Pattern Recognition 122 (2022) 108262

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/patcog

Discrete online cross-modal hashing

Yu-Wei Zhan, Yongxin Wang, Yu Sun, Xiao-Ming Wu, Xin Luo*, Xin-Shun Xu

School of Software, Shandong University, Jinan, China

ARTICLE INFO

Article history:
Received 19 June 2021
Revised 12 August 2021
Accepted 18 August 2021
Available online 19 August 2021

Keywords:
Cross-modal retrieval
Discrete optimization
Online hashing
Learning to hash

ABSTRACT

With the prevalence of multimedia content on the Web which usually continuously comes in a stream fashion, online cross-modal hashing methods have attracted extensive interest in recent years. However, most online hashing methods adopt a relaxation strategy or real-valued auxiliary variable strategy to avoid complex optimization of hash codes, leading to large quantization errors. In this paper, based on Discrete Latent Factor model-based cross-modal Hashing (DLFH), we propose a novel cross-modal online hashing method, i.e., Discrete Online Cross-modal Hashing (DOCH). To generate uniform high-quality hash codes of different modal, DOCH not only directly exploits the similarity between newly coming data and old existing data in the Hamming space, but also utilizes the fine-grained semantic information by label embedding. Moreover, DOCH can discretely learn hash codes by an efficient optimization algorithm. Extensive experiments conducted on two real-world datasets demonstrate the superiority of DOCH.

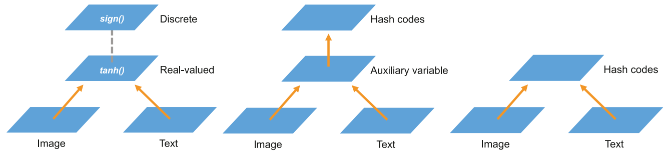
1. Introduction

Due to the explosive growth in multimedia data from a great variety of search engines and social media, cross-modal retrieval has become a compelling topic in recent years [1–3,47]. Cross-modal retrieval aims to search semantically similar instances from one modality (e.g., image) by using a query from another modality (e.g., text) [4–6,45]. With high complexity in both time and storage, the traditional nearest neighbor search seems to be suboptimal and unpractical. As an alternative, approximate nearest neighbors (ANN), especially learning to hash [7,8], has attracted increasing attention. Cross-modal hashing methods transform high-dimensional multimedia data into compact binary codes while generating similar binary codes for similar data items [9–11]. With low storage cost and fast query speed, hashing-based methods can efficiently calculate the hamming distance by using XOR operation and dramatically reduce storage cost by using binary hash codes to represent data in the Hamming space [12–15].

In many scenarios, multimedia data from search engines and social networks usually continuously comes in a stream fashion. However, most existing cross-modal hashing methods are batch-based [16–20], which retrain the hash functions with all accumulated data when new data arrives. In addition, increasingly large datasets make it impractical for all training data to be loaded into memory at once and make the computational cost unacceptable. Therefore, several online cross-modal hashing methods [21] are proposed to support the efficient search over streaming data, which update hash functions based on the newly coming data while preserving the effectiveness of binary codes for the past streaming data. As shown in Fig. 1, the learning strategies of binary codes can be roughly divided into three categories: discrete strategy, relaxation-based strategy, and auxiliary variable strategy. Without continuous relaxation, discrete methods [22] try to directly learn hash codes with binary constraints. Relaxation-based methods [23] approximate the discrete *sign*$(\cdot)$ function using the continuous *tanh*$(\cdot)$ function to avoid the NP-hard problem. To avoid complex optimization of hash codes, auxiliary variable strategy is adopted in some papers [24], which replace the hash codes matrix, i.e., **B**, with a real-valued auxiliary variable. However, relaxation strategy or auxiliary variable strategy mostly obtain the relaxed continuous solutions first and then quantize them into the binary hash codes. These two strategies may lead to large quantization errors between discrete values and real values, resulting in information loss and performance degradation. From another point of view, one of the primary goals of supervising hashing methods is to learn the hash code in the Hamming space to maintain the similarity among data in the original space. However, most online hashing methods relax the hash code from the Hamming space to the real-valued space to avoid complex optimization. They preserve the similarity between newly coming data and previously accumulated data in the latent space (real-valued space), which is insuffi-

[*] Corresponding author.
E-mail address: luoxin.lxin@gmail.com (X. Luo).

https://doi.org/10.1016/j.patcog.2021.108262



(a) Relaxation-based (b) Auxiliary variable (c) Discrete

Fig. 1. The learning strategies of binary codes.

ciently direct and effective compared with the methods of measuring similarity directly in Hamming space.

To overcome the issues mentioned above, we propose a novel online supervised hashing method named Discrete Online Cross-modal Hashing (DOCH), which incorporates similarity preserving and label embedding into one unified framework. In our method, discrete strategy in Fig. 1-(c) is used to learn hash codes with binary constraints, which avoids quantization error and get better performance. To update hash functions that can not only fit the new data well but also ensure the representation ability of the previous unchanged hash codes, DOCH directly preserves the similarity between newly coming data and existing old data in the Hamming space. Besides, the label information is embedded into the learning of hash codes to make up for the lack of binary coarse-grained similarity. Moreover, we adopt a discrete online optimization without any relaxation to directly learn binary codes. The main contributions of DOCH are summarized as follows:

- A novel supervised online cross-modal hashing method is proposed. By directly preserving the similarity between newly coming data and old existing data in the Hamming space and embedding supervised label information into to-be-learned hash codes, DOCH can learn more accurate unified hash codes for multiple modal and easily extend to more modalities.
- Both time complexity and space complexity are significantly reduced by randomly selecting some instances as anchors. Moreover, the performance of DOCH can be maintained well, even if the number of selected instances is small.
- An efficient discrete online optimization algorithm is proposed. Based on the algorithm, DOCH discretely learns hash codes with the binary constraints maintained, which can avoid large quantization errors.
- Extensive experiments are conducted over two widely-used benchmark datasets, which demonstrate the superiority of DOCH compared with several state-of-the-art hashing baselines. As the online cross-modal hashing field is challenging and less studied, we have already released the code and hope that it could facilitate other researchers and the community[2].

2. Related work

2.1. Online hashing

In many real-world scenarios, multimedia data is usually generated periodically or collected in a stream fashion [25–27]. Additionally, the computational cost of batch-based hashing methods may be unaffordable and unbearable with larger and larger datasets. Therefore, online hashing has become one of the hottest topics in recent years. To learn hash functions in an online manner, Online Hashing (OH) [28] learns structured prediction with a prediction loss function. Balanced Similarity for Online Discrete Hashing (BSODH) [29] employs an asymmetric graph regularization to preserve the similarity between the streaming data and the existing dataset and learns the hash codes discretely. Hadamard Matrix Guided Online Hashing (HMOH) [30] first introduces Hadamard matrix and uses each column of this matrix as the target code for each class label to guide the learning.

Although impressive performance has been achieved, the aforementioned online hashing methods only are designed for dealing with unimodal data. There are also several online hashing works are proposed for multi-modal data, which handle multi-view retrieval by fusing multiple modalities. Representative methods include, but not limited to, Dynamic Multi-View Hashing (DMVH) [31] and Flexible Online Multi-modal Hashing (FOMH) [32]. Nevertheless, these multi-modal methods still cannot carry out the cross-modal search.

Only a few online cross-modal hashing methods are proposed for heterogeneous data with multiple modalities at present. According to whether supervised information is utilized, existing online cross-modal hashing methods can be roughly divided into two categories: the unsupervised and supervised ones. Unsupervised methods exploit the inherent correction among data without any supervised information. For example, Online Collective Matrix Factorization Hashing (OCMFH) [24], an online version of Collective Matrix Factorization Hashing (CMFH) [33], transforms multi-modal data into latent space by collective matrix factorization. For supervised cross-modal online hashing [34], label information can be further utilized to guide the learning of hash codes and boost performance. Online Latent Semantic Hashing (OLSH) [22] maps discrete labels into a continuous latent semantic concept space and uses it to guide hash-code learning. Label Embedding Online Hashing (LEMON) [35] updates hash functions based on the correlation between newly coming data and existing data. By classifying the common semantic vector to labels with a multi-class classification, Online Label Consistent Hashing (OLCH) [36] learns discriminative hash code, whereby the intra-modal semantic similarity and inter-modal semantic similarity are preserved. However, to the best of our knowledge, only one of the existing online cross-modal hashing methods [22] can directly learn the binary codes in hamming space without any relaxation strategy, but its optimization is time-consuming and its performance is not satisfying.

2.2. Discrete latent factor model-based cross-modal hashing

Our method is an online improved version of Discrete Latent Factor Model-Based Cross-Modal Hashing (DLFH) [37]. We briefly introduce DLFH in this section.

DLFH utilizes a discrete latent factor model to preserve the similarity information and directly learns the different binary hash codes for different modalities without continuous relaxation. The problem is defined as,

$$\max_{\mathbf{U},\mathbf{V}} L = \log p(\mathbf{S} \mid \mathbf{U},\mathbf{V}) = \sum_{i,j=1}^{n} \left[S_{ij}\Theta_{ij} - \log\left(1+e^{\Theta_{ij}}\right)\right] \qquad (1)$$

$$\text{s.t.} \quad \mathbf{U},\mathbf{V} \in \{-1,1\}^{n\times r},$$

where $\mathbf{U},\mathbf{V} \in \{-1,1\}^{n\times r}$ denote the binary codes for image modality and text modality, respectively, r is the hash code length, n is the number of training data, $\mathbf{S} \in \{0,1\}^{n\times n}$ is a cross-modal supervised similarity matrix, and $\Theta_{ij} = \frac{\lambda}{c}\mathbf{U}_{i*}\mathbf{V}_{j*}^{T}$. To reduce computational cost, DLFH randomly samples m columns (rows) from $\mathbf{S}$ to calculate the loss.

Although we design our approach based on the DLFH, it is very different. 1) DLFH generates different hash codes for heterogeneous modalities, which fails to better incorporate multiple modalities and extend it to the more modalities case. 2) DLFH utilizes a binary coarse-grained similarity matrix as supervised information to guide the learning of the hash codes, resulting in the loss of partial semantic information. 3) DLFH is inefficient for steaming data, which needs to accumulate all data and retrains the hash functions when new data arrives. As a result of the above points, DLFH could

[2] https://github.com/yw-zhan/DOCH

not directly deal with streaming data, so we innovatively improve DLFH and propose a new online cross-modal hashing method: 1) Our method transforms heterogeneous data into unified hash codes while preserving the similarity between newly coming data and existing data in the hamming space. 2) To make up for the shortcomings of binary similarity, we use more fine-grained supervised information, e.g., the class label, to help the model generate more accurate hash codes. 3) DOCH discretely learns binary codes of newly coming data with those of the old database unchanged, making it efficient for streaming data and scalable to large-scale datasets.

3. Our method

In this section, we first give the notions and problem formulation; then, show the details of our proposed method including the framework, optimization scheme, complexity and convergence analysis, and its extensions to out-of-sample data and more modalities. DOCH captures correlations between the newly coming data and existing data in the Hamming space to learn more quality hash codes of newly coming data. To make our model adaptable for large-scale datasets, we also randomly select some instances from old existing data as anchors to avoid high time complexity and space complexity. Besides, we embed supervised label information into to-be-learned hash codes so that binary codes are easy to be classified. The following is a detailed description of each module.

3.1. Notations and problem definitions

In this paper, we utilize boldface uppercase letters and boldface lowercase letters, e.g., $\mathbf{Y}$ and $\mathbf{y}$ to denote matrices and vectors. $\mathbf{Y}_{i*}$ indicates the i-th row of matrix $\mathbf{Y}$ and $\mathbf{Y}_{*j}$ indicates the j-th column of matrix $\mathbf{Y}$. The transpose of $\mathbf{Y}$ is denoted as $\mathbf{Y}^\top$ and the inverse of $\mathbf{Y}$ is denoted as $\mathbf{Y}^{-1}$. $\| \mathbf{Y} \|_F^2$ is the Frobenius-norm of $\mathbf{Y}$ and $\| \mathbf{Y} \|_F^2 = Tr(\mathbf{Y}^\top \mathbf{Y})$, where $Tr(\cdot)$ is the trace of a square matrix.

Suppose the training dataset composed of M modalities comes at a streaming manner. At the t-th round, n_t instances with class labels $\vec{\mathbf{L}}^{(t)} \in \{0,1\}^{n_t \times c}$ arrive, where c is the number of classes. The m-th modality feature of new data chunk is denoted as $\vec{\mathbf{X}}_m^{(t)} \in \mathbb{R}^{n_t \times d_m}$, where $m \in \{1,\ldots,M\}$ and d_m is the dimensionality of the m-th modality feature. Correspondingly, $\tilde{\mathbf{X}}_m^{(t)} \in \mathbb{R}^{N_{t-1} \times d_m}$ is the m-th modality feature of the existing data accumulated before round t and $\tilde{\mathbf{L}}^{(t)} \in \{0,1\}^{N_{t-1} \times c}$ is the label matrix, where N_{t-1} is the size of existing data and $N_{t-1} = \sum_{i=1}^{t-1} n_i$. In addition, $\mathbf{S}$ denotes the instance pairwise similarity, where $\mathbf{S}_{ij} = 1$ means the i-th instance and the j-th instance are semantically similar, e.g., they share at least one common class label, and $\mathbf{S}_{ij} = 0$ otherwise.

In this paper, when a new data chunk comes at round t, we aim to: 1) update hash functions for M modalities to fit both the newly coming data and existing data. 2) generate r-bit binary hash codes for newly coming data, e.g., $\vec{\mathbf{B}}^{(t)} \in \{-1,1\}^{n_t \times r}$, with the hash codes of existing data $\tilde{\mathbf{B}}^{(t)}$ unchanged.

We summarize the notations used in this paper in Table 1, which will be detailed in the following contexts.

Table 1
Notations and the meanings.

Notation	Meaning
$\vec{\mathbf{X}}_m^{(t)}$	the m-th modality feature of new data chunk at t-th round
$\tilde{\mathbf{X}}_m^{(t)}$	the m-th modality feature of the existing data accumulated before round t
$\vec{\mathbf{L}}^{(t)}$	label matrix of new data chunk at t-th round
$\tilde{\mathbf{L}}^{(t)}$	label matrix of the existing data accumulated before round t
$\vec{\mathbf{B}}^{(t)}$	hash codes of new data chunk at t-th round
$\tilde{\mathbf{B}}^{(t)}$	hash codes of the existing data accumulated before round t
$\mathbf{S}$	the instance pairwise similarity
$\mathbf{A}^{(t)}$	hash codes of anchors at t-th round
n_t	number of instances at t-th round
N_t	number of the existing instances before round t
d_m	dimensionality of the m-th modality feature
c	number of classes
z	number of anchor points at each round
r	number of hash codes bits

3.2. Hash-Code learning

3.2.1. Similarity preserving

For supervised hashing methods, preserving the similarity of original space with binary codes is a crucial challenge. In other words, instances sharing at least one same class should have similar hash codes and vice versa. Most existing cross-modal online hashing methods commonly leverage the inner product between hash codes to approximate measuring similarity [35], however, the binary constraint of the hash codes makes the optimization an NP-hard problem. Some methods adopt a relaxation strategy [23], i.e., approximating the discrete $sign(\cdot)$ function using the continuous $tanh(\cdot)$ to avoid complex optimization of hash codes. Real-valued auxiliary variable strategy [35] is widely used for supervised online cross-modal retrieval, which replaces the $\mathbf{B}$ with a real-valued auxiliary variable. Both two strategies lead to large quantization errors and degrade the performance. Although there exist some studies that discretely solve the optimization problem, they are too complex and time-consuming to be used for large-scale datasets [38,39]. To overcome the above issue, like [37], the following problem is given,

$$\begin{aligned} \max_{\vec{\mathbf{B}}^{(t)}} \quad & \sum_{i=1}^{n_t} \sum_{j=1}^{N_{t-1}} \left[S_{ij}\Psi_{ij} - \log\left(1 + e^{\Psi_{ij}}\right)\right], \\ s.t. \quad & \vec{\mathbf{B}}^{(t)} \in \{-1,1\}^{n_t \times r}, \end{aligned} \tag{2}$$

where $\Psi_{ij} = \alpha \vec{\mathbf{B}}_{i*}^{(t)} \tilde{\mathbf{B}}_{j*}^{(t)\top}$, $\vec{\mathbf{B}}^{(t)}$ is the binary hash codes of newly coming data, $\tilde{\mathbf{B}}^{(t)}$ is the binary hash codes of existing data, and α is a hyper-parameter. In online hashing settings, relying only on the newly arrived data to learn their hash codes may lose the information of existing data and become suboptimal. Thus, we preserve the similarity between newly coming data and existing old data in the Hamming space in Eq. (2).

The likelihood function of $\mathbf{S}$ is as follows,

$$p\left(S_{ij} \mid \vec{\mathbf{B}}_{i*}^{(t)}, \tilde{\mathbf{B}}_{j*}^{(t)}\right) = \begin{cases} \sigma\left(\Psi_{ij}\right), & S_{ij} = 1 \\ 1 - \sigma\left(\Psi_{ij}\right), & S_{ij} = 0 \end{cases} \tag{3}$$

where $\sigma\left(\Psi_{ij}\right) = \frac{1}{1+e^{-\Psi_{ij}}}$. We can find that by maximizing the likelihood, the similarity, e.g., inner product, between $\vec{\mathbf{B}}_{i*}^{(t)}$ and $\tilde{\mathbf{B}}_{j*}^{(t)}$ is larger when $S_{ij} = 1$ and is small otherwise.

If all data is used for training, both time complexity and space complexity of $\mathbf{S}$ are $O(n_t N_{t-1})$, which makes our model intractable for large-scale datasets. Hence, we randomly select some instances from old existing data to compute $\mathbf{S}$.

At the t-th round ($t \geq 2$), to contribute equally from previous rounds, we randomly select z samples from each round as anchor points. The hash codes of anchors are donated as,

$$\mathbf{A}^{(t)} = [\vec{\mathbf{B}}_{i_q*}^{(1)};\ \vec{\mathbf{B}}_{i_q*}^{(2)};\ \ldots;\ \vec{\mathbf{B}}_{i_q*}^{(t-1)}], \tag{4}$$

where $\mathbf{A}^{(t)} \in \{-1,1\}^{(z \times (t-1)) \times r}$ is the hash codes matrix of anchors and $\{i_q\}_{q=1}^{z}$ is the index of selected samples.

At the first round, there is no data of the previous round. As an alternative, we choose z samples from the current round as anchor points. The hash codes of anchors are donated as,

$$\mathbf{A}^{(1)} = \vec{\mathbf{B}}_{i_q*}^{(1)}. \tag{5}$$

We replace $\tilde{\mathbf{B}}^{(t)}$ in Eq. (2) with hash codes of anchors $\mathbf{A}^{(t)}$ in Eq. (4) and Eq. (5), the optimization problem becomes the following one,

$$\begin{aligned} \max_{\vec{\mathbf{B}}^{(t)}} \quad & \sum_{i=1}^{n_t} \sum_{j=1}^{a_t} \left[S_{ij}\Psi_{ij} - \log\left(1 + e^{\Psi_{ij}}\right)\right], \\ s.t. \quad & \vec{\mathbf{B}}^{(t)} \in \{-1, 1\}^{n_t \times r}, \end{aligned} \tag{6}$$

where $\Psi_{ij} = \alpha \vec{\mathbf{B}}_{i*}^{(t)} \mathbf{A}_{j*}^{(t)\top}$ and a_t is the number of anchors at t-th round. Our method is expected to be robust for training large-scale data, thus we let $z << n_t$. The complexity of $\mathbf{S}$ decreases from $O(n_t N_{t-1})$ to $O(n_t a_t)$.

3.2.2. Label embedding

As mentioned previously, we consider the two samples are similar if they share the common class label. However, for multi-label data, this strategy seems to be suboptimal due to the coarse-grained binary representation of similarity between samples. Thus, we further seek help from the class label. Specifically, we embed supervised label information into to-be-learned hash codes so that binary codes are easy to be classified. From another perspective, at t-th round, the hash codes of the newly coming data and the old accumulated data are both expected to reconstruct label respectively, which is formulated as follows,

$$\begin{aligned} \min_{\vec{\mathbf{B}}^{(t)}} \quad & \| \vec{\mathbf{L}}^{(t)} - \vec{\mathbf{B}}^{(t)} \mathbf{P}^{(t)} \|_F^2 + \| \widetilde{\mathbf{L}}^{(t)} - \widetilde{\mathbf{B}}^{(t)} \mathbf{P}^{(t)} \|_F^2, \\ s.t. \quad & \vec{\mathbf{B}}^{(t)} \in \{-1, 1\}^{n_t \times r}, \end{aligned} \tag{7}$$

where $\mathbf{P}^{(t)} \in \mathbb{R}^{r \times c}$ is the projection matrix.

3.2.3. Overall objective function

Combining Eq. (6) and Eq. (7), we obtain the overall problem,

$$\begin{aligned} \max_{\vec{\mathbf{B}}^{(t)}} \quad & \textstyle\sum_{i=1}^{n_t} \sum_{j=1}^{a_t} \left[S_{ij}\Psi_{ij} - \log\left(1 + e^{\Psi_{ij}}\right)\right] \\ & -\theta \| \vec{\mathbf{L}}^{(t)} - \vec{\mathbf{B}}^{(t)} \mathbf{P}^{(t)} \|_F^2 - \theta \| \widetilde{\mathbf{L}}^{(t)} - \widetilde{\mathbf{B}}^{(t)} \mathbf{P}^{(t)} \|_F^2, \\ s.t. \quad & \vec{\mathbf{B}}^{(t)} \in \{-1, 1\}^{n_t \times r}, \end{aligned} \tag{8}$$

where θ is the trade-off parameter, a_t is the number of anchors at t-th round, and $\Psi_{ij} = \alpha \vec{\mathbf{B}}_{i*}^{(t)} \mathbf{A}_{j*}^{(t)\top}$. Through this formula, we can: 1) learn hash codes of the newly coming data while preserving the binary codes of the past streaming data unchanged to fit the online scenario. 2) build a framework simultaneously exploring similarity preserving and label embedding to generate discriminative binary codes. 3) learn unified hash codes from different modalities, which can not only incorporate multiple modalities but also easily extend our method to the more modalities case.

3.3. Optimization

As mentioned in Section 1, relaxation-based strategy and auxiliary variable strategy may lead to large quantization error, resulting in information loss and performance degradation. So, to optimize the overall objective function in Eq. (8), we design an alternating optimization strategy. In each step, we optimize variables with the others fixed.

3.3.1. Optimize $\vec{\mathbf{B}}^{(t)}$

Fixing other variables but $\vec{\mathbf{B}}^{(t)}$, Eq. (8) can be rewritten as,

$$\begin{aligned} & \mathcal{L} = \sum_{i=1}^{n_t} \sum_{j=1}^{a_t} \left[S_{ij}\Psi_{ij} - \log\left(1 + e^{\Psi_{ij}}\right)\right] - \theta \| \vec{\mathbf{L}}^{(t)} - \vec{\mathbf{B}}^{(t)} \mathbf{P}^{(t)} \|_F^2, \\ & s.t. \quad \vec{\mathbf{B}}^{(t)} \in \{-1, 1\}^{n_t \times r}, \end{aligned} \tag{9}$$

where $\Psi_{ij} = \alpha \vec{\mathbf{B}}_{i*}^{(t)} \mathbf{A}_{j*}^{(t)\top}$.

To reduce time complexity, we optimize $\vec{\mathbf{B}}^{(t)}$ bit-by-bit, which means one column of $\vec{\mathbf{B}}^{(t)}$ is updated each time with other columns fixed. Specifically, for k-th column $\vec{\mathbf{B}}_{*k}^{(t)}$, we can get a closed form solution by optimize the lower bound of $\mathcal{L}(\vec{\mathbf{B}}_{*k}^{(t)})$, which is efficient and simple. First, we construct the lower bound of $\mathcal{L}(\vec{\mathbf{B}}_{*k}^{(t)})$ as follows,

$$\hat{\mathcal{L}}\left(\vec{\mathbf{B}}_{*k}^{(t)}\right) = \vec{\mathbf{B}}_{*k}^{(t)\top} \left[\frac{\partial \mathcal{L}}{\partial \vec{\mathbf{B}}_{*k}^{(t)}}(g) - \mathbf{H}\vec{\mathbf{B}}_{*k}^{(t)}(g) \right] + \frac{\vec{\mathbf{B}}_{*k}^{(t)\top} \mathbf{H} \vec{\mathbf{B}}_{*k}^{(t)}}{2} + \mathrm{const} \tag{10}$$

where $\vec{\mathbf{B}}_{*k}^{(t)}(g)$ is the value of $\vec{\mathbf{B}}_{*k}^{(t)}$ at the g-th iteration, $\frac{\partial \mathcal{L}}{\partial \vec{\mathbf{B}}_{*k}^{(t)}}(g)$ is the gradient with respect to $\vec{\mathbf{B}}_{*k}^{(t)}(g)$, $\mathbf{H}$ is the Hessian of the objective function $\mathcal{L}$, and $\mathbf{H} = \frac{\partial^2 \mathcal{L}}{\partial \vec{\mathbf{B}}_{*k}^{(t)} \partial \vec{\mathbf{B}}_{*k}^{(t)\top}}$.

We compute the gradient and Hessian of the objective function $\mathcal{L}$ with respect to $\vec{\mathbf{B}}_{*k}^{(t)}$ as follows,

$$\begin{aligned} \tfrac{\partial \mathcal{L}}{\partial \vec{\mathbf{B}}_{*k}^{(t)}} &= \alpha \textstyle\sum_{j=1}^{a_t} \mathbf{A}_{jk}\left(\mathbf{S}_{*j} - \mathbf{E}_{*j}\right) + 2\theta\left(\left(\mathbf{L}\mathbf{P}^{(t)\top}\right)_{*k} - \sum_{j=1}^{r} \mathbf{D}_{jk} \vec{\mathbf{B}}_{*j}^{(t)}\right) \\ \mathbf{H} &= \tfrac{\partial^2 \mathcal{L}}{\partial \vec{\mathbf{B}}_{*k}^{(t)} \partial \vec{\mathbf{B}}_{*k}^{(t)\top}} = -\alpha^2 diag(e_1, e_2, \cdots, e_{n_t}) - 2\theta \mathbf{D}_{kk}\mathbf{I}, \end{aligned} \tag{11}$$

where a_t is the number of anchors at t-th round, $\mathbf{D} = \mathbf{P}^{(t)}\mathbf{P}^{(t)\top}$, $\mathbf{E} = \sigma\left(\Psi_{ij}\right) = \frac{1}{1+e^{-\Psi_{ij}}}$, $e_i = \sum_{j=1}^{n_t} \mathbf{E}_{ij}\left(1 - \mathbf{E}_{ij}\right)$, $diag\left(e_1, e_2, \cdots, e_{n_t}\right)$ is a diagonal matrix with the i-th diagonal element being e_i, and $\mathbf{I}$ is an identity matrix.

According to $0 < \mathbf{E}_{ij} < 1$, it is easy to get that $0 < \mathbf{E}_{ij}(1 - \mathbf{E}_{ij}) < \frac{1}{4}$, Thus, we have the lower bound of $\mathbf{H}$, which is defined as $\hat{\mathbf{H}}$ and $\hat{\mathbf{H}} = -\frac{a_t \alpha^2}{4}\mathbf{I}$. We substitute $\hat{\mathbf{H}}$ into the Eq. (10) and rewritten it as follows,

$$\hat{\mathcal{L}}\left(\vec{\mathbf{B}}_{*k}^{(t)}\right) = \vec{\mathbf{B}}_{*k}^{(t)\top} \left[\frac{\partial \mathcal{L}}{\partial \vec{\mathbf{B}}_{*k}^{(t)}}(g) - \hat{\mathbf{H}}\vec{\mathbf{B}}_{*k}^{(t)}(g) \right] + const, \tag{12}$$

where $\frac{1}{2}\vec{\mathbf{B}}_{*k}^{(t)\top} \mathbf{H} \vec{\mathbf{B}}_{*k}^{(t)} = -\frac{a_t n_t^2 \alpha^2}{4}\mathbf{I}$, which is independent of the variable $\vec{\mathbf{B}}_{*k}^{(t)}$. The proof that $\hat{\mathcal{L}}(\vec{\mathbf{B}}_{*k}^{(t)})$ is the lower bound of $\mathcal{L}(\vec{\mathbf{B}}_{*k}^{(t)})$ is similar to [37], due to page limit, the details are omitted, and one can easily prove it by analogy.

Now, we can learn k-th column $\vec{\mathbf{B}}_{*k}^{(t)}$ by maximizing the lower bound $\hat{\mathcal{L}}(\vec{\mathbf{B}}_{*k}^{(t)})$. The problem is defined as follows,

$$\max_{\vec{\mathbf{B}}_{*k}^{(t)}} \hat{\mathcal{L}}(\vec{\mathbf{B}}_{*k}^{(t)}), \qquad \text{s.t. } \vec{\mathbf{B}}_{*k}^{(t)} \in \{-1, +1\}^{n_t}. \tag{13}$$

Because of $\vec{\mathbf{B}}_{*k}^{(t)} \in \{-1, +1\}^{n_t}$, to maximize $\hat{\mathcal{L}}(\vec{\mathbf{B}}_{*k}^{(t)})$, we need to set $\vec{\mathbf{B}}_{ik}^{(t)} = 1$ if the i-th element of $\left[\frac{\partial \mathcal{L}}{\partial \vec{\mathbf{B}}_{*k}^{(t)}}(g) - \hat{\mathbf{H}}\vec{\mathbf{B}}_{*k}^{(t)}(g) \right]$ is greater than 0, and $\vec{\mathbf{B}}_{ik}^{(t)} = -1$ otherwise. So, we get the closed form solution to update $\vec{\mathbf{B}}_{*k}^{(t)}$ as follows,

$$\vec{\mathbf{B}}_{*k}^{(t)}(g+1) = sign\left(\tfrac{\partial \mathcal{L}}{\partial \vec{\mathbf{B}}_{*k}^{(t)}}(g) - \hat{\mathbf{H}}\vec{\mathbf{B}}_{*k}^{(t)}(g) \right). \tag{14}$$

3.3.2. Optimize $\mathbf{P}^{(t)}$

Fixing other variables but $\mathbf{P}^{(t)}$, Eq. (8) can be rewritten as,

$$\min_{\mathbf{P}^{(t)}} \| \vec{\mathbf{L}}^{(t)} - \vec{\mathbf{B}}^{(t)} \mathbf{P}^{(t)} \|_F^2 + \| \widetilde{\mathbf{L}}^{(t)} - \widetilde{\mathbf{B}}^{(t)} \mathbf{P}^{(t)} \|_F^2. \tag{15}$$

By setting the derivative of Eq. (15) w.r.t. $\mathbf{P}^{(t)}$ to zero, we can update it by,

$$\mathbf{P}^{(t)} = (\mathbf{M}_1^{(t)})^{-1} \mathbf{M}_2^{(t)}, \tag{16}$$

where

$$\begin{aligned} \mathbf{M}_1^{(t)} &= \tilde{\mathbf{B}}^{(t)\top} \tilde{\mathbf{B}}^{(t)} + \vec{\mathbf{B}}^{(t)\top} \vec{\mathbf{B}}^{(t)}, \\ \mathbf{M}_2^{(t)} &= \tilde{\mathbf{B}}^{(t)\top} \tilde{\mathbf{L}}^{(t)} + \vec{\mathbf{B}}^{(t)\top} \vec{\mathbf{L}}^{(t)}. \end{aligned} \tag{17}$$

Apparently, we have the following algebraic transformation,

$$\tilde{\mathbf{B}}^{(t)\top} \tilde{\mathbf{B}}^{(t)} = \left[\tilde{\mathbf{B}}^{(t-1)}; \vec{\mathbf{B}}^{(t-1)} \right]^\top \left[\tilde{\mathbf{B}}^{(t-1)}; \vec{\mathbf{B}}^{(t-1)} \right] = \mathbf{M}_1^{(t-1)}. \tag{18}$$

So, $\mathbf{M}_1^{(t)}$ and $\mathbf{M}_2^{(t)}$ can be rewritten as follows,

$$\begin{aligned} \mathbf{M}_1^{(t)} &= \mathbf{M}_1^{(t-1)} + \vec{\mathbf{B}}^{(t)\top} \vec{\mathbf{B}}^{(t)}, \quad \mathbf{M}_1^{(t-1)} = \tilde{\mathbf{B}}^{(t)\top} \tilde{\mathbf{B}}^{(t)}, \\ \mathbf{M}_2^{(t)} &= \mathbf{M}_2^{(t-1)} + \vec{\mathbf{B}}^{(t)\top} \vec{\mathbf{L}}^{(t)}, \quad \mathbf{M}_2^{(t-1)} = \tilde{\mathbf{B}}^{(t)\top} \tilde{\mathbf{L}}^{(t)}. \end{aligned} \tag{19}$$

3.4. Hash functions learning

DOCH is a two-step hashing method, which means the hash codes are generated in the first step and then we need to learn hash functions for multiple modalities based on the obtained binary codes in the second step [35]. Most existing hash methods learn classifiers to perform out-of-sample extensions for generating the binary codes of new queries, including SVM, neural network, and so on. In general, the more complex the classifier, the better results, and the longer it takes. In this paper, we adopt a simple linear regression model for each modality as hash functions. To support the online scenario, we expect it can update hash functions based on not only the newly coming data but also the old accumulated data at the t-th round. Thus, the function for m-th modality is defined as follows,

$$\min_{\mathbf{W}_m^{(t)}} \|\widetilde{\mathbf{B}}^{(t)} - \mathbf{W}_m^{(t)}\widetilde{\mathbf{X}}_m^{(t)}\|_F^2 + \|\vec{\mathbf{B}}^{(t)} - \mathbf{W}_m^{(t)}\vec{\mathbf{X}}_m^{(t)}\|_F^2. \tag{20}$$

By setting the derivative of Eq. (20) with respect to $\mathbf{W}_m^{(t)}$ to zero, we can derive the solution of $\mathbf{W}_m^{(t)}$ as follows,

$$\mathbf{W}_m^{(t)} = \mathbf{M}_3^{(t)}\left(\mathbf{M}_4^{(t)}\right)^{-1}, \tag{21}$$

where

$$\begin{aligned} \mathbf{M}_3^{(t)} &= \mathbf{M}_3^{(t-1)} + \vec{\mathbf{X}}_m^{(t)\top}\vec{\mathbf{X}}_m^{(t)}, \quad & \mathbf{M}_3^{(t-1)} &= \tilde{\mathbf{X}}_m^{(t)\top}\tilde{\mathbf{X}}_m^{(t)}, \\ \mathbf{M}_4^{(t)} &= \mathbf{M}_4^{(t-1)} + \vec{\mathbf{B}}^{(t)\top}\vec{\mathbf{X}}_m^{(t)}, \quad & \mathbf{M}_4^{(t-1)} &= \tilde{\mathbf{X}}_m^{(t)\top}\tilde{\mathbf{B}}^{(t)}. \end{aligned} \tag{22}$$

When a new query sample comes with the m-th modality feature, i.e., $\mathbf{x}_m$, we generate its hash code, i.e., $\mathbf{b}_q$ as follows,

$$\mathbf{b}_q = \operatorname{sign}\left(\mathbf{W}_m^{(t)}\mathbf{x}_m\right). \tag{23}$$

To have an overall view, the whole optimization algorithm is summarized in Algorithm 1.

Algorithm 1 Discrete Online Cross-Modal Hashing.

```
Input: Data chunks $[\vec{\mathbf{X}}_m^{(1)}, \cdots, \vec{\mathbf{X}}_m^{(t)}]$ with label matrix $[\vec{\mathbf{L}}^{(1)}, \cdots, \vec{\mathbf{L}}^{(t)}]$, iteration number of each round $h$, iterative number of updating hash codes for each iteration $g$ and hyper-parameters.
Output: Hash code matrix and hash functions.
  % The first step of DOCH, i.e., hash-code learning.
if at the first round then
    Randomly choose $z$ samples from the first round as anchor points;
    Generate $\vec{\mathbf{A}}^{(1)}$ by Eq. (5);
else
    Randomly select $z$ samples from each round as anchor points;
    Generate $\vec{\mathbf{A}}^{(t)}$ by Eq. (6);
end if
for i = 1→h do
    for j = 1→g do
        Update $k$-th column of $\vec{\mathbf{B}}^{(t)}$ by Eq. (14), where $k \in \{1, \cdots, r\}$;
    end for
    Update $\vec{\mathbf{P}}^{(t)}$ by Eq. (16);
end for
return Hash codes;
  % The second step of DOCH, i.e., hash functions learning.
Update $\mathbf{W}_m^{(t)}$ by solving Eq. (21), where $m \in \{1, \cdots, M\}$;
return Hash functions.
```

3.5. Discussions

3.5.1. Complexity analysis

We first give the computational complexity analysis of DOCH. At round t, the computational complexity of updating $\mathbf{P}^{(t)}$ is $O((crn_t + r^2n_t + 2rn_t)h)$. The computational complexity of updating $\vec{\mathbf{B}}^{(t)}$ is $O((2rn_t + 2n_ta_t + 4n_t)rhg)$. Updating $\mathbf{W}_m$ requires $O(d_mrn_t + r^2n_t)$. Thereinto, h is the iterative number of each round, g is the iterative number of updating hash codes for each iteration, a_t is the number of anchors, and we let $a_t << n_t$. It can be seen that the computational complexity of our method is linearly with the size of the newly coming data n_t, which proves the scalability of DOCH.

In addition, we avoid computing similarity matrix $\mathbf{S} \in \{0, 1\}^{n_t \times N_{t-1}}$ at t-th round by anchor points, The complexity of $\mathbf{S}$ decreases from $O(n_tN_{t-1})$ to $O(n_ta_t)$.

To conclude, both the computational and space complexity are linear with the size of newly coming data chunk, showing that DOCH is scalable for large-scale cross-modal retrieval.

3.5.2. Convergence analysis

The objective function Eq. (8) is convex to one variable by fixing the others. Therefore, a lower or equal value of the objective function will be got by optimizing one variable in each step. Our iterative updating rules will monotonously decrease the objective function value and the optimization process eventually achieves a local minimum after several iterations.

3.6. Extension to more modalities

As mentioned previously, DOCH can be easily extended to more modalities. Specifically, the training process of unified hash codes for all modalities is the same as that for a bimodal case, in which the overall problem is defined in Eq. (8). The hash functions for new modalities need to be learned independently by Eq. (20), where the m-th modality is a newly added one. And the problem can be solved by the proposed online optimization algorithm in Section 3.3.

4. Experiment

In this section, we first introduce the datasets and the experimental settings including evaluation, parameters setting, and baselines. And then we provide the experimental results and further analysis.

4.1. Datasets

To evaluate the performance of our proposed method, we conducted extensive experiments on two widely-used benchmark datasets, i.e., MIRFlickr [40], and NUS-WIDE [41]. Both the two datasets are with two modalities, i.e., image and text.

MIRFlickr includes 25,000 image-text pairs labeled by at least one of 24 categories. Similar to [35], we only selected those instances that textual tags appear at least 20 times and 20,015 image-text pairs are left finally. Each image is represented by a 512-dimensional GIST feature vector and each text is represented as a BOW vector associated with $1,386$ user-provided tags. 2,000 instances are randomly selected as a query set and the remaining are served as the training set. Based on the online hashing settings, we split the training set into 9 data chunks. Each of the first 8 chunks contains 2,000 samples and the last chunk contains 2,015 instances.

NUS-WIDE is a large-scale dataset collected from Flickr, which contains 269,648 instances. We selected 21 most frequent labels and the corresponding $195,834$ image-text pairs are left. For each image, the $4,096$-dimensional output of the pre-trained VGG-F modal [42] is used. Each text is represented as a 1,000-dimensional binary tagging vector. We randomly selected $2,000$ image-text pairs as the query set and $100,000$ samples as the training set. Similarly, to support the online scenario, the training set is divide into 10 chunks, and each of chunk contains 10,000 points.

MIRFlickr and NUS-WIDE are both composed of multi-label data. In this paper, the two samples are thought to be similar if they share at least one common class label and vice versa.

4.2. Experimental settings

4.2.1. Evaluation metrics

We conducted two cross-modal retrieval tasks to evaluate the performance of DOCH, "Image-to-Text" task retrieves similar texts given an image as a query and "Text-to-Image" task retrieves similar images given a text as a query. In this paper, a widely-used evaluation criterion is adopted, i.e., Mean Average Precision (MAP). For this metric, a larger value indicates better performance. First, we will give the MAP results at the last data chunk. And then, for more deep evaluation of the online scene in each round, we updated the hash functions and evaluated the performance when new data comes which means we would report 8 MAP results (8 chunks) on MIRFlickr and 10 results (10 chunks) on NUS-WIDE in each code length.

4.2.2. Baselines

To validate the effectiveness of the proposed method, we compared DOCH with some state-of-the-art cross-modal hashing baselines including DCH [20], LCMFH [43], SCRATCH [44], OCMFH [24], and LEMON [35]. Among them, OCMFH and LEMON are online cross-modal methods and others are batch-based methods. In addition, OCMFH belongs to unsupervised online cross-modal hashing and others are supervised methods. For all offline cross-modal methods, we used accumulated data to retrain their hash functions and regenerated their hash codes at each round. The source codes of baselines are all publicly available.

4.2.3. Experimental details

The parameters of DOCH are selected experimentally. Our method achieves the best results when $\theta = 0.1$ on MIRFlickr and $\theta = 20$ on NUS-WIDE. Additionally, we found that α is sensitive to hash code length and parameter sensitivity analyses are reported in Section 4.4.3. The iterative number of each round h is set to 7 and the iterative number of updating hash codes for each iter g is set to 3. Moreover, we set the number of anchors $z = 50$ on MIRFlickr and $z = 80$ on NUS-WIDE.

All experiments are conducted on a Linux workstation with Intel Xeon E5-2650 CPU @2.20GHz, 128GB RAM.

4.3. Comparison with baselines

To evaluate the performance of DOCH in online scenarios, we listed the MAP results at the last round of DOCH and all the comparison methods on MIRFlickr and NUS-WIDE in Table 2 and Table 3. In addition, the MAP-round curves of all methods on two datasets with the cases of 16 and 32 bits are plotted in Fig. 2 and Fig. 3.

From these results, we can draw the following observations:

- DOCH outperforms all the adopted state-of-the-art baselines in most cases on two datasets, demonstrating its effectiveness.
- Supervised hashing methods, i.e., DCH, LCMFH, SCRATCH, and LEMON always outperform the unsupervised ones, i.e., OCMFH, which proves that the label information can benefit the supervised cross-modal hashing methods in generating compact and effective hash codes. And DOCH outperforms other supervised hashing methods, indicating that DOCH can generate more discriminative hash codes by embedding supervised label information into to-be-learned hash codes and directly preserving the similarity between newly coming data and old existing data in the hamming space.
- LEMON learns binary codes by exploring label information and the correlation between old data and new data. Compared with it, DOCH gets better performances in most cases on two benchmark datasets, which demonstrates the effectiveness of discretely learning binary codes.
- MAP-round curves plotted in Fig. 2 and Fig. 3 on two datasets demonstrate a similar trend, which reveals the effectiveness of our method.

Table 2
The MAP results of various methods on MIRFlickr at the last chunk. The best MAP values of each case are shown in boldface and the results with underlines are the best results of all baselines.

Task	Method	8 bits	16 bits	32 bits	64 bits	128 bits
Image to Text	DCH [20]	0.6940	0.7127	0.7105	0.7195	0.7402
	LCMFH [43]	0.5985	0.6024	0.5743	0.5909	0.6089
	SCRATCH [44]	0.6502	0.6498	0.6471	0.6553	0.6574
	OCMFH [24]	0.5582	0.5564	0.5556	0.5546	0.5538
	LEMON [35]	0.7210	0.7399	0.7372	0.7477	0.7505
	DOCH	**0.7589**	**0.7737**	**0.7829**	**0.7881**	**0.7914**
Text to Image	DCH [20]	0.7488	0.7699	0.7694	0.7817	0.8046
	LCMFH [43]	0.6811	0.6871	0.6877	0.7065	0.7127
	SCRATCH [44]	0.7495	0.7703	0.7733	0.7894	0.7957
	OCMFH [24]	0.5581	0.5562	0.5554	0.5546	0.5539
	LEMON [35]	0.8042	0.8230	0.8277	0.8367	0.8389
	DOCH	**0.8165**	**0.8361**	**0.8482**	**0.8575**	**0.8632**

Table 3
The MAP results of various methods on NUS-WIDE at the last chunk. The best MAP values of each case are shown in boldface and the results with underlines are the best results of all baselines.

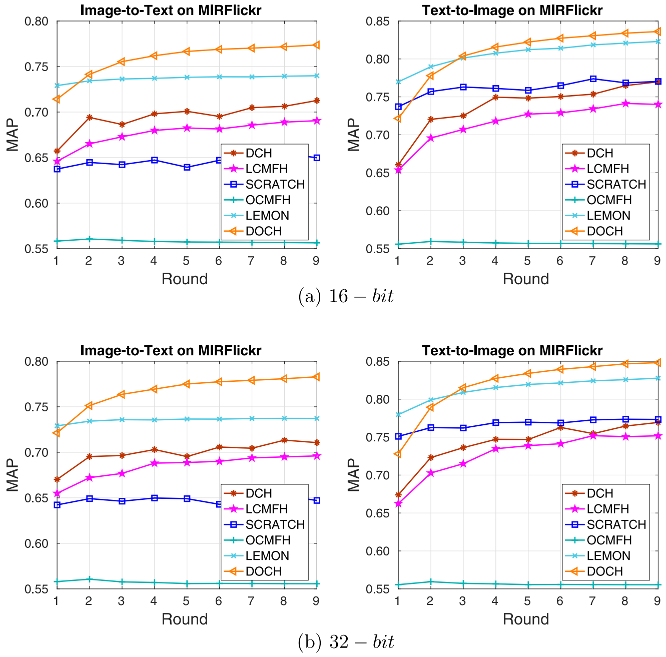
Task	Method	8 bits	16 bits	32 bits	64 bits	128 bits
Image to Text	DCH [20]	0.7397	0.7710	0.7765	0.7878	0.7864
	LCMFH [43]	0.5626	0.5878	0.6241	0.5551	0.6220
	SCRATCH [44]	0.7425	0.7912	0.8017	0.8159	0.8242
	OCMFH [24]	0.4276	0.4530	0.4849	0.4692	0.4504
	LEMON [35]	0.7820	0.8071	0.8174	0.8286	0.8307
	DOCH	**0.7821**	**0.8215**	**0.8375**	**0.8470**	**0.8533**
Text to Image	DCH [20]	0.6890	0.7081	0.7186	0.7274	0.7231
	LCMFH [43]	0.4588	0.5450	0.5354	0.5563	0.5351
	SCRATCH [44]	0.6839	0.7276	0.7248	0.7413	0.7507
	OCMFH [24]	0.4053	0.4384	0.4722	0.4556	0.4414
	LEMON [35]	0.7240	0.7482	0.7560	0.7665	0.7702
	DOCH	**0.7249**	**0.7556**	**0.7729**	**0.7822**	**0.7885**

In a nutshell, our method can incorporate similarity preserving and label embedding into one unified framework and discretely learn uniform hash codes for multiple modalities by an efficient optimization algorithm. DOCH provides the most satisfactory results on both datasets with all code lengths, which verifies its effectiveness for online cross-modal retrieval.

4.4. Further analysis

4.4.1. Ablation experiments

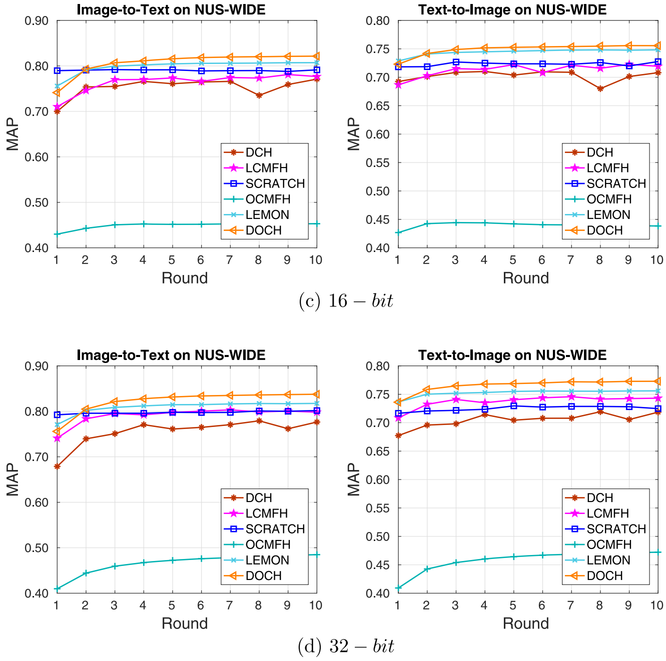
DOCH embeds the label information into the phase of learning hash codes and generates unified hash codes for multiple modalities. To verify their effectiveness, two derivatives are designed, i.e., DOCH-1 and DOCH-2. DOCH-1 sets $\theta = 0$ in Eq. (24). For the sec-



(a) $16-bit$

(b) $32-bit$

Fig. 2. Cross-modal retrieval in online scenarios. The MAP-round curves of all methods on MIRFlickr.



(c) $16-bit$

(d) $32-bit$

Fig. 3. Cross-modal retrieval in online scenarios. The MAP-round curves of all methods on NUS-WIDE.

ond derivative DOCH-2, its objective function at t-th round is,

$$\begin{aligned}\max_{\vec{\mathbf{U}}^{(t)},\vec{\mathbf{V}}^{(t)}} \quad & \sum_{i=1}^{n_t}\sum_{j=1}^{a_t}\left[S_{ij}\Psi_{ij} - \log\left(1+e^{\Psi_{ij}}\right)\right] \\ & + \sum_{i=1}^{n_t}\sum_{j=1}^{a_t}\left[S_{ij}\Theta_{ij} - \log\left(1+e^{\Theta_{ij}}\right)\right] \\ & -\theta \parallel \vec{\mathbf{L}}^{(t)} - \vec{\mathbf{U}}^{(t)}\mathbf{P}_x^{(t)} \parallel_F^2 -\theta \parallel \tilde{\mathbf{L}}^{(t)} - \tilde{\mathbf{U}}^{(t)}\mathbf{P}_x^{(t)} \parallel_F^2 \\ & -\theta \parallel \vec{\mathbf{L}}^{(t)} - \vec{\mathbf{V}}^{(t)}\mathbf{P}_y^{(t)} \parallel_F^2 -\theta \parallel \tilde{\mathbf{L}}^{(t)} - \tilde{\mathbf{V}}^{(t)}\mathbf{P}_y^{(t)} \parallel_F^2, \\ s.t. \quad & \vec{\mathbf{U}}^{(t)} \in \{-1,1\}^{n_t\times r}, \quad \vec{\mathbf{V}}^{(t)} \in \{-1,1\}^{n_t\times r},\end{aligned} \tag{24}$$

where $\Psi_{ij} = \alpha\vec{\mathbf{U}}_{i*}^{(t)\top}\tilde{\mathbf{U}}_{j*}^{(t)}$, $\Theta_{ij} = \alpha\vec{\mathbf{V}}_{i*}^{(t)\top}\tilde{\mathbf{V}}_{j*}^{(t)}$, $\vec{\mathbf{U}}^{(t)}$ and $\tilde{\mathbf{U}}_{j*}^{(t)}$ are the hash codes of newly coming data and old existing data for image modality separately, $\vec{\mathbf{V}}^{(t)}$ and $\tilde{\mathbf{V}}_{j*}^{(t)}$ are corresponding hash codes for text modality. For DOCH-2, we still adopt the strategy of randomly selecting anchors.

Table 4
The MAP results of various methods on MIRFlickr at the last chunk. The best MAP values of each case are shown in boldface the results with underlines are the suboptimal results.

Task	Method	8 bits	16 bits	32 bits	64 bits	128 bits
	DOCH-1	0.7531	0.7642	0.7720	0.7768	0.7799
I→ T	DOCH-2	0.7475	0.7636	0.7727	0.7790	0.7820
	DOCH	**0.7589**	**0.7737**	**0.7829**	**0.7881**	**0.7914**
	DOCH-1	0.8160	0.8342	0.8449	0.8558	0.8611
T→ I	DOCH-2	0.8125	0.8337	0.8474	0.8561	0.8615
	DOCH	**0.8165**	**0.8361**	**0.8482**	**0.8575**	**0.8632**

Table 5
The MAP results of various methods on NUS-WIDE at the last chunk. The best MAP values of each case are shown in boldface the results with underlines are the suboptimal results.

Task	Method	8 bits	16 bits	32 bits	64 bits	128 bits
	DOCH-1	0.7612	0.7988	0.8265	0.8413	0.8517
I→ T	DOCH-2	0.7600	0.8117	0.8248	0.8373	0.8417
	DOCH	**0.7821**	**0.8215**	**0.8375**	**0.8470**	**0.8533**
	DOCH-1	0.7074	0.7413	0.7662	0.7812	**0.7908**
T→ I	DOCH-2	0.7045	**0.7578**	0.7660	0.7803	0.7885
	DOCH	**0.7229**	0.7556	**0.7729**	**0.7822**	0.7885

Table 6
The training time (seconds) of all methods on MIRFlickr.

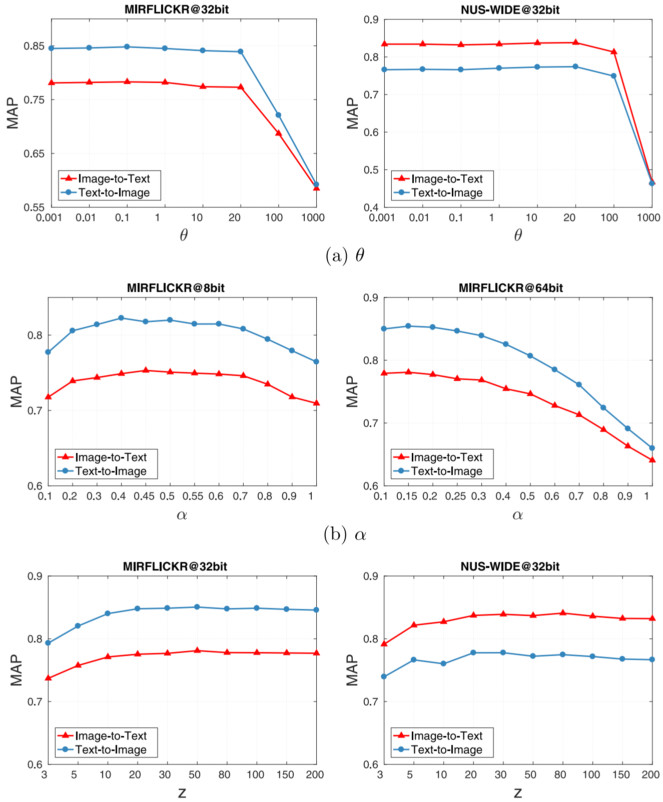
Bits	Method	R1	R2	R3	R4	R5	R6	R7
	DCH [20]	3.22	5.23	8.87	11.87	15.30	18.06	21.59
	LCMFH [43]	2.41	3.58	4.80	5.72	6.17	7.58	9.22
8	SCRATCH [44]	1.53	2.06	2.71	3.93	3.77	4.26	5.02
	OCMFH [24]	13.35	1.52	1.36	1.43	1.49	1.48	1.46
	LEMON [35]	0.24	0.18	0.15	0.19	0.18	0.18	0.14
	DOCH	1.18	1.16	1.45	1.47	1.71	1.86	2.01
	DCH [20]	4.08	6.50	11.71	14.62	19.60	23.62	27.81
	LCMFH [43]	3.02	4.51	5.13	6.83	7.39	10.12	10.33
32	SCRATCH [44]	2.29	3.96	3.01	5.53	4.77	6.01	6.91
	OCMFH[24]	15.74	1.43	1.34	1.93	1.79	1.88	1.83
	LEMON [35]	0.25	0.24	0.22	0.19	0.23	0.26	0.19
	DOCH	2.13	2.43	2.66	3.12	3.46	4.26	4.46

The experimental results on MIRFlickr and NUS-WIDE are presented in Table 4 and Table 5. From this table, we can find:

- DOCH outperforms DOCH-1, demonstrating that by embedding supervised label information into to-be-learned hash codes, better hash codes can be obtained;
- DOCH-2 learns different binary codes for heterogeneous modalities, which is worse than DOCH in most cases, revealing the importance of unified hash codes. As far as we know, DLFH also generates different hash codes for heterogeneous modalities, which may result in poor results composed to DOCH, which learns uniform binary codes.

4.4.2. Time cost analysis

In Section 3.5.1, we analyze that the time complexity of DOCH is linearly dependent on the size of newly coming data n_t. To demonstrate it, we further conducted experiments and the time cost of all methods on MIRFlickr in the cases of 8 and 32 bits is listed in Table 6. We can observe that: 1) With the round increasing, the time cost of DOCH has a small rise because more anchors are involved in the learning process. 2) Although our method does not hold the best training efficiency compared to LEMON and OCMFH, we get the best performances and the time can be reduced by adjusting the number of anchors when time consumption is strictly constrained. 3) Bit-by-bit optimization adopted by DOCH



(a) θ

(b) α

(c) z

Fig. 4. Parameter sensitivity analysis of θ, α, and z.

Table 7
The values of α on MIRFlickr and NUS-WIDE.

	8 bits	16 bits	32 bits	64 bits	128 bits
MIRFlickr	0.45	0.35	0.25	0.15	0.10
NUS-WIDE	0.45	0.35	0.25	0.15	0.10

will consume more time as the number of bits increases, but generally, it is not set too large and the consumption is acceptable.

4.4.3. *Sensitivity to parameters*

To analyze the influence of parameters, we conducted experiments on MIRFlickr and NUS-WIDE. The results of θ are plotted in Fig. 4-(a). From it, we can find that the performance maintains satisfactory when θ ranges from 10^{-3} to 20 on two datasets. DOCH achieves the best results when $\theta = 0.1$ on MIRFlickr and $\theta = 20$ on NUS-WIDE, respectively. In the Fig4-(b), we reported the map values of multiple α on MIRFlickr in the case of 8 and 64 bits. It can be seen that α is sensitive to hash code length, the performance maintains satisfactory when α ranges from 0.4 to 0.5 with the case of 8 bits and ranges from 0.1 to 0.2 with the case of 64 bits. In order to find the best α value, we conducted extensive experiments, the values of α on different cases of hash code length are listed in Table 7. We recorded the MAP results of DOCH on MIRFlickr and NUS-WIDE by varying z from 3 to 200. The results on both the Image-to-Text and the Text-to-Image tasks in the case of 32 bits are plotted in Fig. 4-(c). We can observe that DOCH achieves satisfactory performance even when a small number of samples, i.e. 20, are used as anchors. However, in pursuit of the best performance, through the experiment, we set $z = 50$ on MIRFlickr and $z = 80$ on NUS-WIDE.

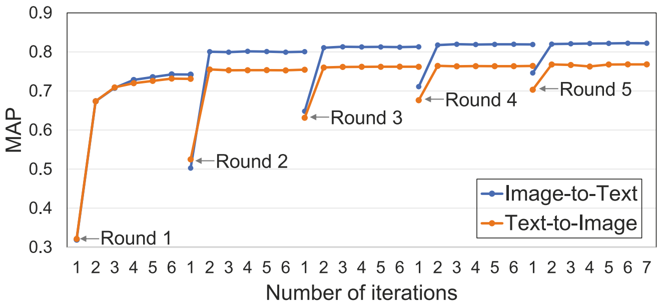


Fig. 5. Convergence analysis on NUS-WIDE.

4.4.4. *Convergence analysis*

We adopt an iterative optimization to update variables when new data comes. Fig. 5 plots the convergence on NUS-WIDE in the case of 16 bits in the first five rounds. It can be seen that our method requires six iterations to get convergence when $t = 1$, and it only takes two iterations to update variables when $t \geq 2$, which validates that the objective function value will monotonously decrease and the optimization process eventually achieves a local minimum after several iterations.

5. Conclusion and future work

In this paper, we propose a novel supervised online hashing method for cross-modal retrieval, i.e., Discrete Online Cross-modal Hashing (DOCH for short). DOCH directly preserves the similarity between newly coming data and existing data in the hamming space to avoid large quantization errors. Besides, a discrete online optimization without any relaxation is adopted to directly learn binary codes of newly coming data. Extensive experiments on two real-world benchmarks have been conducted and the results demonstrate the superiority of DOCH over the state-of-the-art baselines.

As an effective and efficient retrieval algorithm, DOCH can also be easily extended to more modalities. In future work, we will explore its role in other cross-modal retrieval tasks, such as using a text as query to retrieve similar videos. Moreover, deep hashing methods integrate feature extraction and hash-code learning in one framework and have achieved impressive performance. How to combine deep learning with DOCH while ensuring high efficiency is another interesting exploration direction.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This work was supported in part by the National Natural Science Foundation of China under Grant 61872428, in part by Shandong Provincial Key Research and Development Program under Grant 2019JZZY010127, in part by Natural Science Foundation of Shandong Province under Grant ZR2019ZD06, ZR2020QF036, in part by the Fundamental Research Funds of Shandong University under Grant 2019GN075, and in part by the Major Program of the National Natural Science Foundation of China under Grant 61991411.

References

[1] D. Mandal, K.N. Chaudhury, S. Biswas, Generalized semantic preserving hashing for cross-modal retrieval, IEEE Trans. Image Process. 28 (1) (2019) 102–112.

[2] G. Song, X. Tan, J. Zhao, M. Yang, Deep robust multilevel semantic hashing for multi-label cross-modal retrieval, Pattern Recognit (2021) 108084.
[3] Y. Xiong, Y. Xu, X. Shu, Cross-view hashing via supervised deep discrete matrix factorization, Pattern Recognit 103 (2020) 107270.
[4] J. Wang, T. Zhang, J. Song, N. Sebe, H.T. Shen, A survey on learning to hash, IEEE Trans Pattern Anal Mach Intell 40 (4) (2018) 769–790.
[5] Q. Ma, C. Bai, J. Zhang, Z. Liu, S. Chen, Supervised learning based discrete hashing for image retrieval, Pattern Recognit 92 (2019) 156–164.
[6] Y. Zhan, X. Luo, Y. Wang, X. Xu, Supervised hierarchical deep hashing for cross–modal retrieval, in: Proceedings of thed ACM International Conference on Multimedia, 2020, pp. 3386–3394.
[7] X. Liu, X. Nie, Q. Zhou, Y. Yin, Supervised discrete hashing with mutual linear regression, in: Proceedings of the ACM International Conference on Multimedia, 2019, pp. 1561–1568.
[8] P. Zhang, C. Li, M. Liu, L. Nie, X. Xu, Semi-relaxation supervised hashing for cross-modal retrieval, in: Proceedings of thed ACM International Conference on Multimedia, 2017, pp. 1762–1770.
[9] Y. Gong, S. Lazebnik, A. Gordo, F. Perronnin, Iterative quantization: a procrustean approach to learning binary codes for large-scale image retrieval, IEEE Trans Pattern Anal Mach Intell 35 (12) (2013) 2916–2929.
[10] W. Liu, J. Wang, R. Ji, Y. Jiang, S. Chang, Supervised hashing with kernels, in: Proceedings of the International Conference on Computer Vision and Pattern Recognition, 2012, pp. 2074–2081.
[11] L. Wang, J. Yang, M. Zareapoor, Z. Zheng, Cluster-wise unsupervised hashing for cross-modal similarity search, Pattern Recognit 111 (2021) 107732.
[12] J. Tang, K. Wang, L. Shao, Supervised matrix factorization hashing for cross–modal retrieval, IEEE Trans. Image Process. 25 (7) (2016) 3157–3166.
[13] Z. Chen, Y. Wang, H. Li, X. Luo, L. Nie, X. Xu, A two-step cross-modal hashing by exploiting label correlations and preserving similarity in both steps, in: Proceedings of thed ACM International Conference on Multimedia, 2019, pp. 1694–1702.
[14] Q. Jiang, W. Li, Deep cross-modal hashing, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 3270–3278.
[15] V.E. Liong, J. Lu, Y. Tan, Cross-modal discrete hashing, Pattern Recognit 79 (2018) 114–129.
[16] K. Ding, B. Fan, C. Huo, S. Xiang, C. Pan, Cross-modal hashing via rank-order preserving, IEEE Trans Multimedia 19 (3) (2017) 571–585.
[17] X. Luo, X. Yin, L. Nie, X. Song, Y. Wang, X. Xu, SDMCH: supervised discrete manifold-embedded cross-modal hashing, in: Proceedings of the International Joint Conference on Artificial Intelligence, 2018, pp. 2518–2524.
[18] Z. Lin, G. Ding, M. Hu, J. Wang, Semantics-preserving hashing for cross-view retrieval, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 3864–3872.
[19] L. Liu, Z. Lin, L. Shao, F. Shen, G. Ding, J. Han, Sequential discrete hashing for scalable cross-modality similarity retrieval, IEEE Trans. Image Process. 26 (1) (2017) 107–118.
[20] X. Xu, F. Shen, Y. Yang, H.T. Shen, X. Li, Learning discriminative binary codes for large-scale cross-modal retrieval, IEEE Trans. Image Process. 26 (5) (2017) 2494–2507.
[21] L. Xie, J. Shen, L. Zhu, Online cross-modal hashing for web image retrieval, in: Proceedings of the AAAI Conference on Artificial Intelligence, 2016, pp. 294–300.
[22] T. Yao, G. Wang, L. Yan, X. Kong, Q. Su, C. Zhang, Q. Tian, Online latent semantic hashing for cross-media retrieval, Pattern Recognit 89 (2019) 1–11.
[23] M. Lin, R. Ji, S. Chen, X. Sun, C. Lin, Similarity-preserving linkage hashing for online image retrieval, IEEE Trans. Image Process. 29 (2020) 5289–5300.
[24] D. Wang, Q. Wang, Y. An, X. Gao, Y. Tian, Online collective matrix factorization hashing for large-scale cross-media retrieval, in: Proceedings of the International ACM SIGIR conference on Research and Development in Information Retrieval, 2020, pp. 1409–1418.
[25] C. Leng, J. Wu, J. Cheng, X. Bai, H. Lu, Online sketching hashing, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 2503–2511.
[26] J. Fang, H. Xu, Q. Wang, T. Wu, Online hash tracking with spatio-temporal saliency auxiliary, Comput. Vision Image Understanding 160 (2017) 57–72.
[27] Y. Zhan, X. Luo, Y. Sun, Y. Wang, Z. Chen, X. Xu, Weakly-supervised online hashing, in: Proceedings of the IEEE International Conference on Multimedia and Expo, 2021.
[28] L. Huang, Q. Yang, W. Zheng, Online hashing, in: Proceedings of the International Joint Conference on Artificial Intelligence, 2013, pp. 1422–1428.
[29] M. Lin, R. Ji, H. Liu, X. Sun, Y. Wu, Y. Wu, Towards optimal discrete online hashing with balanced similarity, in: Proceedings of the AAAI Conference on Artificial Intelligence, 2019, pp. 8722–8729.
[30] M. Lin, R. Ji, H. Liu, X. Sun, S. Chen, Q. Tian, Hadamard matrix guided online hashing, Int J Comput Vis 128 (8) (2020) 2279–2306.
[31] L. Xie, J. Shen, J. Han, L. Zhu, L. Shao, Dynamic multi-view hashing for online image retrieval, in: Proceedings of the International Joint Conference on Artificial Intelligence, 2017, pp. 3133–3139.
[32] X. Lu, L. Zhu, Z. Cheng, J. Li, X. Nie, H. Zhang, Flexible online multi-modal hashing for large-scale multimedia retrieval, in: Proceedings of thed ACM International Conference on Multimedia, 2019, pp. 1129–1137.
[33] G. Ding, Y. Guo, J. Zhou, Collective matrix factorization hashing for multimodal data, in: Proceedings of the International Conference on Computer Vision and Pattern Recognition, 2014, pp. 2083–2090.
[34] M. Qi, Y. Wang, A. Li, Online cross-modal scene retrieval by binary representation and semantic graph, in: Proceedings of the ACM International Conference on Multimedia, 2017, pp. 744–752.
[35] Y. Wang, X. Luo, X. Xu, Label embedding online hashing for cross-modal retrieval, in: Proceedings of the ACM International Conference on Multimedia, 2020, pp. 871–879.
[36] J. Yi, X. Liu, Y.M. Cheung, X. Xu, Y. He, Efficient online label consistent hashing for large-scale cross-modal retrieval, in: Proceedings of the IEEE International Conference on Multimedia and Expo, 2021, pp. 1–6.
[37] Q. Jiang, W. Li, Discrete latent factor model for cross-modal hashing, IEEE Trans. Image Process. 28 (7) (2019) 3490–3501.
[38] W. Kang, W. Li, Z. Zhou, Column sampling based discrete supervised hashing, in: Proceedings of the AAAI Conference on Artificial Intelligence, 2016, pp. 1230–1236.
[39] G. Lin, C. Shen, Q. Shi, A. van den Hengel, D. Suter, Fast supervised hashing with decision trees for high-dimensional data, in: Proceedings of the International Conference on Computer Vision and Pattern Recognition, 2014, pp. 1971–1978.
[40] M.J. Huiskes, M.S. Lew, The MIR flickr retrieval evaluation, in: Proceedings of the ACM International Conference on Multimedia Information Retrieval, 2008, pp. 39–43.
[41] T. Chua, J. Tang, R. Hong, H. Li, Z. Luo, Y. Zheng, NUS-WIDE: A real-world web image database from national university of singapore, in: Proceedings of ACM International Conference on Image and Video Retrieval, 2009.
[42] T. Carvalho, E.R.S.D. Rezende, M.T.P. Alves, F.K.C. Balieiro, R.B. Sovat, Exposing computer generated images by eye's region classification via transfer learning of vgg19 cnn, in: Proceedings of the International Conference on Machine Learning and Applications, 2017, pp. 866–870.
[43] D. Wang, X. Gao, X. Wang, L. He, Label consistent matrix factorization hashing for large-scale cross-modal similarity search, IEEE Trans Pattern Anal Mach Intell 41 (10) (2019) 2466–2479.
[44] C. Li, Z. Chen, P. Zhang, X. Luo, L. Nie, W. Zhang, X. Xu, SCRATCH: A scalable discrete matrix factorization hashing for cross-modal retrieval, in: Proceedings of the ACM International Conference on Multimedia, 2018, pp. 1–9.
[45] Y. Wang, Z. Chen, X. Luo, R. Li, X. Xu, Fast Cross-Modal Hashing With Global and Local Similarity Embedding, IEEE Trans Cybern (2021) 1–14.
[47] P. Zhang, Y. Li, Z. Huang, H. Yin, Privacy protection in deep multi-modal retrieval, Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (2021) 634–643.

Yu-Wei Zhan received a bachelor's degree in software engineering from Shandong University, China, in 2019. She is currently pursuing the M.S. degree at the School of Software, Shandong University, Jinan, China. Her current research interests include machine learning, binary hashing, multimedia retrieval, and computer vision.

Yongxin Wang received her B.S. and M.S. degrees in computer science from Shandong Normal University, China, in 2014 and 2017, respectively. She is currently working toward the Ph.D. degree at the School of Software, Shandong University. Her research interests include machine learning, hashing, multimedia retrieval, and computer vision.

Yu Sun received the bachelor's degree in software engineering from Shandong University, China, in 2020. She is currently pursuing the M.S. degree at the School of Software, Shandong University, Jinan, China. Her current research interests include machine learning, binary hashing, multimedia retrieval, and computer vision.

Xiao-Ming Wu is currently pursuing the bachelor's degree at the School of Software, Shandong University, Jinan, China. His current research interests include machine learning, multimedia retrieval.

Xin Luo is currently an assistant professor with the School of Software, Shandong University, Jinan, China. His research interests mainly include machine learning, multimedia retrieval, and computer vision. He has published over 20 papers on TIP, TKDE, ACM MM, SIGIR, WWW, IJCAI, et al.

Xin-Shun Xu is currently a professor with the School of Software, Shandong University. He is the founder and the leader of MIMA (Machine Intelligence and Media Analysis) group of Shandong University. His research interests include machine learning, information retrieval, data mining and image/video analysis, and retrieval.