostly based on the idea of "matrix sketch", and its representative works mainly consist of Online Sketching Hashing (SketchHash) [38], Faster Online Sketching Hashing (FROSH) [39], and so on. From the setting of online hashing, the global data grow dynamically with the current arriving data, which just represent the local data at a certain stage. Unsupervised methods that only rely on the distribution of newly arrived data lack a global description of the hashing model.

In image-retrieval application systems, the labeling is carried out manually and the workload is also huge. In addition, manual labeling is prone to errors, and wrong labels will directly lead to retrieval failure. Therefore, the online hashing method that relies too much on data labels while ignoring the structural characteristics of the data itself is subject to many limitations in practical applications, which seriously affects the performance of retrieval accuracy [40,41].

The high-dimensional image data remains on the low-dimensional manifold structure [42], and the query data is often strongly correlated. Therefore, this paper proposes an online hashing model that fuses global and local dual semantics. First, an anchor hash model is built based on manifold learning to retain local features of the original data. Then, a global similarity matrix which is used to constrain the hash codes is constructed, employing the balanced similarity between newly arrived data and previous data [35]. Under a unified framework, an online hash model integrating global and local dual semantics is learned, as well as an effective discrete-binary-optimization scheme being proposed. Compared with several classical and well-established online hashing algorithms, our proposed LSOH method has advantages in many performance indicators.

In summary, the main contributions of our work are as follows:

- Extract the manifold structure of high-dimensional data using Laplacian Eigenmaps, thus constructing an anchor hash model.
- Construct an asymmetric-graph regularization term to constrain the learning process of hash codes using the balanced similarity between current arriving data and previous data sets.
- Integrate the anchor hash model and the asymmetric graph regularization with a seamless formulation to learn global and local dual-semantic information, then use the alternating-iteration algorithm to solve the optimization issue and obtain high retrieval accuracy by performing a large number of experiments.

The remaining contents are arranged as follows. In Section 2, related work in this field is reviewed. In Section 3, we present our proposed online-hashing algorithm, including the optimization method. Section 4 presents the experimental results and analyses in detail. Finally, we give a conclusion of our work in Section 5.

2. Related Work

In this section, online hashing algorithms, such as OKH [24], AdaptHash [32], OSH [33], MIHash [34], BSODH [35], HCOH [36], HMOH [37], SketchHash [38] and FROSH [39] are introduced. Among them, all are supervised hashing methods except SketchHash and FROSH. Supervised hashing is more efficient owing to the semantic data, but online retrieval datasets are often prone to missing labels and labeling errors, while hashing methods without labels are more suitable for massive online-retrieval applications.

Huang et al. [24] presented an online hashing algorithm using a kernel function, termed OKH. First, OKH employes the kernel-based hash function to process linearly inseparable data. Then, OKH formulates an objective function based on the inner product of binary codes. They consider the equivalence between optimizing the inner product of binary codes and Hamming distance, and use the greedy algorithm to solve the hash function effectively. It solves the non-convex optimization problem of Hamming distance. Experiments show that it can be widely applied to image-retrieval scenarios.

Similar to OKH's framework, AdaptHash [32] proposes a fast similarity-search algorithm for hash functions, based on the stochastic gradient descent method. Specifically, it defines a hinge-loss function to determine the number of hash functions that need to be updated in Adaptive Hash and optimizes the model by SGD.

Cakir et al. [33] proposed an adaptive online-hashing method based on Error Correcting Output Codes (ECOC), named OSH. No prior assumptions about label space are made and it is the first supervised hashing algorithm suitable for the growth of label space. OSH presents a two-step hashing framework, first generating ECOC as codebooks, and then assigning codewords to each class label. Finally, the exponential loss is optimized and solved by SGD, to ensure that the learned hash function is suitable for binary ECOC.

Based on the knowledge of information theory, MIHash [34] takes mutual information as the learning objective and proposes a measure to eliminate the updates of the unnecessary hash tables. Thus, they optimize the mutual information objective by stochastic gradient descent. The computational complexity is effectively reduced, and the learning efficiency of the hash function is improved.

BSODH [35] believes that there are two unsolved problems: update imbalance and optimization inefficiency, which lead to the unsatisfactory performance of OH in practical applications. In this paper, two balance parameters are introduced to improve the regularization term of asymmetric graphs. Theoretical analysis and extensive experiments verify the role of parameters in alleviating the unbalanced update. It is also the first time discrete optimization has been applied to online hashing, which improves the online hashing performance.

Lin et al. [36] believe that because of the flaw of unknown category numbers in supervised learning, it does not improve the efficiency of online hash retrieval, despite the addition of semantic information. Therefore, they propose a robust supervised online-hashing scheme, termed HCOH. First, a high-dimensional orthogonal binary matrix, i.e. the Hadamard matrix, is generated. Every column or row of this matrix can be taken as a codebook that corresponds with a class label. Then, LSH is used to convert the codebook into a binary code adapted to the number of hash bits. In an improved version of HMOH [37], hash linear regression is processed as a binary-classification issue, and the case of multi-label is considered as well.

Aimed at the problem of the data embedding into the system in a streaming way and the difficulty of loading into memory for training because of the huge dataset, SketchHash [38] decreases the size of the dataset based on the idea of data sketches, and retains the main features of the dataset to learn an effective hash function. This approach reduces computational complexity and space complexity. Compared with SketchHash, the FROSH [39] method leverages fast transformation to sketch data more compactly. FROSH applies a specific transform on different small data blocks, speeding up the procedure of sketching with the same space cost.

There is also semi-supervised online hashing [43,44], which is relatively complicated because labels may come from existing data or streaming data. In addition, deep-hashing methods [40,45,46] occupy a very important position in the existing offline-hashing methods. However, there are large amounts of parameters to be trained in deep learning, and few examples are applied in online hashing at present. Among them, Online Self-Organizing Hashing [47] obtains hash codes by the Self-Organizing Map (SOM) algorithm, but SOMs with multi-layers structures have not been applied to image retrieval.

3. The Proposed Method

In this section, the variable symbols in this algorithm are first defined, and the modeling process that combines the local structural features and the similarity features of the global datasets is given. Finally, we obtain the objective function and solve it by the alternating-iteration method. The algorithm frame is shown in Figure 1.

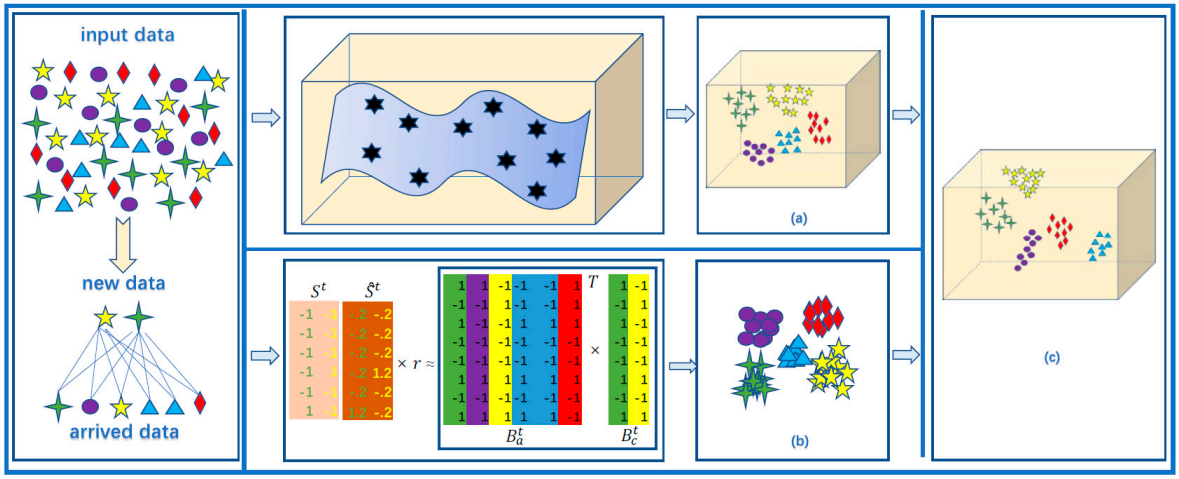


Figure 1. The overall framework of online hashing preserving local features and global-balanced similarity (LSOH). (**a**) Manifold learning preserves the structural features of newly arrived data, and obtains the hash codes of newly arrived data through Laplacian Eigenmaps (LE). (**b**) Learn binary codes by a balanced similarity matrix built from newly arrived data and existing data to keep all the hash codes consistent. (**c**) Our proposed algorithm can learn hash codes preserving dual-semantic information, and obtain satisfactory retrieval results.

3.1. Notations

Assume that n_t training samples are poured into retrieval application at t stage. They are denoted as $X^t = \left[x_1^t, x_2^t, \cdots, x_{n_t}^t\right] \in R^{d \times n_t}$ and their corresponding labels L^t are defined as $L^t = \left[l_1^t; l_2^t; \cdots; l_{n_t}^t\right] \in N^{d \times n_t}$. Each training sample expressed as x_i^t is d-dimensional. The goal of hashing is to learn r-dimensional hash codes, which are denoted as $B^t = \left[b_1^t, b_2^t, \cdots, b_{n_t}^t\right] \in \{1, -1\}^{r \times n_t}$, and meanwhile, r is much smaller than d. The linear-hash mapping is widely used as a hash function [48], i.e.,

$$B^t = F\left(X^t\right) = sgn\left(W^{t^T} X^t\right) \tag{1}$$

where $F(\cdot)$ stands for the hash function, $W^t \in R^{d \times r}$ is the projection matrix to be learned, W^{t^T} is the transpose of W^t, and $sgn(\cdot)$ is the symbolic function, and its definition is the following. All symbol notations utilized in this study are presented in Table 1.

$$f(x) = sgn(x) = \begin{cases} -1, & x < 0 \\ 1, & x \geq 0 \end{cases} \tag{2}$$

Table 1. Notations in this paper.

Symbol	Notations
X^t	input data at t stage
X_a^t	data existing before t stage
L^t	labels of X^t
L_a^t	labels of X_a^t
B^t	hash codes learned for X^t
B_a^t	hash codes learned for X_a^t.
W^t	hashing projection matrix at the t age
d	dimension of all input data
X_c^t	newly arrived data at the t stage
$\boldsymbol{k}$	dimension of every hash code
L_c^t	labels of X_c^t
$\boldsymbol{N}$	amount of input data
B_c^t	binary codes generated for X_c^t
n_t	amount of input data at the t stage

3.2. Manifold Learning

3.2.1. Laplacian Eigenmaps

Laplacian Eigenmaps use the Laplacian operator to make similar data in the original space as close as possible after being mapped to the low-dimensional space, so as to embed high-dimensional images in the low-dimensional space. Assume that the original data denoted as $X^t = \left[x_1^t, x_2^t, \cdots, x_{n_t}^t\right]$ are mapped to the low-dimensional space, in which hash codes are expressed as $B^t = \left[b_1^t; b_2^t; \cdots; b_{n_t}^t\right]$. We construct a graph whose adjacency matrix is O^t to maintain the relationships between different data. Then, we define the objective function to be optimized as follows:

$$\min_{B^t} \sum_{ij} \|b_i - b_j\|_2^2 {O_{ij}}^t \tag{3}$$

$$s.t.\ B^t \in \{-1, 1\}^{k \times n_t}, \quad {B^t}^T B^t = n_t I, \quad {O_{ij}}^t = \exp\left(-\frac{\|x_i - x_j\|^2}{\varphi^2}\right).$$

where I represents the k-dimensional identity matrix, and ${O_{ij}}^t$ represents the adjacency matrix between the sample data. Mathematically, the objective function to be optimized can be transformed into the formula as follows:

$$\min_{B^t} tr\left(B^t (D - O) {B^t}^T\right) \tag{4}$$

$$s.t.\ B^t \in \{-1, 1\}^{k \times n_t}, \quad {B^t}^T B^t = n_t I.$$

where $D_{ii} = \sum_i O_{ij}$ represents the weight matrix of the sample graph, and $D - O$ is the Laplace matrix. By eigen decomposition of $D - O$, the eigenvectors corresponding to the k smallest non-zero eigenvalues are obtained as the required target hash codes.

3.2.2. Anchor Graph Hashing

It is time-consuming and memory-intensive to compute the adjacency matrix for large amounts of sample data. Calculating the adjacency matrix by using an anchor set instead of the dataset can solve the above problem: m anchor points denoted as $[u_1, u_2, \ldots, u_i, \ldots, u_m] \in R^d$ are obtained through the k-mean clustering method. When the number of anchors is less than that of the training samples, both storage cost and

computation time are greatly reduced. The anchor graph is denoted as A^t and its elements are defined as follows:

$$A_{ij}{}^t = \begin{cases} \frac{exp\left(-\|x_i^t - u_j^t\|^2/\theta\right)}{\sum_{j' \in \{i\}} exp\left(-\|x_i^t - u_{jj'}^t\|^2/\theta\right)}, & \forall j \in \{i\} \\ 0, & otherwise \end{cases} \tag{5}$$

where θ is a defined parameter, and $\{i\}$ represents the index set of the k nearest anchor points. Replace the traditional Laplace matrix with anchor graph A^t, and then the objective function is obtained.

$$L_1 = \min_{\widetilde{B}^t, W^t} \sum_{j=1}^{m} \sum_{i=1}^{n} \|\widetilde{b}_j^t - W^{t\,T} x_i^t\|_2^2 A_{ij}^t, \tag{6}$$

where $\widetilde{B}^t = \left[\widetilde{b}_1^t, \widetilde{b}_2^t \ldots \widetilde{b}_j^t, \ldots \widetilde{b}_m^t\right]$ represents the hash codes of anchor points, $W^{t\,T} x_i^t$ represents the hash codes of the input images, and $A_{ij}{}^t$ represents the anchor graph matrix constructed by the input images and anchor data.

3.3. Global-Balanced Similarity

It performs the anchor hashing based on Laplacian Eigenmaps; the hash function of newly arrived data is obtained independently, which will ignore the correlation of the overall data and boost the redundancy of hash codes. Thus, a global similarity constraint is introduced to build an online hashing model.

3.3.1. Similarity

Suppose that the newly arrived data at t stage are denoted as $X_c^t = \left[X_{c1}^t, X_{c2}^t, \ldots, X_{cn_t}^t\right]$, while the existing data arriving before t stage are denoted as $X_a^t = \left[X_a^1, X_a^2, \ldots, X_a^{t-1}\right]$. The similarity matrix, S^t, is constructed by the relationships of the data labels between X_c^t and X_a^t. Each matrix element is defined as follows:

$$S_{ij}^t = \begin{cases} 1, & l_i^t = l_j^t \\ -1, & otherwise. \end{cases} \tag{7}$$

Generally speaking, the more similar the data, the smaller the hash distance. We use the inner product of hash codes to estimate the distance between different vectors in the Hamming space. Constraints on hash codes are constructed using the global similarity matrix which is defined above [35], as shown in Equation (8). Therefore, global semantic information at any stage of the input data remains in the Hamming space. Therefore, the loss function that preserves similarity is defined as follows:

$$L_2 = \min_{B_a^t, B_c^t} \|B_c^{t\,T} B_a^t - kS^t\|_{\mathcal{F}}^2 \tag{8}$$

$$\text{s.t.} B_c^t \in \{1, -1\}^{r \times n_t}, B_a^t \in \{1, -1\}^{r \times m_t}$$

where $m_t = \sum_{i=1}^{t-1} n_i$ represents the total amount of input data arriving before the t stage, $\|\cdot\|_{\mathcal{F}}$ refers to the F norm, and k is the bit length of hash codes.

3.3.2. Balanced Similarity

The introduction of a similarity matrix improves the hash codes generated by anchor hashing based on LE, and the global semantic information is better reflected. However, images with different labels among the massive data account for the majority. According to the definition of the global similarity matrix, the value of the element is 1 only when the labels are identical. Therefore, the global similarity matrix is sparse. Data imbalance will cause the loss of similar information, derail the optimization process and eventually drag

down the retrieval performance. To solve this issue, we employ the balanced similarity matrix $\widetilde{S}^t$ as follows:

$$\widetilde{S}_{ij}^t = \begin{cases} \mu_s S_{ij}^t, & S_{ij}^t = 1 \\ \mu_d S_{ij}^t, & S_{ij}^t = -1, \end{cases} \tag{9}$$

where μ_s represents the equilibrium factors of similar pairs, and μ_d represents the equilibrium factors of dissimilar pairs. Usually, we take $\mu_s < 1$ and $\mu_d < 1$, which means reducing the Hamming distance of similar vectors and increasing the Hamming distance of dissimilar vectors. By adjusting two equilibrium divisors, the effect that comes from data imbalance is eliminated. By replacing the global similarity matrix, S^t, in Equation (8) with a balanced similarity matrix, $\widetilde{S}^t$, the loss function that preserves the balanced similarity is defined as follows:

$$L_2 = \min_{B_a^t, B_c^t} \| {B_c^t}^T B_a^t - k\widetilde{S}^t \|_{\mathcal{F}}^2 \tag{10}$$

$$\text{s.t.} B_c^t \in \{1, -1\}^{r \times n_t}, B_a^t \in \{1, -1\}^{r \times m_t}$$

3.4. Overall Formulation

On one hand, we construct the anchor asymmetric graph to replace the Laplacian graph, preserving the local structural features of the data, and thus obtaining the objective function, L_1. On the other hand, we perform an inner-product operation on the hash codes of existing data and newly arrived data and then constrain the learning process of hash codes with the global-balanced similarity matrix. The loss function, L_2, that retains the semantic information of global-balanced similarity is obtained. Under a unified framework, the online hashing preserves both local and global dual-semantic information, and the total loss function $L = L_1 + L_2$ is obtained. By adding a quantized loss function, L_3, the quantization error between the hash function and the target hash codes is minimized.

$$L_3 = \min_{W^t} \| F(X^t) - B^t \|_{\mathcal{F}}^2 \tag{11}$$

The F norm of the projection matrix is used as the penalty term to prevent the model from overfitting. The final objective function is obtained as follows:

$$\begin{aligned} L = \min_{B_c^t, B_a^t, \widetilde{B}^t, W^t} \alpha^t \sum_{j=1}^{m} \sum_{i=1}^{n} \| \widetilde{b}_j^t - {W^t}^T x_i^t \|_2^2 A_{ij}^t + \| {B_c^t}^T B_a^t - k\hat{S}^t \|_{\mathcal{F}}^2 \\ + \beta^t \| {W^t}^T X_c^t - B_c^t \|_{\mathcal{F}}^2 + \gamma^t \| W^t \|_{\mathcal{F}}^2 \end{aligned} \tag{12}$$

where α^t, β^t, and γ^t are parameters that control the weight of each module.

3.5. Alternating Optimization

Because of the binary constraints, Equation (12) is a non-convex objective function in terms of W^t, B_a^t, B_c^t, $\widetilde{B}^t$. We adopt an alternative optimization approach to deal with the overall formula, L; i.e., and when a variable is updated we assume that the remaining variables are fixed as constants.

- W^t-step: fix B_a^t, B_c^t, $\widetilde{B}^t$, then learn hash weights W^t. The second term in Equation (12) is eliminated, and the objective function becomes:

$$\min_{W^t} \alpha^t \sum_{j=1}^{m} \sum_{i=1}^{n} \| \widetilde{b}_j^t - {W^t}^T x_i^t \|_2^2 A_{ij}^t + \beta^t \| {W^t}^T X_c^t - B_c^t \|_{\mathcal{F}}^2 + \| \gamma^t W^t \|_{\mathcal{F}}^2 \tag{13}$$

We transform and simplify Equation (13) by Equation (14), which reveals the relation between F norm and the trace of a matrix. Then, Equation (15) is obtained.

$$\| A \|_F = \sqrt{tr(A^T A)} = \sqrt{tr(A A^T)} \tag{14}$$

$$\min_{W^t} \left[\left(\alpha^t + \beta^t \right) X_c^t X_c^{t^T} + \gamma^t I \right] tr\left(W^t W^{t^T} \right) - 2tr\left(W^{t^T} X_c^t \left(\alpha^t A^t \widetilde{B}^{t^T} + \beta^t B_c^{t^T} \right) \right) \tag{15}$$

where I represents the identity matrix of d-dimensional. Equation (15) takes the partial derivative with respect to W^t, then assigns it zero. We have the following formula:

$$\left[\left(\alpha^t + \beta^t \right) X_c^t X_c^{t^T} + \gamma^t I \right] W^t - X_c^t \left(\alpha^t A^t \widetilde{B}^{t^T} + \beta^t B_c^{t^T} \right) = 0 \tag{16}$$

Thus, we can get the Equation (17) to update W^t

$$W^t = \left[\left(\alpha^t + \beta^t \right) X_c^t X_c^{t^T} + \gamma^t I \right]^{-1} X_c^t \left(\alpha^t A^t \widetilde{B}^{t^T} + \beta^t B_c^{t^T} \right) \tag{17}$$

- B_a^t-step: fix W^t, B_c^t, $\widetilde{B}^t$, the second term in Equation (12) is retained and the formula becomes:

$$\min_{B_a^t} \| B_c^{t^T} B_a^t - k\hat{S}^t \|_{\mathcal{F}}^2 \tag{18}$$

According to [49], the $L1$ norm replaces the F norm, and the result is as follows:

$$B_a^t = \mathrm{sgn}(B_c^t \hat{S}^t) \tag{19}$$

- B_c^t-step: fix W^t, B_a^t, $\widetilde{B}^t$, the first and the fourth term in Equation (12) are eliminated, and the corresponding sub-problem is:

$$\min_{B_c^t} \| B_c^{t^T} B_a^t - k\hat{S}^t \|_{\mathcal{F}}^2 + \beta^t \| W^{t^T} X_c^t - B_c^t \|_{\mathcal{F}}^2 \tag{20}$$

In Equation (20), we remove irrelevant terms and the optimization problem becomes:

$$\min_{B_c^t} \| B_a^{t^T} B_c^t \|_{\mathcal{F}}^2 - 2tr\left(P^T B_c^T \right) \tag{21}$$

where $tr(\cdot)$ is trace norm, $P = kB_a^t \hat{S}^{t^T} + \beta^t W^{t^T} X_c^t$. In the light of supervised discrete hashing (SDH) [50] and BSODH [35], the solution of Equation (21) becomes NP hard. Therefore, we transfer this issue to row-by-row updating, considering that the matrix is made up of row vectors. Thus, Equation (21) becomes:

$$\min_{\widetilde{b}_{cr}^t} \| \widetilde{b}_{ar}^{t^T} \widetilde{b}_{cr}^t + \widetilde{B}_a^{t^T} \widetilde{B}_c^t \|_{\mathcal{F}}^2 - 2tr\left(\widetilde{p}_r^{t^T} \widetilde{b}_{cr}^t + \widetilde{P}^{t^T} \widetilde{B}_c^t \right) \tag{22}$$

where $\widetilde{b}_{cr}^t$, $\widetilde{b}_{ar}^t$ and $\widetilde{p}_r^t$ are the rth row of B_c^t, B_a^t and $\widetilde{P}^t$, respectively; B_c^t, B_a^t and $\widetilde{P}^t$ stand for the remaining parts of B_c^t, B_a^t and $\widetilde{P}^t$ except $\widetilde{b}_{cr}^t$, $\widetilde{b}_{ar}^t$ and $\widetilde{p}_r^t$ respectively. Expanding Equation (22), we get the following:

$$\min_{\widetilde{b}_{cr}^t} \| \widetilde{b}_{ar}^{t^T} \widetilde{b}_{cr}^t \|_{\mathcal{F}}^2 + \| \widetilde{B}_a^{t^T} \widetilde{B}_c^t \|_{\mathcal{F}}^2 + 2tr\left(\widetilde{B}_c^{t^T} \widetilde{B}_a^t \widetilde{b}_{ar}^{t^T} \widetilde{b}_{cr}^t \right) - 2tr\left(\widetilde{p}_r^{t^T} \widetilde{b}_{cr}^t \right) - 2tr\left(\widetilde{P}^{t^T} \widetilde{B}_c^t \right) \tag{23}$$

After simplification, the Equation (23) becomes the following:

$$\min_{\widetilde{b}_{cr}^t} tr\left(\left(\widetilde{B}_c^{t^T} \widetilde{B}_a^t \widetilde{b}_{ar}^{t^T} - \widetilde{p}_r^{t^T} \right) \widetilde{b}_{cr}^t \right) \tag{24}$$

Therefore, we solve the sub-problem by following updating rules below:

$$\widetilde{b}_{cr}^t = sgn\left(\widetilde{p}_r^t - \widetilde{b}_{ar}^t \widetilde{B}_a^{t^T} \widetilde{B}_c^t \right) \tag{25}$$

- $\widetilde{B}^t$-step: fix W^t, B_a^t, B_c^t, only the first term remains in Equation (12), and it is transformed into the formula, as follows:

$$\max_{\widetilde{B}^t} tr\left(W^{t^T} X_c^t A^t \widetilde{B}^{t^T}\right) \tag{26}$$

Finally, we get the following rule to update the hash codes of the anchor data.

$$\widetilde{B}^t = sgn\left(W^{t^T} X_c^t A^t\right). \tag{27}$$

The proposed LSOH is presented in Algorithm 1.

Algorithm 1 the online hashing preserving local features and global-balanced similarity

Input: training samples, $\boldsymbol{X}$; labels $\boldsymbol{L}$; code length, $\boldsymbol{k}$; the number of sample batches, $\boldsymbol{T}$; divisors $\boldsymbol{\alpha^t}$,$\boldsymbol{\beta^t}$, $\boldsymbol{\gamma^t}$.
Output: hash codes $\boldsymbol{B}$ and the mapping matrix $\boldsymbol{W}$.
Initialize $\boldsymbol{W}$ with the normal Gaussian distribution
while $\boldsymbol{T} \leftarrow \mathbf{1}$ **do**
Denote the newly arrived data as $\boldsymbol{X_c^t}$
Set $\boldsymbol{X_a^t} = [\boldsymbol{X_a^t}; \boldsymbol{X_c^t}]$, $\boldsymbol{B_a^t} = [\boldsymbol{B_a^t}; \boldsymbol{B_c^t}]$
Compute $\boldsymbol{m}$ anchor points $[\boldsymbol{u_1}, \boldsymbol{u_2}, \ldots, \boldsymbol{u_i}, \ldots, \boldsymbol{u_m}]$ by means of $\boldsymbol{K}$-means clustering
Obtain anchor graph $\boldsymbol{A^t}$ via Equation (5) and compute the global-balanced similarity matrix $\widetilde{\boldsymbol{S}}^{\boldsymbol{t}}$ by labels
Update $\boldsymbol{W^t}$, $\boldsymbol{B_a^t}$, $\widetilde{\boldsymbol{B}}^{\boldsymbol{t}}$ via Equation (17), Equation (19) and Equation (27), **respectively**
while $\boldsymbol{r}$ becomes $k \leftarrow \mathbf{1}$ **do**
Update $\widetilde{\boldsymbol{b}}_{\boldsymbol{cr}}^{\boldsymbol{t}}$ via Equation (25)
end while
end while
Set $\boldsymbol{W} = \boldsymbol{W^t}$ and calculate $\boldsymbol{B^t} = \boldsymbol{sgn}\left(\boldsymbol{W^{t^T} X^t}\right)$
Return $\boldsymbol{W}$, $\boldsymbol{B}$

3.6. Computational Complexity

The main computation cost of the iterative algorithm is from the construction of anchor graph A and the optimization of variables. In total, it costs O($Tdn_tm + dn_tm$) to construct an anchor graph, where T is the iteration times to generate anchors by the clustering algorithm. The time complexities of updating W^t, B_a^t , B_c^t and $\widetilde{B}^t$ at the tth round are O($d^3 + n_td^2 + mn_tr + n_tdr$), O(r$n_tm_t$), O(km$n_t$+ kd$n_t + krm + krn_t$) and O(dr$n_t + mrn_t$) respectively. Since d, r, m, n_t, and k are much smaller than m_t, we obtain the time complexity of our proposed LSOH as linear to the size of the data. Obviously, it is scalable to large-scale data.

4. Experiments

4.1. Datasets

The three datasets used in this paper are CIFAR-10, MNIST, and Places205.

CIFAR-10 [51] is a widely recognized dataset. It is made up of 60K samples among 10 classes. Every sample is represented by a 4096-dimensional CNN feature. Following [34], CIFAR-10 is divided into a retrieval set and a test set, where the retrieval set has 59K samples and the test set has 1K samples. The 50K samples within the retrieval set participate in the learning hash function. Twenty example images from each category of CIFAR-10 are shown in Figure 2.

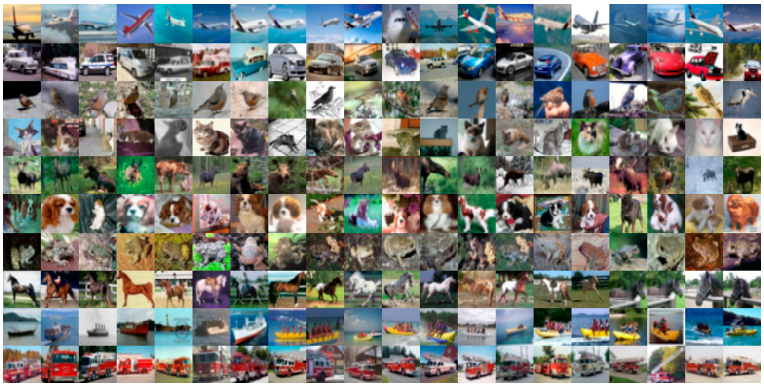


Figure 2. Example images of CIFAR-10 dataset.

MNIST is a set of handwritten digit images, with a total of 70 K samples. Every sample is expressed as a 784-dimensional vector. A test set is constructed by sampling 100 samples from each category and the remaining samples form a retrieval set. There are 60 K instances among the retrieval set which participate in the learning hash function. We select 27 example instances randomly from each category, as shown in Figure 3.

Figure 3. Example images of MNIST dataset.

Places205 has 2.5 million scene images among 205 categories. The instances are firstly processed by the fc7 layer of AlexNet and are fallen to 128-dimensional vectors by PCA. The 20 samples are randomly selected from each category forming the test set. Thus, the rest form the retrieval set. We select a subset of 30K instances randomly from the retrieval set to learn the online hash function. The two hundred images shown in Figure 4 are sampled randomly from Places205.

Figure 4. Example images of Places205 dataset.

4.2. Experimental Setting

4.2.1. Parameter Setting

Given experience, the scopes of α^t, β^t, γ^t for LSOH are set in [0:0.05:1]. For CIFAR-10, the best setting of (α^t, β^t, γ^t) is empirically adopted to (0.05, 0.6, 0.3). For the MNIST, we set (0.05, 0.6, 0.3) as the configuration of (α^t, β^t, γ^t). For Places205, (0.05, 0.8, 0.3) corresponds to (α^t, β^t, γ^t). Table 2 shows the specific parameters of our proposed LSOH on the three datasets mentioned above. We conduct a large number of experiments, where the bit length is taken from the set of [8,16,32,48,64,128]. In addition, the batch size should be greater than that of hash codes in SketchHash [38]. Therefore, the experimental results of SketchHash are presented only when the hashing codes are under 64 bits.

Table 2. Parameter settings on three datasets of LSOH.

Parameter	CIFAR-10	MNIST	Places205
α^t	0.05	0.05	0.05
β^t	0.6	0.6	0.8
γ^t	0.3	0.3	0.3
μ_s	1.2	1.2	1
μ_d	0.2	0.2	0
n_t	5000	10,000	10,000

4.2.2. Evaluation Protocols

Some evaluation indicators are adopted, such as the mean average precision (mAP), precision within a Hamming sphere with a radius of 2 centered on every query point (Precision@H2), and the precision of the top-K retrieved neighbors (Precision@K) to evaluate the proposed LSOH. It is worth noting that we apply the average accuracy of the first 1000 retrieved samples (mAP@1000) for Places205, to save calculation time. We adopt the precision–recall (PR) curves on MNIST and CIFAR-10 as well, to compare LSOH and several algorithms.

4.2.3. Compared Methods

To prove the effectiveness of LSOH, we perform abundant experiments and compare LSOH with several advanced OH algorithms such as OKH [24], SketchHash [38], AdaptHash [32], OSH [33], BSODH [35] and DSBOH [52]. The codes of the above comparison methods are publicly available. All the results of the above methods are achieved on a single computer that runs MATLAB and is equipped with a 3.0 GHz Intel Core i5-8500CPU and 16GB RAM. To reduce the error, each experiment was randomly run three times, and then the average is given in this work.

4.3. Results and Discussion

The values of mAP and Precision@H2 on the CIFAR-10 dataset are shown in Table 3. It lists the results when generating 8-bit, 16-bit, 32-bit, 48-bit, 64-bit, and 128-bit hash codes under different online methods. The results show that (1) mAP: values of our proposed LSOH are the highest in all the cases. Our proposed LSOH improves the accuracy by 3.3%, 0.4%, 1.9%, 3.4%, 1.5%, and 1.5%, respectively, over the second-best algorithm. It can be seen that LSOH improves the average accuracy, effectively. (2) Precision@H2: our proposed LSOH algorithm is 2.4% higher than the suboptimal algorithm in the situation of 48-bit. It is the second-best algorithm in the case of 8-bit, 16-bit, 32-bit, 64-bit, and 128-bit, while no algorithm can rank first in all the cases. In this way, LSOH still performs well, compared with other algorithms.

Table 3. mAP and Precision@H2 comparisons on CIFAR-10.

	mAP						Precision@H2					
Methods	**8-bit**	**16-bit**	**32-bit**	**48-bit**	**64-bit**	**128-bit**	**8-bit**	**16-bit**	**32-bit**	**48-bit**	**64-bit**	**128-bit**
OKH	0.100	0.134	0.223	0.252	0.268	0.350	0.100	0.175	0.100	0.452	0.175	0.372
SketchHash	0.248	0.301	0.302	0.327	-	-	0.256	0.431	0.385	0.059	-	-
AdaptHash	0.116	0.138	0.216	0.297	0.305	0.293	0.114	0.254	0.185	0.093	0.166	0.164
OSH	0.123	0.126	0.129	0.131	0.127	0.125	0.120	0.123	0.137	0.117	0.083	0.038
BSODH	0.564	0.604	0.689	0.656	0.709	0.711	0.305	0.582	0.691	0.697	**0.690**	**0.602**
DSBOH	0.556	0.669	0.703	0.696	0.720	0.727	**0.411**	**0.730**	**0.737**	0.655	0.552	0.371
Ours	**0.589**	**0.673**	**0.722**	**0.730**	**0.735**	**0.742**	0.366	0.662	0.733	**0.721**	0.675	0.541

The best results are displayed in bold.

Table 4 reveals the values of mAP and Precision@H2 on MNIST. Results show that (1) mAP: In the case of 8-bit, 16-bit, 48-bit, 64-bit, and 128-bit, our proposed LSOH is 2.6%, 0.7%, 0.7%, 0.7%, and 0.8%, respectively, higher than the suboptimal algorithm orderly. It is the second-best algorithm when generating 32-bit hash codes. The advantage of LSOH is verified. (2) Precision@H2: values of Precision@H2 on LSOH are the highest under 8-bit and it ranks second in other code bits. As the bit length grows, the performance of LSOH is worse than that of BSODH under 48-bit, 64-bit, and 128-bit, but better than DSBOH.

Table 4. mAP and Precision@H2 comparisons on MNIST.

	mAP						Precision@H2					
Methods	**8-bit**	**16-bit**	**32-bit**	**48-bit**	**64-bit**	**128-bit**	**8-bit**	**16-bit**	**32-bit**	**48-bit**	**64-bit**	**128-bit**
OKH	0.100	0.155	0.224	0.273	0.301	0.404	0.100	0.220	0.457	0.724	0.522	0.124
SketchHash	0.257	0.312	0.348	0.369	-	-	0.261	0.596	0.691	0.251	-	-
AdaptHash	0.138	0.207	0.319	0.318	0.292	0.208	0.153	0.442	0.535	0.335	0.163	0.168
OSH	0.130	0.144	0.130	0.148	0.146	0.143	0.131	0.146	0.192	0.134	0.109	0.019
BSODH	0.593	0.700	0.747	0.743	0.766	0.760	0.308	0.709	0.826	**0.804**	**0.814**	**0.643**
DSBOH	0.596	0.721	**0.759**	0.751	0.781	0.781	0.403	**0.803**	**0.849**	0.788	0.651	0.415
Ours	**0.622**	**0.728**	0.756	**0.758**	**0.788**	**0.789**	**0.418**	0.757	0.846	0.796	0.761	0.487

The best results are displayed in bold.

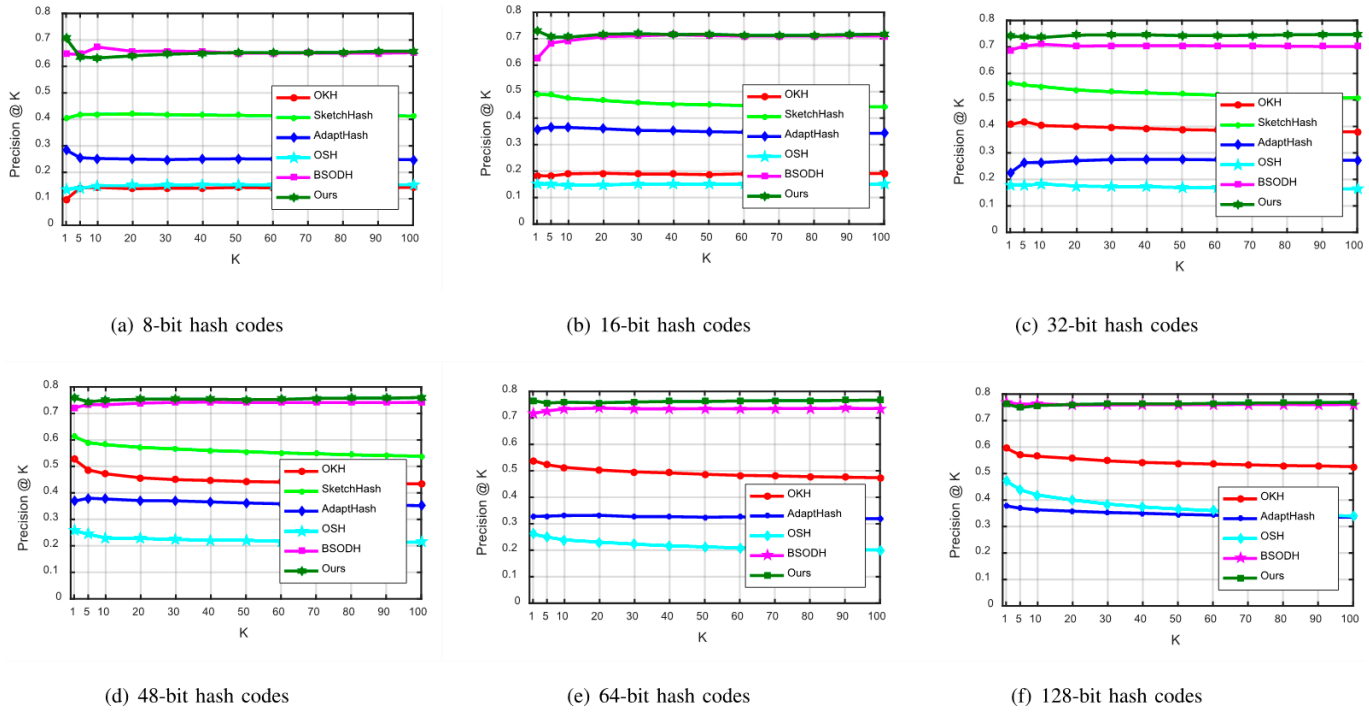
The outcomes of mAP and Precision@H2 on the Places205 dataset are expressed in Table 5. (1) mAP: our proposed LSOH is the best algorithm with 3.3% and 1.1% higher than the suboptimal algorithm under 16-bit and 32-bit, respectively. In other cases, the results of LSOH are not optimal. Due to the huge amount of data in Places205, other comparison algorithms have not always performed optimally. In contrast, the LSOH algorithm has better stability and relatively higher retrieval accuracy. (2) Precision@H2: our proposed LSOH has optimal values under 16-bit and 48-bit, with 1.1% and 0.8%, respectively, better than the second-best algorithm, and it ranks second in other code bits. In conclusion, our proposed LSOH algorithm performs well and has high retrieval accuracy on Precision@H2.

Table 5. mAP@1000 and Precision@H2 comparisons on Places205.

	mAP						Precision@H2					
Methods	**8-bit**	**16-bit**	**32-bit**	**48-bit**	**64-bit**	**128-bit**	**8-bit**	**16-bit**	**32-bit**	**48-bit**	**64-bit**	**128-bit**
OKH	0.018	0.033	0.122	0.048	0.114	0.258	0.007	0.010	0.026	0.017	**0.217**	0.075
SketchHash	**0.052**	0.120	0.202	0.242	-	-	**0.017**	0.066	0.220	0.176	-	-
AdaptHash	0.028	0.097	0.195	0.223	0.222	0.229	0.009	0.051	0.012	0.185	0.021	0.022
OSH	0.018	0.021	0.022	0.032	0.043	0.164	0.007	0.009	0.012	0.023	0.030	0.059
BSODH	0.035	0.174	0.250	0.273	0.308	0.337	0.009	0.101	0.241	0.246	0.212	**0.101**
DSBOH	0.046	0.154	0.240	**0.286**	**0.313**	**0.347**	0.011	0.089	**0.264**	0.175	0.119	0.037
Ours	0.043	**0.187**	**0.251**	0.282	0.296	0.323	0.013	**0.110**	0.244	**0.254**	0.213	0.098

The best results are displayed in bold.

For further testing of our proposed LSOH, Precision@K curves in the case of 8-bit, 16-bit, 32-bit, 48-bit, 64-bit, and 128-bit are drawn on CIFAR-10 and MNIST, as displayed in Figures 5 and 6. Comparative experiments of these metrics on the Places205 dataset are not conducted, due to its large memory requirements.

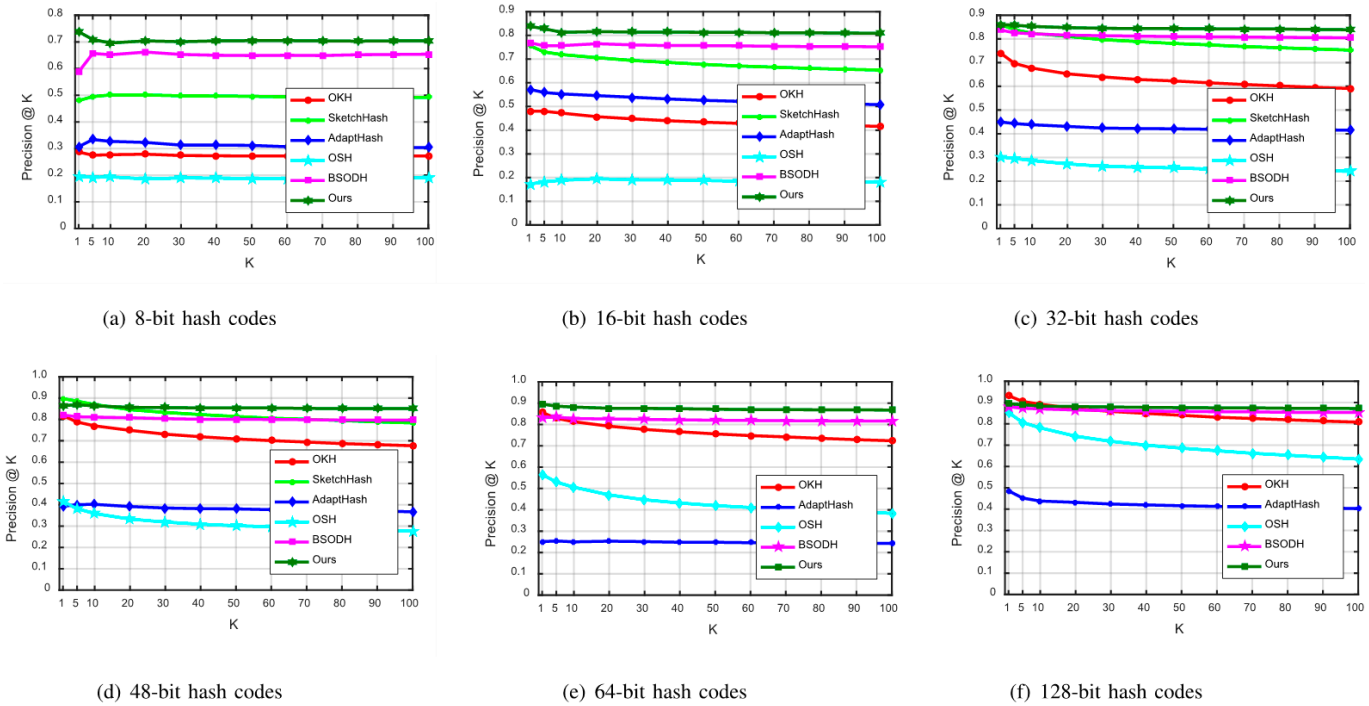


(a) 8-bit hash codes (b) 16-bit hash codes (c) 32-bit hash codes

(d) 48-bit hash codes (e) 64-bit hash codes (f) 128-bit hash codes

Figure 5. Comparisons of Precision@K curves on CIFAR-10.

From Figure 5, we can see the Precision@K curves on the CIFAR-10 dataset. It is obvious that the Precision@K curve of our proposed LSOH is higher than other comparison curves in the case of 8-bit, 16-bit, 32-bit, 48-bit, 64-bit, and 128-bit. Thus, the performance of 32-bit and 64-bit hash codes is particularly outstanding. As shown in Figure 6, LSOH continuously reveals a higher Precision@K curve, compared with other algorithms on the MNIST dataset. Only when generating 8-bit hash codes, does our proposed LSOH have a temporary fluctuation on the CIFAR-10 dataset. As hash bits increase, the retrieval accuracy goes up slightly, which shows the robustness and superiority of the LSOH algorithm.



(a) 8-bit hash codes (b) 16-bit hash codes (c) 32-bit hash codes

(d) 48-bit hash codes (e) 64-bit hash codes (f) 128-bit hash codes

Figure 6. Comparisons of Precision@K curves on MNIST.

Figure 7 shows the precision–recall (PR) curves under 32-bit. By calculating the area under curve (AUC) of the PR curves, we obtain the values of 95.85% and 91.77% in turn, which demonstrates that the proposed LSOH has a doubly high ratio of precision and recall.

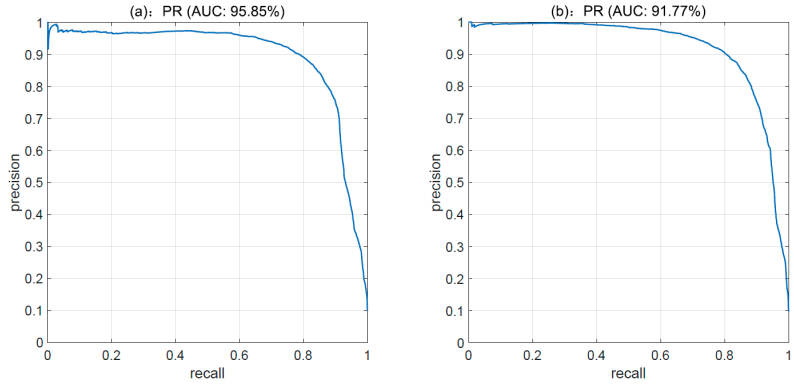
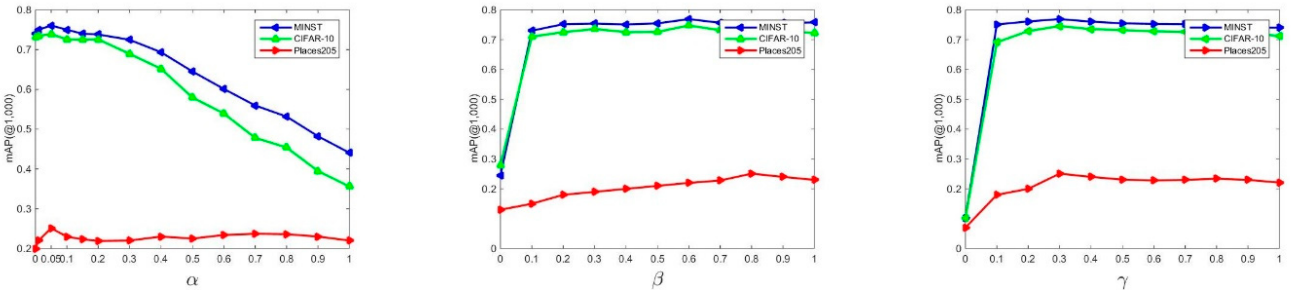


Figure 7. PR curve under 32-bit hash codes. (**a**) PR curve on CIFAR-10 (**b**) PR curve on MNIST.

4.4. Parameter Sensitivity

In this subsection, we conduct the ablation studies on the hyper-parameters of α^t, β^t, and γ^t, as defined in Equation (12). Without loss of generality, we conduct experiments with varying values of these hyper-parameters concerning mAP(mAP@1000) in the case of 32-bit, in Figure 8. (Detailed values used in this paper are outlined in Table 2.) Similar experimental results can be observed in other hashing bits.



(**a**) α^t (**b**) β^t (**c**) γ^t

Figure 8. Comparisons of mAP(@1000) performances concerning varying values of α^t, β^t, γ^t when the hashing bit is 32.

As shown in Equation (12), α^t is used to reflect the importance of anchor graph hashing. Figure 8a plots the influence of different values of α^t on the performance. Generally speaking, when $\alpha^t = 0.05$ on CIFAR-10 and MNIST, LSOH obtains the best mAP (0.739 on CIFAR-10 and 0.760 on MNIST). When $\alpha^t = 0.05$ on Places205, LSOH obtains the best mAP@1000, with 0.251. Moreover, when $\alpha^t = 0$, LSOH suffers a performance degradation, as can be seen in Figure 8a. More specifically, in this case, the mAP(mAP@1000) scores are 0.740, 0.731, and 0.200 on MNIST, CIFAR-10, and Places205, respectively. To analyze, when $\alpha^t = 0$, LSOH is similar to BSODH. In the experiments, we empirically set the values of α^t as 0.05 on all three datasets.

As shown in Equation (12), β^t is used to reflect the importance of the quantized loss function. From Figure 8b, we can observe that when $\beta^t = 0.6$ on CIFAR-10 and MNIST, LSOH obtains the best mAP (0.768 on MNIST and 0.747 on CIFAR-10). When $\beta^t = 0.8$ on Places205, LSOH obtains the best mAP@1000, with 0.251. Moreover, when $\beta^t = 0$, LSOH suffers great performance degradation, as can be seen in Figure 8b (0.278 on CIFAR-10, 0.244 on MNIST, and 0.13 on Places205). We can observe from Figure 8b that properly applying the quantized loss term in Equation (11) can significantly boost the performance of the three datasets. In the experiments, we empirically set the values of β^t as 0.6 on CIFAR-10 and MNIST, and 0.8 on Places205.

From Figure 8c, we can observe that when $\gamma^t = 0.3$ on CIFAR-10 and MNIST, LSOH obtains the best mAP (0.768 on MNIST and 0.745 on CIFAR-10). When $\gamma^t = 0.3$ on Places205, LSOH obtains the best mAP@1000, with 0.251. Moreover, when $\gamma^t = 0$, LSOH suffers great performance degradation, as can be seen in Figure 8c. Thus, it is necessary to use a penalty term properly to prevent the model from overfitting. In the experiments, we empirically set the values of γ^t as 0.3 on the three datasets.

4.5. Limitations and Potential Improvements

By comparing the weights of each module in Equation (12), it can be seen that the global-balanced similarity plays an important role in training hash codes. However, some operations on a matrix need to be processed, due to the introduction of anchor hashing, which leads to the training time of LSOH being slightly slower than that of the BSODH algorithm. For example, LSOH takes several seconds longer than BSODH when generating a 32-bit hash code, but it is shorter than OSH. In addition, the inverse of the matrix is required when calculating W^t, and its time complexity is $O(d^3)$. In other words, it is time-consuming when the dimension of the retrieval image is too large. Therefore, employing a more effective and efficient method to perform the matrix operation is desirable and worthwhile.

5. Conclusions

In this paper, a novel hashing algorithm preserving both the local and global dual semantics for image retrieval, i.e. LSOH, was proposed. By extracting the local manifold

structure for data coming at the same time, and constructing a global-balanced similarity matrix from data at a different time, we obtain a relatively comprehensive hash constraint, which avoids the problem of over-reliance on labels and imbalanced data updates. Then, an alternative-iteration algorithm is used to solve the discrete binary optimization. Extensive experiments on the benchmark datasets verify that LSOH has significant advantages, compared with other advanced algorithms. However, similar to other state-of-the-art online-hashing algorithms, LSOH decreases the retrieval accuracy with the hash-bits increase. Recently, cross-modal retrieval has had more application requirements, and our method of mining the local structural features of the retrieval data and finding similarity measures of the global data is also worthy of reference and of application in cross-modal retrieval. Given the strong capability for feature representation, the research on online hashing with deep learning networks is also a valuable topic for the future.

Author Contributions: Conceptualization, X.C. and Y.L.; methodology, X.C.; software, X.C. and C.C.; validation, C.C.; formal analysis, X.C.; investigation, X.C. and C.C.; resources, X.C. and Y.L.; data curation, C.C.; writing—original draft preparation, C.C. and X.C.; writing—review and editing, X.C.; visualization, X.C.; supervision, X.C.; project administration, Y.L.; funding acquisition, Y.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by the National Natural Science Foundation of China under grant 62261009, Ministry of Education Key Laboratory of Cognitive Radio and Information Processing (CRKL200106), and Innovation Project of Guangxi Graduate Education (YCBZ2022106).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Masood, F.; Masood, J.; Zahir, H.; Driss, K.; Mehmood, N.; Farooq, H.S. Novel approach to evaluate classification algorithms and feature selection filter algorithms using medical data. *J. Comput. Cogn. Eng.* **2022**, *2*, 1–11.
2. Zhang, H.; Xu, C.; Shi, C.; Bi, H.; Li, Y.; Mian, S. HSCA-Net: A Hybrid Spatial-Channel Attention Network in Multiscale Feature Pyramid for Document Layout Analysis. *J. Artif. Intell. Technol.* **2022**, *3*, 10–17. [CrossRef]
3. Zheng, M.; Zhi, K.; Zeng, J.; Tian, C.; You, L. A hybrid CNN for image denoising. *J. Artif. Intell. Technol.* **2022**, *2*, 93–99. [CrossRef]
4. Latif, A.; Rasheed, A.; Sajid, U.; Ahmed, J.; Ali, N.; Ratyal, N.I.; Zafar, B.; Dar, S.H.; Sajid, M.; Khalil, T. Content-Based Image Retrieval and Feature Extraction: A Comprehensive Review. *Math. Probl. Eng.* **2019**, *2019*, 9658350. [CrossRef]
5. Bani, N.T.; Fekri-Ershad, S. Content-based image retrieval based on combination of texture and colour information extracted in spatial and frequency domains. *Electron. Libr.* **2019**, *37*, 650–666. [CrossRef]
6. Indyk, P.; Motwani, R. Approximate nearest neighbors: Towards removing the curse of dimensionality. In Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing, Dallas, TX, USA, 24–26 May 1998; pp. 604–613.
7. Gionis, A.; Indyk, P.; Motwani, R. Similarity search in high dimensions via hashing. In Proceedings of the 25th International Conference on Very Large Data Bases, San Francisco, CA, USA, 7–10 September 1999; Volume 99, pp. 518–529.
8. Martínez, S.; Gérard, S.; Cabot, J. Efficient model similarity estimation with robust hashing. *Softw. Syst. Model.* **2022**, *21*, 337–361. [CrossRef]
9. Weng, Z.; Zhu, Y. Online hashing with bit selection for image retrieval. *IEEE Trans. Multimed.* **2020**, *23*, 1868–1881. [CrossRef]
10. Lai, Z.; Chen, Y.; Wu, J.; Wong, W.K.; Shen, F. Jointly sparse hashing for image retrieval. *IEEE Trans. Image Process.* **2018**, *27*, 6147–6158. [CrossRef]
11. Liu, X.; Yi, J.; Cheung, Y.-M.; Xu, X.; Cui, Z. Omgh: Online manifold-guided hashing for flexible cross-modal retrieval. *IEEE Trans. Multimed.* **2022**, *8*, 99. [CrossRef]
12. Charikar, M.S. Similarity estimation techniques from rounding algorithms. In Proceedings of the Thiry-Fourth Annual ACM Symposium on Theory of Computing, Montreal, QC, USA, 21–28 May 2002; pp. 380–388.
13. Ding, K.; Huo, C.; Fan, B.; Xiang, S.; Pan, C. In defense of locality-sensitive hashing. *IEEE Trans. Neural Netw. Learn. Syst.* **2016**, *29*, 87–103. [CrossRef]
14. Datar, M.; Immorlica, N.; Indyk, P.; Mirrokni, V.S. Locality-sensitive hashing scheme based on p-stable distributions. In Proceedings of the Twentieth Annual Symposium on Computational Geometry, Brooklyn, NY, USA, 8–11 June 2004; pp. 253–262.
15. Chum, O.; Philbin, J.; Zisserman, A. Near duplicate image detection: Min-hash and tf-idf weighting. *Bmvc* **2008**, *810*, 812–815.

16. Kulis, B.; Grauman, K. Kernelized locality-sensitive hashing for scalable image search. In Proceedings of the 2009 IEEE 12th International Conference on Computer Vision, Kyoto, Japan, 29 September–2 October 2009; pp. 2130–2137.
17. Weiss, Y.; Torralba, A.; Fergus, R. Spectral hashing. In Proceedings of the 21st International Conference on Neural Information Processing Systems, Ser. NIPS'08, Washington, DC, USA, 8 December 2008; Curran Associates Inc.: Red Hook, NY, USA, 2008; pp. 1753–1760.
18. Liu, W.; Wang, J.; Kumar, S.; Chang, S.-F. Hashing with graphs. In Proceedings of the 28th International Conference on International Conference on Machine Learning, Ser. ICML'11, Washington, DC, USA, 28 June–2 July 2011; Omnipress: Madison, WI, USA, 2011; pp. 1–8.
19. Gong, Y.; Lazebnik, S.; Gordo, A.; Perronnin, F. Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval. *IEEE Trans. Pattern Anal. Mach. Intell.* **2012**, *35*, 2916–2929. [CrossRef] [PubMed]
20. Liu, W.; Wang, J.; Ji, R.; Jiang, Y.-G.; Chang, S.-F. Supervised hashing with kernels. In Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, 16–21 June 2012; pp. 2074–2081.
21. Zhang, P.; Zhang, W.; Li, W.-J.; Guo, M. Supervised hashing with latent factor models. In Proceedings of the 37th international ACM SIGIR Conference on Research & Development in Information Retrieval, Gold Coast, QLD, Australia, 6–11 July 2014; pp. 173–182.
22. Xia, R.; Pan, Y.; Lai, H.; Liu, C.; Yan, S. Supervised hashing for image retrieval via image representation learning. In Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, Québec City, QC, Canada, 27–31 July 2014.
23. Lin, G.S.; Shen, C.H.; Shi, Q.F.; Van den Hengel, A.; Suter, D. Fast supervised hashing with decision trees for high-dimensional data. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 1963–1970.
24. Huang, L.-K.; Yang, Q.; Zheng, W.-S. Online hashing. In Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), Beijing, China, 3–9 August 2013; pp. 1422–1428.
25. Huang, L.-K.; Yang, Q.; Zheng, W.-S. Online hashing. *IEEE Trans. Neural Netw. Learn. Syst.* **2018**, *29*, 2309–2322. [CrossRef] [PubMed]
26. Wang, L.; Wang, Q.; Wang, D.; Wan, B.; Shang, B. Online matrix factorization hashing for large-scale image retrieval. In Proceedings of the Big Data: 6th CCF Conference, Big Data 2018, Xi'an, China, 11–13 October 2018; Xu, Z., Gao, X., Miao, Q., Zhang, Y., Bu, J., Eds.; Springer: Singapore, 2018; pp. 124–134.
27. Lin, M.B.; Ji, R.R.; Chen, S.; Sun, X.S.; Lin, C.-W. Similarity-preserving linkage hashing for online image retrieval. *IEEE Trans. Image Process.* **2020**, *29*, 5289–5300. [CrossRef] [PubMed]
28. Yi, J.; Liu, X.; Cheung, Y.-M.; Xu, X.; Fan, W.; He, Y. Efficient online label consistent hashing for large-scale cross-modal retrieval. In Proceedings of the 2021 IEEE International Conference on Multimedia and Expo (ICME), Shenzhen, China, 5–9 July 2021; pp. 1–6.
29. Zhan, Y.-W.; Luo, X.; Sun, Y.; Wang, Y.; Chen, Z.-D.; Xu, X.-S. Weakly-supervised online hashing. In Proceedings of the 2021 IEEE International Conference on Multimedia and Expo (ICME), Shenzhen, China, 5–9 July 2021; pp. 11–16.
30. Chen, X.; Yang, H.; Zhao, S.; King, I.; Lyu, M.R. Making online sketching hashing even faster. *IEEE Trans. Knowl. Data Eng.* **2021**, *33*, 1089–1101. [CrossRef]
31. Lin, H.; Meng, M.; Wu, J. Online robust specific and consistent hashing. In Proceedings of the 2022 IEEE International Conference on Multimedia and Expo (ICME), Taipei, Taiwan, 18–22 July 2022; pp. 1–6.
32. Cakir, F.; Sclaroff, S. Adaptive hashing for fast similarity search. In Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015; pp. 1044–1052.
33. Cakir, F.; Bargal, S.A.; Sclaroff, S. Online supervised hashing. *Comput. Vis. Image Underst.* **2017**, *156*, 162–173. [CrossRef]
34. Cakir, F.; He, K.; Bargal, S.A.; Sclaroff, S. Mihash: Online hashing with mutual information. In Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017.
35. Lin, M.B.; Ji, R.R.; Liu, H.; Sun, X.S.; Wu, Y.J.; Wu, Y.S. Towards optimal discrete online hashing with balanced similarity. In Proceedings of the 33rd AAAI Conference on Artificial Intelligence (AAAI 2019), Honolulu, HI, USA, 28–29 January 2019; pp. 8722–8729.
36. Lin, M.B.; Ji, R.R.; Liu, H.; Wu, Y.J. Supervised online hashing via hadamard codebook learning. In Proceedings of the 2018 ACM Multimedia Conference, Seoul, Republic of Korea, 22–26 October 2018.
37. Lin, M.B.; Ji, R.R.; Liu, H.; Sun, X.S.; Chen, S.; Tian, Q. Hadamard matrix guided online hashing. *Int. J. Comput. Vis.* **2020**, *128*, 2279–2306. [CrossRef]
38. Leng, C.; Wu, J.; Cheng, J.; Bai, X.; Lu, H. Online sketching hashing. In Proceedings of the IEEE Conference on Computer Vision & Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 2503–2511.
39. Chen, X.; King, I.; Lyu, M.R. Frosh: Faster online sketching hashing. In Proceedings of the Thirty-Third Conference on Uncertainty in Artificial Intelligence, UAI 2017, Sydney, Australia, 11–15 August 2017; pp. 1–10.
40. Jin, L.; Shu, X.; Li, K.; Li, Z.; Qi, G.-J.; Tang, J. Deep ordinal hashing with spatial attention. *IEEE Trans. Image Process.* **2019**, *28*, 2173–2186. [CrossRef]
41. He, S.; Ye, G.; Hu, M.; Yang, Y.; Shen, F.; Shen, H.T.; Li, X. Learning binary codes with local and inner data structure. *Neurocomputing* **2018**, *282*, 32–41. [CrossRef]

42. Ji, R.; Liu, H.; Cao, L.; Liu, D.; Wu, Y.; Huang, F. Toward optimal manifold hashing via discrete locally linear embedding. *IEEE Trans. Image Process.* **2017**, *26*, 5411–5420. [CrossRef]
43. Zhang, J.; Peng, Y. Ssdh: Semi-supervised deep hashing for large scale image retrieval. *IEEE Trans. Circuits Syst. Video Technol.* **2017**, *29*, 212–225. [CrossRef]
44. Wang, G.; Hu, Q.; Cheng, J.; Hou, Z. Semi-supervised generative adversarial hashing for image retrieval. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 17–24 May 2018; pp. 469–485.
45. Lai, H.; Yan, P.; Shu, X.; Wei, Y.; Yan, S. Instance-aware hashing for multi-label image retrieval. *IEEE Trans. Image Process.* **2016**, *25*, 2469–2479. [CrossRef] [PubMed]
46. Ma, Q.; Bai, C.; Zhang, J.; Liu, Z.; Chen, S. Supervised learning based discrete hashing for image retrieval. *Pattern Recognit.* **2019**, *92*, 156–164. [CrossRef]
47. Chen, J.; Li, Y.; Lu, H. Online self-organizing hashing. In Proceedings of the 2016 IEEE International Conference on Multimedia and Expo (ICME), Seattle, WA, USA, 11–15 July 2016; pp. 1–6.
48. Gong, Y.; Lazebnik, S. Iterative quantization: A procrustean approach to learning binary codes. In Proceedings of the CVPR 2011, Springs, CO, USA, 20–25 June 2011; pp. 817–824.
49. Kang, W.-C.; Li, W.-J.; Zhou, Z.-H. Column sampling based discrete supervised hashing. In Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, Ser. AAAI'16, Phoenix, AZ, USA, 12–17 February 2016; pp. 1230–1236.
50. Shen, F.; Shen, C.; Liu, W.; Shen, H.T. Supervised discrete hashing. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015; pp. 37–45.
51. Torralba, A.; Fergus, R.; Freeman, W.T. 80 million tiny images: A large data set for nonparametric object and scene recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **2008**, *30*, 1958–1970. [CrossRef] [PubMed]
52. Chen, C.; Wang, X.Q.; Chen, X.; Lan, R.S.; Liu, Z.B.; Luo, X.N. Discriminative Similarity-Balanced Online Hashing for Supervised Image Retrieval. *Sci. Program.* **2022**, *2022*, 2809222. [CrossRef]