

Deep Cross-Modal Hashing with Hashing Functions and Unified Hash Codes Jointly Learning

Rong-Cheng Tu, Xian-Ling Mao, Bing Ma, Yong Hu, Tan Yan, Wei Wei and Heyan Huang

Abstract—Due to their high retrieval efficiency and low storage cost, cross-modal hashing methods have attracted considerable attention. Generally, compared with shallow cross-modal hashing methods, deep cross-modal hashing methods can achieve a more satisfactory performance by integrating feature learning and hash codes optimizing into a same framework. However, most existing deep cross-modal hashing methods either cannot learn a unified hash code for the two correlated data-points of different modalities in a database instance or cannot guide the learning of unified hash codes by the feedback of hashing function learning procedure, to enhance the retrieval accuracy. To address the issues above, in this paper, we propose a novel end-to-end Deep Cross-Modal Hashing with Hashing Functions and Unified Hash Codes Jointly Learning (DCHUC). Specifically, by an iterative optimization algorithm, DCHUC jointly learns unified hash codes for image-text pairs in a database and a pair of hash functions for unseen query image-text pairs. With the iterative optimization algorithm, the learned unified hash codes can be used to guide the hashing function learning procedure; Meanwhile, the learned hashing functions can feedback to guide the unified hash codes optimizing procedure. Extensive experiments on three public datasets demonstrate that the proposed method outperforms the state-of-the-art cross-modal hashing methods.

Index Terms—Cross-modal Retrieval, Deep Hashing, Multimedia



1 INTRODUCTION

WITH a tremendous amount of multimedia data being generated on the Internet everyday such as texts, images and so on, similarity-preserving hashing methods [1], [2], [3], [4], [5], [6], [7], [8], [9] have been extensively studied for large-scale multimedia search due to their high retrieval efficiency and low storage cost. Because the corresponding data of different modalities may have semantic correlations, it is essential to support cross-modal retrieval that returns relevant results of one modality when querying another modality, e.g., retrieving images with text queries. Hence, cross-modal hashing methods [10], [11], [12], [13], [14], [15] get more and more attention.

Roughly speaking, cross-modal hashing methods can be divided into shallow cross-modal hashing methods [16], [2], [17], [10], [18], [19] and deep cross-modal hashing methods [20], [21], [14], [22], [11], [23]. Shallow cross-modal hashing methods mainly use hand-crafted features to learn projections for mapping each example into a binary code. The feature extraction procedure in shallow cross-modal hashing methods is independent of the hash codes learning procedure. It means that the shallow cross-modal hashing methods may not achieve satisfactory performance in real applications, because the hand-crafted features might not be optimally suitable for hash codes optimizing procedure. Compared with shallow cross-modal hashing methods, deep cross-modal hashing methods can integrate feature learning and hash

codes learning into a same framework, and capture non-linear correlations among cross-modal instances more effectively to get better performance, where each instance contains two correlated data-points of different modalities like image-text pairs.

However, the existing deep cross-modal hashing methods either cannot learn a unified hash code for the two correlated data-points of different modalities in a database instance or cannot guide the learning of unified hash codes by the feedback of hashing function learning procedure, to enhance the retrieval accuracy. First, most deep cross-modal hashing methods assume that there are different hash codes for the two correlated data-points of different modalities in a database instance, and then try to decrease the gap between two hash codes through optimizing certain pre-defined loss functions. Thus, they just learn the similar hash codes for two correlated data-points of different modalities in a same instance, and cannot obtain unified hash codes. However, the unified hash code schema has been proved that it can enhance the retrieval accuracy [24], [25], [26]. Second, as far as we know, until now there is only one deep cross-modal hashing method that can learn unified hash codes [11]. The method is a two step framework. It first learns unified hash codes for instances in a database, and then utilizes the learned unified hash codes to learn modal-specific hashing function. It means the deep hashing method cannot guide the learning of unified hash codes by the feedback of hashing function learning procedure.

To address the issues above, in this paper, we propose a novel Deep Cross-Modal Hashing with Hashing Functions and Unified Hash Codes Jointly Learning, called DCHUC. DCHUC can jointly learn unified hash codes for database instances and modal-specific hashing functions for unseen query points in an end-to-end framework. More specifically, by minimising the objective function, DCHUC uses a four-step iterative scheme to optimize the unified hash codes of the database instances and the hash codes of query

- R. Tu, X. Mao, B. Ma, Y. Hu, T. Y and H. Huang are with the Department of Computer Science and Technology, Beijing Institute of Technology, Beijing 100081, Chain.
E-mail: {tu_rc, maoxl, 3220180838, huyong, yantan, hhy63}@bit.edu.cn.
- W. Wei is with School of Computer Science, Huazhong University of Science and Technology, Wuhan 430074, Chain.
E-mail: weiw@hust.edu.cn

data-points generated by the learned hashing networks. With the iterative optimization algorithm, the learned unified hash codes can guide the hashing functions learning procedure; Meanwhile, the learned hashing function can feedback to guide the unified hash codes optimizing procedure. Moreover, the objective function consists of a hashing loss and a classification loss. The hashing loss is used to make the learned hash codes can preserve both inter-modal and intra-modal similarity, and the classification loss can be used to make the learned hashing codes preserve more discriminative semantic information.

In addition, because the training phase of deep models is typically time-consuming, so it is hard to use all instances in a large-scale database to train hashing model. Inspired by ADSH [27], we use an asymmetric scheme to reduce the training time complexity to $O(mn)$. Specially, we samples m anchors instances from n database instances ($m \ll n$) to approximate query datasets, and constructs an asymmetric affinity to supervise hashing functions learning for unseen query instances and unified hash codes optimizing for instances in a database.

To summarize, the main contributions of DCHUC are outlined as follows:

- To the best of our knowledge, DCHUC is the first deep method that can jointly learn unified hash codes for database instances and hashing functions for unseen query points in an end-to-end framework. By using the end-to-end framework, our method can get the high-quality hash codes to improve the retrieval accuracy.
- By treating the query instances and database instances in an asymmetric way, DCHUC can use the whole set of database instances in training phase to generate higher-quality hash codes even if the size of a database is large.
- Experiments on three large-scale datasets show that DCHUC can outperform the state-of-the-art cross-modal hashing baselines in real applications.

2 RELATED WORK

In this section, we briefly review the related works of cross-modal hashing methods, including shallow cross-modal hashing methods and deep cross-modal hashing methods.

2.1 Shallow Cross-Modal Hashing Methods

Shallow cross-modal hashing methods [4], [28], [29], [25], [18], [19] mainly use hand-crafted features to learn a single pair of linear or non-linear projections to map each example into a binary vector. The representative methods in this category include Cross Modality Similarity Sensitive Hashing (CMSSH) [4], Semantic Correlation Maximization (SCM) [28], Cross View Hashing (CVH) [29], Latent Semantic Sparse Hashing (LSSH) [25], Collective Matrix Factorization Hashing (CMFH) [26], Semantics Preserving Hashing (SePH) [30], Supervised Discrete Manifold-embedded Cross-Modal Hashing (SDMCH) [18], Discrete Latent Factor hashing (DLFH) [19] and Discrete Cross-modal Hashing (DCH) [31]. CMSSH is a supervised hashing methods, which designs a cross-modal hashing method by preserving the intra-class similarity via eigen-decomposition and boosting. SCM utilizes label information to learn a modality-specific transformation, and preserves the maximal correlation between modalities. CVH presents an unsupervised cross-modal spectral hashing method so that the cross-modality similarity is also preserved in the

learned hash functions. LSSH utilizes sparse coding and matrix factorization in the common space to obtain a unified binary by a latent space learning method. CMFH learns a unified binary hash code by performing matrix factorization with latent factor model in the training stage. SePH generates a unified binary hash code by constructing an affinity matrix in a probability distribution while at the same time minimizing the Kullback-Leibler divergence. SDMCH generates binary hash codes by exploiting the non-linear manifold structure of data and constructing the correlations among heterogeneous multiple modalities with semantic information. DLFH directly learns the binary hash codes without continuous relaxation by a discrete latent factor model. DCH jointly learns the unified binary codes and the modality-specific hash functions under the classification framework with discrete optimization algorithm.

Despite of significant progress in this category has been achieved, the performance of hand-crafted feature based methods are still unsatisfactory in many real-world applications. Because the feature extraction procedure is independent of the hash-code learning procedure in hand-crafted feature based methods, which means that the hand-crafted features might not be optimally suitable for the hash codes optimizing procedure.

2.2 Deep Cross-Modal Hashing Methods

Recently, deep cross-modal hashing methods [21], [32], [20], [14], [22], [11] have been proposed to achieve promising performance due to the powerful arbitrary non-linear representation of deep neural network. For example, deep visual-semantic hashing (DVSH) [21] learns a visual semantic fusion network with cosine hinge loss to generate the binary codes and learns modality-specific deep networks to obtain hashing functions. However, DVSH can only be used for some special cross-modal cases where one of the modalities have to be temporal dynamics. Deep cross-modal hashing (DCMH) [32] utilized a negative log-likelihood loss to generate cross-modal similarity preserving hash codes by an end-to-end deep learning framework. Correlation Autoencoder Hashing (CAH) [20] learns hashing functions by designing an auto-encoder architecture to jointly maximize the feature and semantic correlation between different modalities. Adversarial cross-modal retrieval (ACMR) [14] utilizes a classification manner with adversarial learning approach to discriminate between different modalities and generate binary hash codes. Self-supervised adversarial hashing (SSAH) [22] generates binary hash codes by utilizing two adversarial networks to jointly model different modalities and capture their semantic relevance under the supervision of the learned semantic feature. Cross-modal deep variational hashing (CMDVH) [11] uses a two step framework. In the first step the method learns unified hash code for image-text pair in a database, and utilize the learned unified hash codes to learn hashing functions in the second step. Thus, for CMDVH, the learned hashing function in the second stage cannot give feedback to guide unified hash codes optimizing.

Typically, deep cross-modal hashing methods can outperform shallow hashing methods in terms of retrieval accuracy. However, most of existing deep cross-modal hashing methods cannot bridge the modality gap well to generate unified hash codes for image-text pairs in a database. Although CMDVH can generates unified binary codes for points of modalities, its hashing function learning procedure cannot feedback to guide the unified hash codes optimizing. Hence, CMDVH cannot get the optimal unified

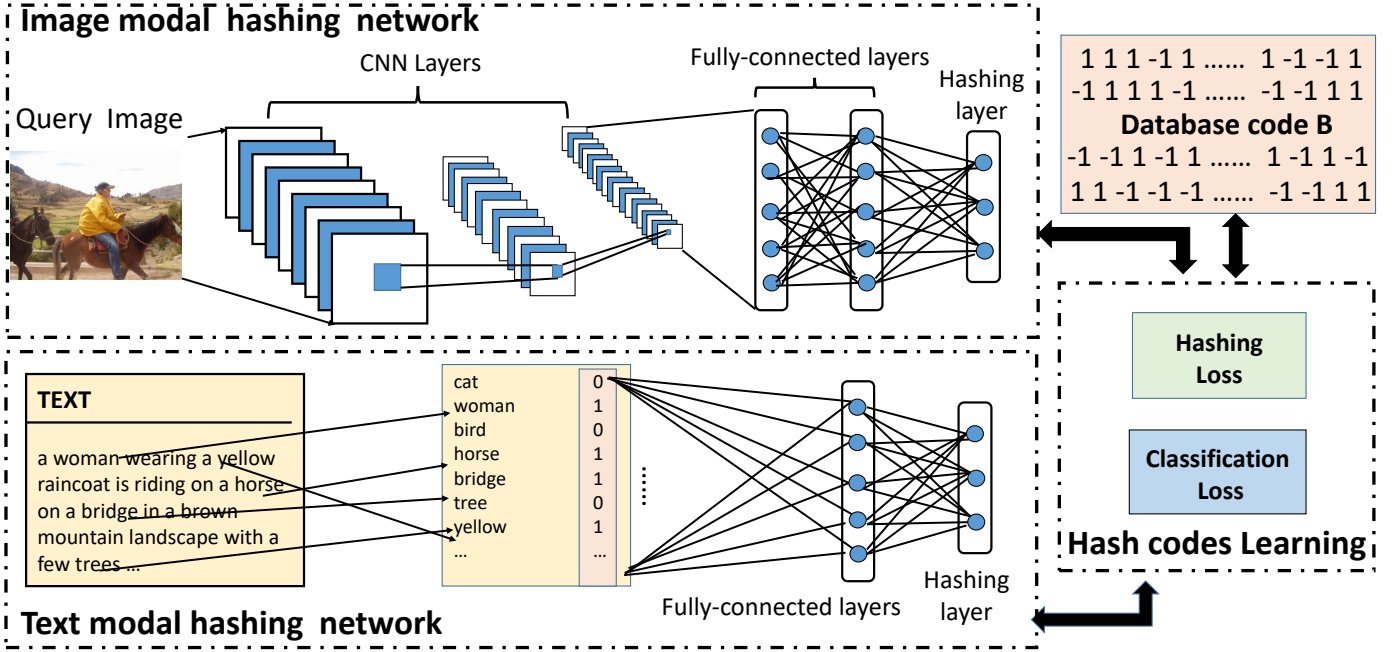


Fig. 1. The DCHUC learning framework. It contains three parts: image modal hashing network, text modal hashing network and hash code optimizing. The hash codes of image modal query data-points can be generated by the image modal hashing network with an element-wise function $sign(\cdot)$, and the hash codes of text modal data-points can be generated text modal hashing network with an element-wise function $sign(\cdot)$. In the hash codes optimizing part, a four-step iterative scheme is used to optimize hash codes for both database instances and query instances by minimising the hashing loss and the classification loss

hash codes to bridge the modality gap well. Furthermore, please note that, although DCH can jointly learn unified hash codes for instances in a database and hashing functions for query instances, it is a shallow hashing method. Its feature extraction procedure is independent of the hash codes learning procedure, and DCH need use all the database instances to learn hashing functions which means it is hard to reconstruct DCH to a deep architecture. Thus, we propose a novel deep hashing method that can learn the unified hash codes for instances in a database and hashing functions for query instances in an end-to-end framework.

3 OUR METHOD

3.1 Problem Definition

Assume that we have n training instances in a database, and each instance has two modal data points. Without loss of generality, we use image-text databases for illustration in this paper, which means that each instance in the database has both a data point of text modality and a data point of image modality. We use $O = \{o_i\}_{i=1}^n$ to denote a cross-modal dataset with n instances, and $o_i = (x_i, y_i, l_i)$, where x_i and y_i denote the original image and text points in the i^{th} instance o_i , respectively. $l_i = [l_{i1}, l_{i2}, \dots, l_{ic}]^T$ is the label annotation assigned to o_i , where c is the class number. If o_i belongs to the j^{th} class $l_{ij} = 1$, otherwise $l_{ij} = 0$. Furthermore, a pairwise similarity matrix $S \in \{-1, +1\}^{n \times n}$ is used to describe the semantic similarities between two instances. If $S_{ij} = 1$, it means that o_i is semantically similar to o_j , otherwise $S_{ij} = -1$. Specifically, if two instances o_i and o_j are annotated by multiple labels, we define $S_{ij} = 1$ when o_i and o_j share at least one label, otherwise $S_{ij} = -1$.

Given the above database O and similarity information S , the goal of DCHUC is to learn the similarity-preserving hash codes $B = \{b_i\}_{i=1}^n \in \{-1, +1\}^{n \times k}$ for instances in the database,

where k is the length of each binary code and b_i denotes the learned hash code for the instance o_i , i.e., a unified hash code for the image-text pair x_i and y_i . Meanwhile, the Hamming distance between b_i and b_j should be as small as possible when $S_{ij} = 1$. Otherwise, the Hamming distance should be as large as possible. Moreover, in order to generate a binary code for any unseen image modal query point x_q or text modal query point y_q , DCHUC should learn two modal-specific hashing functions $b_{x_q} = \mathcal{F}(x_q) \in \{-1, +1\}^k$ and $b_{y_q} = \mathcal{P}(y_q) \in \{-1, +1\}^k$, respectively. In order to learn the two hash functions, we sample a subset or use the whole set of O as the query set $Q = O^\Phi$ for training, where O^Φ denotes the query instances indexed by Φ from the database O . Moreover, we use $\Upsilon = \{1, 2, \dots, n\}$ to denote the indices of all the database instances and $\Phi = \{i_1, i_2, \dots, i_m\} \subseteq \Upsilon$ to denote the indices of the m sampled query instances, and X^Φ and Y^Φ denote image modal points and text modal points in query set Q , respectively. Correspondingly, the similarity between query instances and database instances can be denoted as $S^\Phi \in \{-1, +1\}^{m \times n}$, which is formed by the rows of S indexed by Φ . In addition, in this paper, $sign(\cdot)$ is an element-wise sign function which returns 1 if the element is positive and returns -1 otherwise.

3.2 Deep Cross-Modal Hashing with Hashing Functions and Unified Hash Codes Jointly Learning

The model architecture for DCHUC is shown in Fig. 1, which contains three parts: **image modal hashing network**, **text modal hashing network** and **hash codes optimizing**.

For the image modal hashing network part, it contains a convolutional neural network (CNN) which is adapted from Alexnet [33]. The CNN component contains eight layers. The first seven layers are the same as those in Alexnet [33]. The eighth layer is

a fully-connected layer with the output being the learned image features, which is named as hashing layer. The hashing layer contains k units where k is the length of hash codes. An activation function $\tanh(\cdot)$ is used to make the output features close to "−1" or "+1". We use $\mathbf{v}_i = \mathcal{F}(\mathbf{x}_i; \Theta) \in \mathcal{R}^k$ to denote the final output features of the image modal hashing network.

For the text modal hashing network part, a neural network containing two fully-connected layers is used to learn text modal features. We represent each text point \mathbf{y}_i as a bag-of-words (BoW) vector, and use the BoW as the input of the two-fully-connected neural network. The first fully-connected layer has 10,240 hidden units, and the activation function for the first fully-connect layer is RELU [33]. The second fully-connected layer is also named as hashing layer with k nodes. Similar to the image feature learning part, a $\tanh(\cdot)$ function is used as an activation function to make the output features close to "−1" or "+1". We use $\mathbf{t}_i = \mathcal{P}(\mathbf{y}_i; \Psi) \in \mathcal{R}^k$ to denote the final output features of the text modal hashing network.

For the hash codes optimizing part, it will optimize hash codes for both database instances and query instances the objective function whose details will be introduced in section 3.3. More specially, with a four-step iterative scheme, the unified hash codes \mathbf{B} for database instances will be learned directly and the modal-specific hashing functions can be learned by back-propagation algorithm which will be introduced in section 3.4 in detail. Furthermore, the hash codes for query instances are generated by the final output features of modal-specific hashing network with an element-wise function $\text{sign}(\cdot)$. Specifically, for an image modal query point \mathbf{x}_i , we can get its binary hash codes $\mathbf{h}_i = \text{sign}(\mathbf{v}_i)$; for a text modal query point \mathbf{y}_i , its binary hash codes can be generated by $\mathbf{g}_i = \text{sign}(\mathbf{t}_i)$.

3.3 Objective Function

The goal of DCHUC is to map instances in the database and the unseen query data-points into a semantic similarity-preserving Hamming space where the hash codes of data-points from the same categories should be similar no matter which modalities they belong to, and the hash codes of data-points from different categories should be dissimilar. In the following, we present more details about the objective function of our CMDAH.

In order to bridge the gap across different modalities well, we first assume the image point \mathbf{x}_i and text point \mathbf{y}_i for any instance \mathbf{o}_i in a database share the same hash code \mathbf{b}_i , i.e., learn a unified hash code \mathbf{b}_i for an image-text pair \mathbf{x}_i and \mathbf{y}_i . Thus, the hash code \mathbf{b}_i can preserve the image modal information and text modal information at the same time. Moreover, in order to make the learned hash codes of instances in the database and the hash codes of query data-points generated by the learned hashing functions can preserve the semantic similarity, one common way is to minimize the Frobenius norm loss between the semantic similarities and inner product of binary code pairs. Therefore, the hashing loss can be defined as follow:

$$\begin{aligned} \min_{\mathbf{B}, \mathbf{H}, \mathbf{G}} \mathcal{L}_h &= \left\| \mathbf{H}\mathbf{B}^T - k\mathbf{S}^\Phi \right\|_F^2 + \left\| \mathbf{G}\mathbf{B}^T - k\mathbf{S}^\Phi \right\|_F^2 \\ &+ \mu \left\| \mathbf{H}\mathbf{G}^T - k\mathbf{S}^\Phi \right\|_F^2 \\ \text{s.t. } \mathbf{B} &= [\mathbf{b}_1, \mathbf{b}_2, \dots, \mathbf{b}_n]^T \in \{-1, +1\}^{n \times k}, \\ \mathbf{H} &= \text{sign}(\mathbf{V}) \in \{-1, +1\}^{m \times k}, \\ \mathbf{G} &= \text{sign}(\mathbf{T}) \in \{-1, +1\}^{m \times k} \end{aligned} \quad (1)$$

where μ is a hyper-parameter, $\mathbf{B} \in \{-1, +1\}^{n \times k}$ denotes the unified binary hash codes for n database instances; \mathbf{S}^Φ denotes the columns of \mathbf{S}^Φ indexed by Φ ; $\mathbf{H} \in \{-1, +1\}^{m \times k}$ denotes the binary hash codes for m images modal query data-points, and $\mathbf{G} \in \{-1, +1\}^{m \times k}$ denotes the binary hash codes for m text modal query data-points; $\mathbf{V} = [\mathbf{v}_{i_1}, \mathbf{v}_{i_2}, \dots, \mathbf{v}_{i_m}]^T$ is the output of images modal hashing network for image query set \mathbf{X}^Φ , and $\mathbf{T} = [\mathbf{t}_{i_1}, \mathbf{t}_{i_2}, \dots, \mathbf{t}_{i_m}]^T$ is the output of text modal hashing network for text query set \mathbf{Y}^Φ .

Furthermore, in order to make the learned hashing codes preserve more discriminative semantic information, we expect the learned hashing codes can be ideal for classification too. Then the classification loss function can be defined as follow:

$$\begin{aligned} \min_{\mathbf{B}, \mathbf{H}, \mathbf{G}, \mathbf{W}} \mathcal{L}_c &= \alpha \left(\left\| \mathbf{H}\mathbf{W} - \mathbf{L}^\Phi \right\|_F^2 + \left\| \mathbf{G}\mathbf{W} - \mathbf{L}^\Phi \right\|_F^2 \right) \\ &+ \beta \left\| \mathbf{B}\mathbf{W} - \mathbf{L} \right\|_F^2 + \eta \left\| \mathbf{W} \right\|_F^2 \\ \text{s.t. } \mathbf{B} &= [\mathbf{b}_1, \mathbf{b}_2, \dots, \mathbf{b}_n]^T \in \{-1, +1\}^{n \times k}, \\ \mathbf{H} &= \text{sign}(\mathbf{V}) \in \{-1, +1\}^{m \times k}, \\ \mathbf{G} &= \text{sign}(\mathbf{T}) \in \{-1, +1\}^{m \times k}. \end{aligned} \quad (2)$$

where $\mathbf{L} = [l_1, l_2, \dots, l_n]^T \in \{0, 1\}^{n \times c}$ is the label matrix of instances in the database \mathbf{O} , and $\mathbf{L}^\Phi \in \{0, 1\}^{m \times c}$ denotes the label matrix of query instances indexed by Φ from the label matrix \mathbf{L} . $\mathbf{W} = [\mathbf{w}_1, \mathbf{w}_2, \dots, \mathbf{w}_c] \in \mathcal{R}^{k \times c}$ and $\mathbf{w}_j \in \mathcal{R}^{k \times 1}$ is the classification projected vector of the class j .

Thus, our objective hashing function can be defined as follow:

$$\begin{aligned} \min_{\mathbf{B}, \mathbf{H}, \mathbf{G}, \mathbf{W}} \mathcal{L} &= \mathcal{L}_h + \mathcal{L}_c \\ \text{s.t. } \mathbf{B} &= [\mathbf{b}_1, \mathbf{b}_2, \dots, \mathbf{b}_n]^T \in \{-1, +1\}^{n \times k}, \\ \mathbf{H} &= \text{sign}(\mathbf{V}) \in \{-1, +1\}^{m \times k}, \\ \mathbf{G} &= \text{sign}(\mathbf{T}) \in \{-1, +1\}^{m \times k}. \end{aligned} \quad (3)$$

However, it is hard to learn functions $\mathbf{V} = \mathcal{F}(\mathbf{X}^\Phi; \Theta)^T$ and $\mathbf{T} = \mathcal{P}(\mathbf{Y}^\Phi; \Psi)^T$ due to the derivation of $\text{sign}(\cdot)$ function is 0. Moreover, considering the query set is sampled from the whole database, the hash codes generated by the learned hashing function should be the same with the directly learned hash codes, i.e., if an instance \mathbf{o}_i in the database is sampled as query instance, then the hash code h_i for image modality data-point and g_i for text modality data-point in \mathbf{o}_i should be the same with \mathbf{b}_i . Thus, we can further reformulate Formula (3) as:

$$\begin{aligned} \min_{\mathbf{B}, \mathbf{V}, \mathbf{T}} \mathcal{L} &= \left\| \mathbf{V}\mathbf{B}^T - k\mathbf{S}^\Phi \right\|_F^2 + \left\| \mathbf{T}\mathbf{B}^T - k\mathbf{S}^\Phi \right\|_F^2 \\ &+ \mu \left\| \mathbf{V}\mathbf{T}^T - k\mathbf{S}^\Phi \right\|_F^2 + \beta \left\| \mathbf{B}\mathbf{W} - \mathbf{L} \right\|_F^2 \\ &+ \alpha \left(\left\| \mathbf{V}\mathbf{W} - \mathbf{L}^\Phi \right\|_F^2 + \left\| \mathbf{T}\mathbf{W} - \mathbf{L}^\Phi \right\|_F^2 \right) \\ &+ \eta \left\| \mathbf{W} \right\|_F^2 + \gamma \left\| \mathbf{B}^\Phi - 0.5(\mathbf{V} + \mathbf{T}) \right\|_F^2 \\ \text{s.t. } \mathbf{B} &\in \{-1, +1\}^{n \times k}. \end{aligned} \quad (4)$$

where $\alpha, \beta, \eta, \gamma, \mu$ are hyper-parameters, $\mathbf{B}^\Phi \in \{-1, +1\}^{m \times k}$ is formed by the rows of \mathbf{B} indexed by Φ .

3.4 Optimization

In order to optimize Formula (4), we propose a four-step iterative scheme as shown below. More specifically, in each iteration we sample a query set from the database and then carry out our

Algorithm 1 Learning algorithm for DCHUC

Require: Database instances $\mathcal{O} = \{\mathbf{X}, \mathbf{Y}, \mathbf{L}\}$, the length of hash codes k .

Ensure: Database instances codes \mathcal{B} , image modal hashing network parameters Θ and text modal hashing network parameters Ψ .

- 1: Initialize parameters: $\mathcal{B}, \Theta, \Psi, \alpha, \eta, \gamma$. learning rate: lr , iteration number: t_{out}, t_{in} , the size of mini-batch $z = 64$ (see Implementation Details).
- 2: Utilize label \mathbf{L} to generate similarity matrix \mathbf{S} .
- 3: **repeat**
- 4: Randomly generate index set Φ and sample m instances $\mathcal{O}^\Phi = \{\mathbf{X}^\Phi, \mathbf{Y}^\Phi, \mathbf{L}^\Phi\}$ from database \mathcal{O} as query set. Select \mathbf{S}^Φ from \mathbf{S}
- 5: **for** $iter = 1, 2, \dots, t_{in}$ **do**
- 6: **for** $iter_batch = 1, 2, \dots, m/z$ **do**
- 7: Randomly sample z image points from \mathbf{X}^Φ as a mini-batch
- 8: Update parameter Θ based on Formula (5)
- 9: **end for**
- 10: **for** $iter_batch = 1, 2, \dots, m/z$ **do**
- 11: Randomly sample z image points from \mathbf{Y}^Φ as a mini-batch
- 12: Update parameter Ψ based on Formula (6)
- 13: **end for**
- 14: **end for**
- 15: **for** $iter_bit = 1, 2, \dots, k$ **do**
- 16: Update \mathbf{B}_{*iter_bit} based on Formula (12)
- 17: **end for**
- 18: Update \mathbf{W} based on Formula (14)
- 19: **until** Up to t_{out}

learning algorithm based on both the query set and database. The whole four-step learning algorithm for DCHUC is briefly outlined in Algorithm 1, and the detailed derivation steps will be introduced in the following content of this subsection.

3.4.1 Learn Θ with Ψ, \mathcal{B} and \mathbf{W} fixed

When Ψ, \mathcal{B} and \mathbf{L} are fixed, we update the parameter Θ of image hashing network by using a mini-batch stochastic gradient descent with back-propagation (BP) algorithm. More specifically, for each sampled image point \mathbf{x}_i in \mathbf{X}^Φ , we first compute the following gradient:

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \mathbf{v}_i} &= 2 \sum_{j=1}^n [(\mathbf{v}_i^T \mathbf{b}_j - k \mathbf{S}_{ij}^\Phi) \mathbf{b}_j] + 2\mu \sum_{j=1}^m [(\mathbf{v}_i^T \mathbf{t}_j - k \mathbf{S}_{\Phi ij}^\Phi) \mathbf{t}_j] \\ &\quad + 2\alpha \sum_{j=1}^c [(\mathbf{v}_i^T \mathbf{w}_j - \mathbf{L}_{ij}^\Phi) \mathbf{w}_j] + \gamma (\mathbf{v}_i + \mathbf{t}_i - 2\mathbf{b}_i) \end{aligned} \quad (5)$$

Then we can compute $\frac{\partial \mathcal{L}}{\partial \Theta}$ based on $\frac{\partial \mathcal{L}}{\partial \mathbf{v}_i}$ by using chain rule, and use BP to update the parameter Θ .

3.4.2 Learn Ψ with Θ, \mathcal{B} and \mathbf{L} fixed

When Θ and \mathcal{B} are fixed, we also update the parameter Ψ of text hashing network by using a mini-batch stochastic gradient descent

with BP algorithm. More specifically, for each sampled text point \mathbf{y}_i in \mathbf{Y}^Φ , we first compute the following gradient:

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \mathbf{t}_i} &= 2 \sum_{j=1}^n [(\mathbf{t}_i^T \mathbf{b}_j - k \mathbf{S}_{ij}^\Phi) \mathbf{b}_j] + 2\mu \sum_{j=1}^m [(\mathbf{t}_i^T \mathbf{v}_j - k \mathbf{S}_{\Phi ij}^\Phi) \mathbf{v}_j] \\ &\quad + 2\alpha \sum_{j=1}^c [(\mathbf{t}_i^T \mathbf{w}_j - \mathbf{L}_{ij}^\Phi) \mathbf{w}_j] + \gamma (\mathbf{v}_i + \mathbf{t}_i - 2\mathbf{b}_i) \end{aligned} \quad (6)$$

Then we can compute $\frac{\partial \mathcal{L}}{\partial \Psi}$ based on $\frac{\partial \mathcal{L}}{\partial \mathbf{t}_i}$ by using chain rule, and use BP to update the parameter Ψ .

3.4.3 Learn \mathcal{B} with Θ, Ψ and \mathbf{W} fixed

When Θ, Ψ and \mathbf{W} are fixed, we can reformulate Formula (4) as follows:

$$\begin{aligned} \min_{\mathcal{B}} \mathcal{L} &= \left\| \mathbf{V} \mathbf{B}^T - k \mathbf{S}^\Phi \right\|_F^2 + \left\| \mathbf{T} \mathbf{B}^T - k \mathbf{S}^\Phi \right\|_F^2 \\ &\quad + \beta \left\| \mathbf{B} \mathbf{W} - \mathbf{L} \right\|_F^2 + \gamma \left\| \mathbf{B}^\Phi - 0.5(\mathbf{V} + \mathbf{T}) \right\|_F^2 \\ &= \left\| \mathbf{V} \mathbf{B}^T \right\|_F^2 - 2ktr(\mathbf{B} \mathbf{V}^T \mathbf{S}^\Phi) + \left\| \mathbf{T} \mathbf{B}^T \right\|_F^2 \\ &\quad - 2ktr(\mathbf{B} \mathbf{T}^T \mathbf{S}^\Phi) + \beta \left\| \mathbf{B} \mathbf{W} \right\|_F^2 - 2\beta tr(\mathbf{B} \mathbf{W} \mathbf{L}^T) \\ &\quad - \gamma tr(\mathbf{B}^\Phi (\mathbf{V}^T + \mathbf{T}^T)) + const \\ s.t. \quad &\mathcal{B} \in \{-1, +1\}^{n \times k} \end{aligned} \quad (7)$$

where $const$ is a constant independent of \mathcal{B} and $tr(\cdot)$ is the trace norm. For convenience of calculations, we can further reformulate Formula (7) as follows:

$$\begin{aligned} \min_{\mathcal{B}} \mathcal{L} &= \left\| \mathbf{V} \mathbf{B}^T \right\|_F^2 + \left\| \mathbf{T} \mathbf{B}^T \right\|_2^F + \beta \left\| \mathbf{B} \mathbf{W} \right\|_F^2 - tr(\mathbf{B} (\gamma \hat{\mathbf{V}}^T \\ &\quad + \gamma \hat{\mathbf{T}}^T + 2k \mathbf{V}^T \mathbf{S}^\Phi + 2k \mathbf{T}^T \mathbf{S}^\Phi + 2\beta \mathbf{W} \mathbf{L}^T)) \\ &\quad + const \\ &= \left\| \mathbf{V} \mathbf{B}^T \right\|_F^2 + \left\| \mathbf{T} \mathbf{B}^T \right\|_2^F + \beta \left\| \mathbf{B} \mathbf{W} \right\|_F^2 \\ &\quad - tr(\mathbf{B} \mathbf{D}) + const \\ s.t. \quad &\mathcal{B} \in \{-1, +1\}^{n \times k} \end{aligned} \quad (8)$$

where $\mathbf{D} = \gamma \hat{\mathbf{V}}^T + \gamma \hat{\mathbf{T}}^T + 2k \mathbf{V}^T \mathbf{S}^\Phi + 2k \mathbf{T}^T \mathbf{S}^\Phi + 2\beta \mathbf{W} \mathbf{L}^T$; $\hat{\mathbf{V}} = [\hat{v}_1, \hat{v}_2, \dots, \hat{v}_n]^T$; $\hat{\mathbf{T}} = [\hat{t}_1, \hat{t}_2, \dots, \hat{t}_n]^T$, and \hat{v}_i, \hat{t}_i are respectively defined as follows:

$$\hat{v}_i = \begin{cases} \mathbf{v}_i, & \text{if } i \in \Phi, \\ \mathbf{0}, & \text{if } i \notin \Phi. \end{cases} \quad (9)$$

$$\hat{t}_i = \begin{cases} \mathbf{t}_i, & \text{if } i \in \Phi, \\ \mathbf{0}, & \text{if } i \notin \Phi. \end{cases} \quad (10)$$

The above Formula (8) is NP hard. Inspired by SDH [34], the binary codes \mathcal{B} can be learned by the discrete cyclic coordinate descent (DCC) method. It means that we directly learn hash codes \mathcal{B} bit by bit. Specifically, we update one column of \mathcal{B} with the other column fixed. We let \mathcal{B}_{*i} denotes the i^{th} column of \mathcal{B} , and $\hat{\mathcal{B}}_i$ denotes the matrix of \mathcal{B} without the column \mathcal{B}_{*i} ; Let \mathbf{V}_{*i} denotes the i^{th} column of \mathbf{V} , and $\hat{\mathbf{V}}_i$ denotes the matrix of \mathbf{V} without the column \mathbf{V}_{*i} ; Let \mathbf{T}_{*i} denotes the i^{th} column of \mathbf{T} , and $\hat{\mathbf{T}}_i$ denotes the matrix of \mathbf{T} without the column \mathbf{T}_{*i} ; Let \mathbf{W}_{i*} denotes the i^{th} row of \mathbf{W} , and $\hat{\mathbf{W}}_i$ denotes the matrix of \mathbf{W} without the row \mathbf{W}_{i*} ; Let \mathbf{D}_{i*} denotes the i^{th} row of \mathbf{D} , and

\widetilde{D}_i denotes the matrix of D without the row D_{i*} . Then we can optimize B_{*i} by the following function:

$$\begin{aligned} \min_{B_{*i}} \mathcal{L} &= \left\| \mathbf{V} \mathbf{B}^T \right\|_F^2 + \left\| \mathbf{T} \mathbf{B}^T \right\|_2^F + \beta \left\| \mathbf{B} \mathbf{W} \right\|_F^2 \\ &\quad - \text{tr}(\mathbf{B} \mathbf{D}) + \text{const} \\ &= \text{tr}(\mathbf{B}_{*i} (2\mathbf{V}_{*i}^T \widehat{\mathbf{V}}_i \widehat{\mathbf{B}}_i^T + 2\mathbf{T}_{*i}^T \widehat{\mathbf{T}}_i \widehat{\mathbf{B}}_i^T + 2\beta \mathbf{W}_{i*} \widetilde{\mathbf{W}}_i^T \widehat{\mathbf{B}}_i^T \\ &\quad - \widetilde{\mathbf{D}}_{i*})) + \text{const} \\ \text{s.t. } B_{*i} &\in \{-1, +1\}^{n \times k} \end{aligned} \quad (11)$$

Finally, we can get the optimal solution of Formula (11):

$$B_{*i} = -\text{sign}(2\widehat{\mathbf{B}}_i \widehat{\mathbf{V}}_i^T \mathbf{V}_{*i} + 2\widehat{\mathbf{B}}_i \widehat{\mathbf{T}}_i^T \mathbf{T}_{*i} + 2\beta \widehat{\mathbf{B}}_i \widetilde{\mathbf{W}}_i \mathbf{W}_{i*}^T - \widetilde{\mathbf{D}}_{i*}^T) \quad (12)$$

then we can use Formula (12) to update B bit by bit.

3.4.4 Learn W with Θ , Ψ and B fixed

When Θ , Ψ and B are fixed, we can reformulate Formula (4) as follows:

$$\begin{aligned} \min_{\mathbf{W}} \mathcal{L} &= \alpha \left(\left\| \mathbf{V} \mathbf{W} - \mathbf{L}^\Phi \right\|_F^2 + \left\| \mathbf{T} \mathbf{W} - \mathbf{L}^\Phi \right\|_F^2 \right) \\ &\quad + \beta \left\| \mathbf{B} \mathbf{W} - \mathbf{L} \right\|_F^2 + \eta \left\| \mathbf{W} \right\|_F^2 \end{aligned} \quad (13)$$

For Formula (13), it is easy to solve W by the regularized least squares problem, which has a closed-form solution:

$$\mathbf{W} = (\alpha \mathbf{V}^T \mathbf{V} + \alpha \mathbf{T}^T \mathbf{T} + \beta \mathbf{B}^T \mathbf{B} + \eta \mathbf{I})^{-1} (\alpha \mathring{\mathbf{V}} + \alpha \mathring{\mathbf{T}} + \beta \mathbf{B})^T \mathbf{L} \quad (14)$$

3.5 Out-of-Sample Extension

For any instance which is not in the retrieval set, we can obtain the hash code of its two modalities. In particular, given the image modality \mathbf{x}_q in an instance \mathbf{o}_q , we can adopt forward propagation to generate the hash code as follows:

$$\mathbf{h}_q = \text{sign}(\mathcal{F}(\mathbf{x}_i; \Theta)) \quad (15)$$

Similarly, we can also use the text hashing network to generate the hash code of the instance \mathbf{o}_q with only textual modality \mathbf{y}_q :

$$\mathbf{g}_q = \text{sign}(\mathcal{P}(\mathbf{y}_i; \Psi)) \quad (16)$$

4 EXPERIMENTS

To evaluate the performance of DCHUC, we will carry out extensive experiments on three image-text datasets and compared it with seven state-of-the-art cross-modal hashing methods.

4.1 Datasets

Three datasets are used for evaluation, i.e., *MIRFLICKR-25K* [35], *IAPR TC-12* [36] and *NUS-WIDE* [37], which are described below.

The *MIRFLICKR-25K* dataset [35] contains 25,000 instances collected from Flickr website. Each image is labeled with several textual tags. Here, we follow the experimental protocols given in DCMH [32]. In total, 20,015 data instances which have at least 20 textual tags have been selected for our experiment. The text modality for each instance is represented as a 1,386-dimensional bag-of-words (BoW) vector. Furthermore each instance is manually labeled with at least one of the 24 unique labels. For this dataset, we randomly sampled 2,000 instances as the test set, and the remaining as the database (retrieval set). Furthermore, the training

phase of the existing deep cross-modal hashing methods are typically time-consuming, which makes them cannot efficiently work on large-scale datasets. Therefore, for deep methods, we randomly sample 10,000 instances from the retrieval set as the training set.

The *IAPR TC-12* [36] consists of 20,000 instances which are annotated using 255 labels. After pruning the instance that is without any text information, a subset of 19999 image-text pairs are select for our experiment. The text modality for each instance is represented as a 2000-dimensional BoW vector. For this dataset, we randomly sampled 2,000 instances as test set, with the rest of the instances as retrieval set. We randomly select 10,000 instances from retrieval set for training deep cross-modal baselines.

The *NUS-WIDE* dataset [37] contains 269,648 instances crawled from Flickr. Each image is associated with textual tags, and each instance is annotated with one or multiple labels from 81 concept labels. Only 195,834 image-text pairs that belong to the 21 most frequent concepts are selected for our experiment. The text modality for each instance is represented as a 1000-dimensional BoW vector. For this dataset, we randomly sampled 2,100 instance as test set, with the rest of the instances as retrieval set. Because the deep hashing baselines are very time-consuming for training, we randomly select 10,500 instances from database for training deep cross-modal baselines.

For all the shallow cross-modal baselines, all the database are used for training. For all datasets, the image \mathbf{x}_i and text \mathbf{y}_j will be defined as a similar pair if \mathbf{x}_i and \mathbf{y}_j share at least one common label. Otherwise, they will be defined as a dissimilar pair.

4.2 Baselines and Implementation Details

We compare our DCHUC with seven state-of-the-art methods, including four shallow cross-modal hashing methods, i.e., DLFH [19], SCM [28], CCA-ITQ [38] and DCH [31], and three deep cross-modal hashing methods, i.e., DCMH [32], CMDVH [11] and SSAH [22]. The source codes of all baselines but CMDVH and DCH are kindly provided by the authors. We carefully tuned their parameters according to the scheme suggested by the authors. For CMDVH and DCH, we implement it carefully by ourselves. In order to make a fair comparison, we utilize Alexnet [33], which has been pretrained on the ImageNet dataset [39] to extract deep features as the image inputs of all shallow cross-modal baselines, and the input for image modality hashing network of each deep cross-modal baseline is the 224×224 raw pixels.

For the proposed method, we initialize the first seven layers neural network in image feature learning part with the pre-trained Alexnet [33] model on ImageNet [39]. All the parameters of the text modal hashing network and the hashing layer of image hashing network are initialized by Xavier initialization [40]. The unified binary code B is initialized randomly and zero-centered. The image input is the 224×224 raw pixels, and the text inputs are the BoW vectors. The hyper-parameters $\alpha, \gamma, \beta, \mu, \eta$ in DCHUC are empirically set as 50, 200, 1, 50, 50, respectively, and they will be discussed in Section 4.7. We set $t_{out} = 30$, $t_{in} = 3$, $|\Phi| = 2000$ by using a validation strategy for all datasets. We adopt SGD with a mini-batch size of 64 as our optimization algorithm. The learning rate is initialized as 0.0001 for image hashing network and 0.004 for text modal hashing network. To avoid effect caused by class-imbalance problem between positive and negative similarity information, we empirically set the weight of the element " - 1" in S as the ratio between the number of element "1" and the number of element " - 1" in S .

TABLE 1

MAP. The best accuracy is shown in boldface and the second best accuracy is underlined. The baselines are based on Alexnet features

Task	Method	MIRFLICKR-25K				IAPR TC-12				NUS-WIDE			
		16bits	32bits	48bits	64bits	16bits	32bits	48bits	64bits	16bits	32bits	48bits	64bits
$T \rightarrow I$	CCA-ITQ	0.599	0.587	0.582	0.578	0.403	0.399	0.396	0.390	0.426	0.415	0.410	0.401
	SCM	0.639	0.612	0.584	0.592	0.438	0.423	0.414	0.398	0.403	0.371	0.349	0.328
	DCH	0.759	0.780	0.793	0.794	0.536	0.559	0.564	0.582	<u>0.619</u>	<u>0.652</u>	<u>0.653</u>	<u>0.681</u>
	DLFH	0.769	0.796	0.805	0.809	0.470	0.498	0.516	0.555	0.599	0.608	0.619	0.630
	DCMH	0.763	0.771	0.771	0.779	0.511	0.525	0.527	0.535	0.629	0.642	0.652	0.662
	CMDVH	0.612	0.610	0.553	0.600	0.381	0.383	0.396	0.381	0.371	0.359	0.399	0.424
	SSAH	<u>0.783</u>	0.793	0.800	0.783	0.538	0.566	0.580	0.586	0.613	0.632	0.635	0.633
	DCHUC	0.850	0.857	0.853	0.854	0.615	0.666	0.681	0.693	0.698	0.728	0.742	0.749
$I \rightarrow T$	CCA-ITQ	0.593	0.582	0.577	0.574	0.312	0.311	0.310	0.309	0.424	0.412	0.398	0.387
	SCM	0.626	0.595	0.588	0.578	0.313	0.310	0.309	0.308	0.395	0.368	0.353	0.335
	DCH	0.748	0.786	0.799	0.805	0.486	0.486	0.496	0.502	0.648	0.678	0.699	0.708
	DLFH	0.719	0.732	0.742	0.748	0.417	0.451	0.484	0.490	0.558	0.578	0.591	0.593
	DCMH	0.721	0.733	0.729	0.742	0.464	0.485	0.490	0.498	0.588	0.607	0.615	0.632
	CMDVH	0.611	0.626	0.553	0.598	0.376	0.373	0.365	0.376	0.370	0.373	0.414	0.425
	SSAH	<u>0.779</u>	<u>0.789</u>	0.796	0.794	<u>0.539</u>	<u>0.564</u>	<u>0.581</u>	<u>0.587</u>	<u>0.659</u>	0.666	0.679	0.667
	DCHUC	0.878	0.882	0.880	0.881	0.630	0.677	0.695	0.701	0.750	0.771	0.783	0.791

TABLE 2

Precision@1000. The best accuracy is shown in boldface and the second best accuracy is underlined. The baselines are based on Alexnet features

Task	Method	MIRFLICKR-25K				IAPR TC-12				NUS-WIDE			
		16bits	32bits	48bits	64bits	16bits	32bits	48bits	64bits	16bits	32bits	48bits	64bits
$T \rightarrow I$	CCA-ITQ	0.690	0.676	0.666	0.652	0.491	0.492	0.488	0.482	0.622	0.672	0.684	0.683
	SCM	0.749	0.714	0.675	0.639	0.504	0.506	0.523	0.497	0.598	0.576	0.532	0.668
	DCH	<u>0.848</u>	0.848	0.843	0.852	<u>0.664</u>	<u>0.695</u>	<u>0.701</u>	<u>0.712</u>	0.808	<u>0.819</u>	<u>0.808</u>	<u>0.815</u>
	DLFH	0.834	<u>0.857</u>	<u>0.865</u>	0.870	0.563	0.604	0.638	0.660	0.685	0.707	0.717	0.735
	DCMH	0.815	0.824	0.834	0.835	0.596	0.610	0.613	0.626	0.694	0.710	0.721	0.731
	CMDVH	0.613	0.636	0.545	0.601	0.396	0.410	0.403	0.396	0.340	0.293	0.408	0.417
	SSAH	0.824	0.834	0.846	0.855	0.641	0.664	0.674	0.677	0.701	0.729	0.736	0.731
	DCHUC	0.896	0.897	0.890	0.888	0.711	0.760	0.771	0.782	<u>0.799</u>	0.825	0.839	0.849
$I \rightarrow T$	CCA-ITQ	0.666	0.656	0.649	0.635	0.401	0.341	0.302	0.302	0.607	0.657	0.667	0.666
	SCM	0.738	0.704	0.676	0.660	0.376	0.349	0.324	0.315	0.606	0.565	0.550	0.504
	DCH	0.844	0.866	0.860	0.868	0.593	0.604	0.612	0.617	0.813	0.829	0.822	0.817
	DLFH	0.800	0.817	0.824	0.825	0.480	0.536	0.584	0.596	0.646	0.682	0.703	0.698
	DCMH	0.764	0.795	0.817	0.822	0.546	0.572	0.580	0.595	0.667	0.686	0.704	0.709
	CMDVH	0.693	0.761	0.695	0.733	0.371	0.380	0.331	0.371	0.493	0.527	0.598	0.589
	SSAH	0.840	0.854	0.859	0.863	<u>0.648</u>	<u>0.663</u>	<u>0.681</u>	<u>0.678</u>	0.738	0.749	0.765	0.749
	DCHUC	0.917	0.918	0.912	0.911	0.724	0.766	0.781	0.783	0.845	0.859	0.872	0.881

The source codes of CMDVH, DCH and our proposed method will be available at: <https://github.com/Academic-Hammer>

4.3 Evaluation Protocol

For hashing-based cross-modal retrieval task, Hamming ranking and hash lookup are two widely used retrieval protocols to evaluate the performance of hashing methods. In our experiments, we use three evaluation criterions: the mean average precision (MAP), the precision at n (P@n) and the precision-recall (PR) curve. MAP is the widely used metric to measure the accuracy of the Hamming ranking protocol, which is defined as the mean of average precision for all queries. PR curve is used to evaluate the accuracy of the hash lookup protocol, and P@n is used to evaluate precision by considering only the number of top returned points.

4.4 Experimental results

All experiments are run 3 times to reduce randomness, then the average accuracy is reported.

4.4.1 Hamming Ranking Task

Table 1 and Table 2 present the MAP and Precision@1000 on MIRFLICKR-25K, IAPR TC-12 and NUS-WIDE datasets, respectively. " $I \rightarrow T$ " denotes retrieving texts with image queries, and " $T \rightarrow I$ " denotes retrieving images with text queries. In general, from Table 1 and Table 2, we have three observations: (i) Our proposed method can outperforms the other cross-modal hashing methods for different length of hash code. For example, on MIRFLICKR-25K, comparing with the best competitor SSAH on 16-bits, the results of DCHUC for " $I \rightarrow T$ " have a relative increase of 12.7% on MAP and 9.2% on Precision@1000; the results of DCHUC for " $T \rightarrow I$ " have a relative increase of 8.6% on MAP and 8.7% on Precision@1000. On IAPR TC-12, comparing with the competitor SSAH on 64-bits, the results of DCHUC for " $I \rightarrow T$ " have a relative increase of 19.4% on MAP and 15.5% on Precision@1000; the results of DCHUC for " $T \rightarrow I$ " have a relative increase of 18.3% on MAP and 15.5% on Precision@1000. On NUS-WIDE, comparing with the best competitor DCH on 64-bits, the results of DCHUC for " $I \rightarrow T$ " have a relative increase of 12.3% on MAP and 7.8% on Precision@1000; (ii) Integrating the feature learning of data-

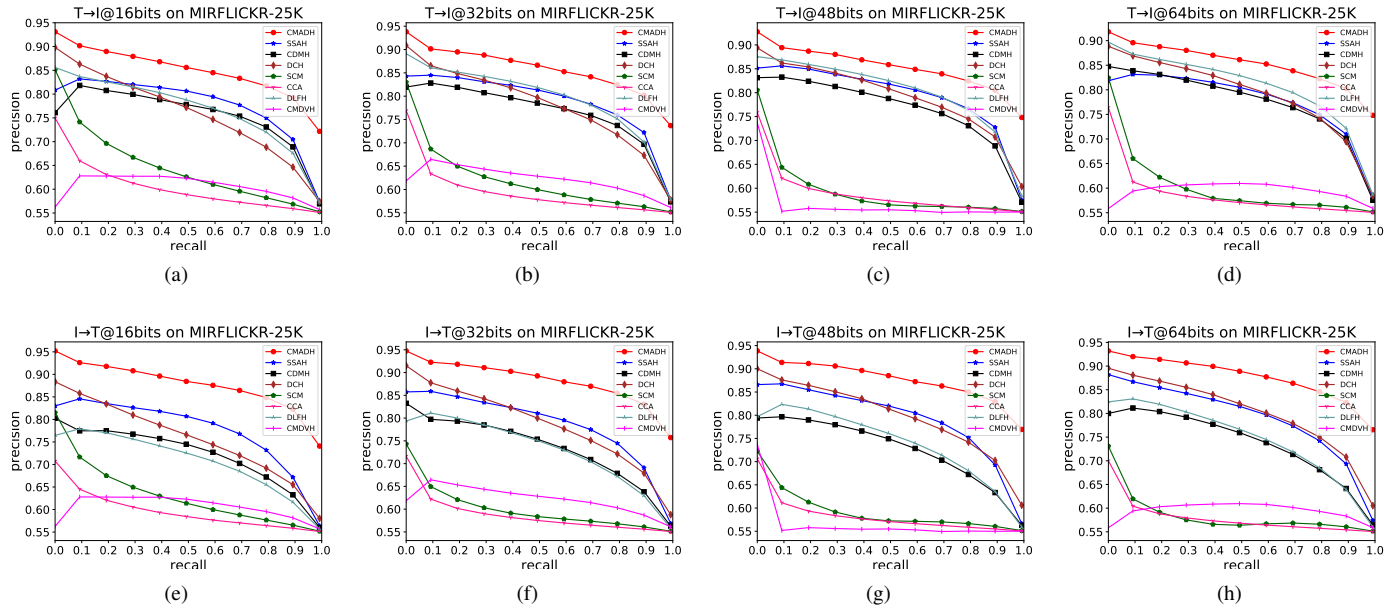


Fig. 2. Precision-recall curve on MIRFLICKR-25K dataset

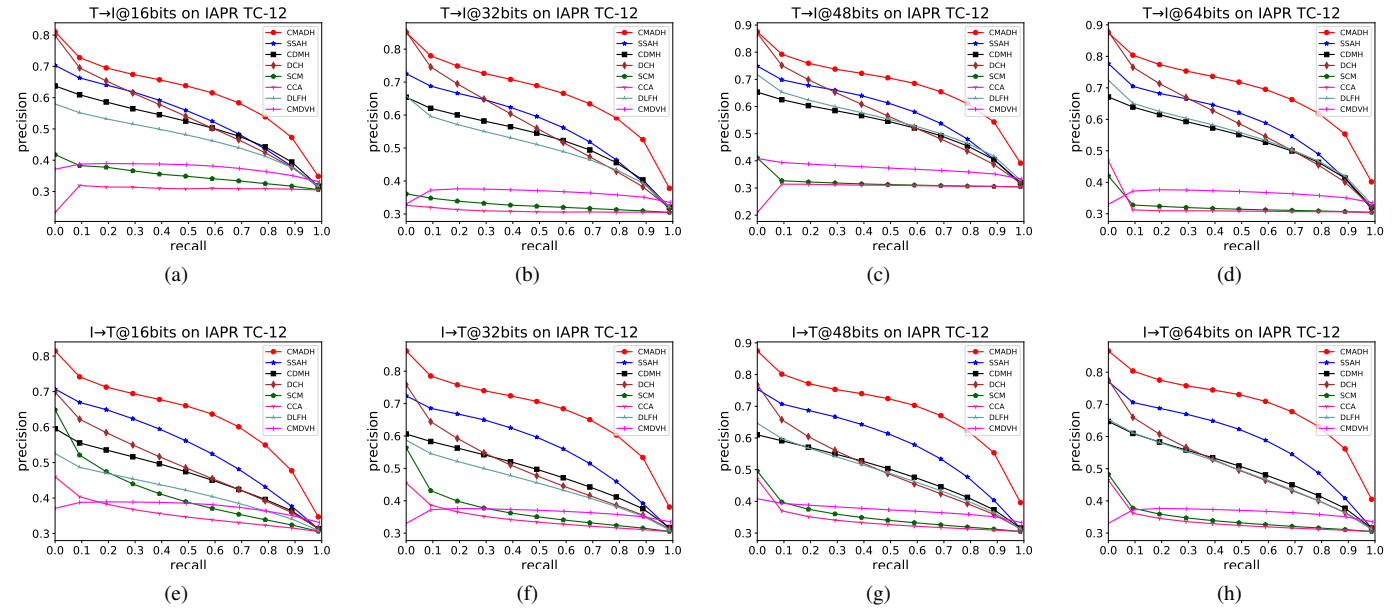


Fig. 3. Precision-recall curve on IAPR TC-12 dataset

points and hashing function learning into an end-to-end network can get the better performance. For example, our proposed method can get a better performance than DCH which also can jointly learning unified hashing codes for instances in the database and modal-specific hashing functions for unseen data-points but the feature extraction procedure is independent of the hash codes learning procedure. (iii) Jointly learning unified hashing codes for database instances and modality-specific hashing functions for unseen data-points can greatly increase the retrieval performance. For instance, DCHUC can get better performance on MAP and Precision@1000 over three benchmark datasets than CMDVH. Note that, the results of CMDVH is not as good as the results of the original article. It maybe the reason that we used more classes

of label to carry out our experimental, which is hard to train the svm used in CMDVH. Furthermore, although DCH is a shallow hashing method, its retrieval performances on MIRFLICKR-25K and IAPR TC-12 datasets are similar to the best deep baseline SSAH, and its retrieval performances on NUS-WIDE dataset is batter than SSAH.

4.4.2 Hash Lookup Task

When considering the lookup protocol, we compute the precision and recall (PR) curve for the returned points given any Hamming radius. The PR curve can be obtained by varying the Hamming radius from 0 to k with a step-size of 1. Fig. 2, Fig. 3 and Fig. 4 show the precision-recall curve on MIRFLICKR-25K,

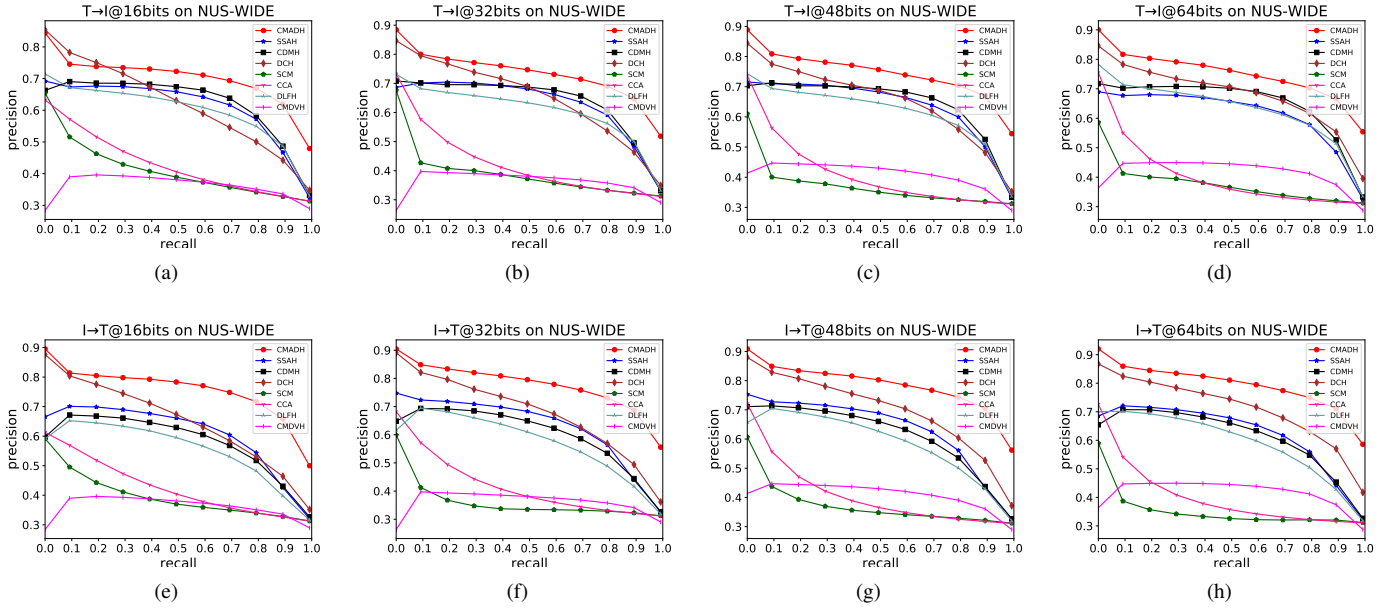


Fig. 4. Precision-recall curves on NUS-WIDE dataset

IAPR TC-12 and NUS-WIDE datasets, respectively. It is easy to find that DCHUC can dramatically outperform the state-of-the-art baselines, which means our DCHUC generates hash codes for similar points in a small Hamming radius. For example, compared with baselines, the precision value of DCHUC decreases more slowly with the recall value increasing, and DCHUC can get a high precision value even though the recall value increasing to 0.9 on MIRFLICKR-25K and NUS-WIDE datasets.

4.5 Convergence Analysis

To verify the convergence property of DCHUC, we conduct an experiment over NUS-WIDE dataset with the code length being 64. Fig. 5 shows the convergence of objective function value and MAP. As shown in Fig. 5 (a), the objective function value can convergence after only 10 iterations. In Fig. 5 (b), " $I \rightarrow T$ " denotes retrieving texts with image queries, and " $T \rightarrow I$ " denotes retrieving images with text queries. We can find the MAP values of both the two retrieval task can convergence. Furthermore, combining the two subfigure Fig. 5 (a) and (b), we can find both the two map values can increase with the objective function value decrease and eventually converge.

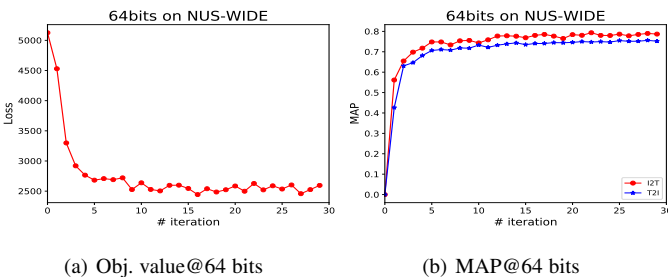


Fig. 5. Objective function value and MAP of DCHUC over NUS-WIDE on 64 bits.

4.6 Training efficiency

To evaluate the training speed of DCHUC, we conduct experiments between the deep cross-modal baselines except CMDVH on three datasets. Fig. 6 shows the variation between MAP and training time on the three datasets for DCHUC, SSAH and DCMH. It can be find that DCHUC can not only training faster than the two deep cross-modal baselines, but also get a better performance on retrieval tasks than them. For the CMDVH baseline, it is a two step method. Then it is unfair to compare MAP-Time curve. In here, we calculate the the whole training time of CMDVH. The cost times of training phase on IAPR TC-12, MIRFLICKR-25K and NUS-WIDE datasets with 32-bits are 16.3s, 21.2s and 39.2s for CMDVH, and are 11.8s, 12.9s and 28.2s for DCHUC, respectively. We can find that DCHUC is also the faster one.

4.7 Sensitivity to Parameters

We study the influence of the hyper-parameters $\alpha, \gamma, \beta, \eta$ and μ on IAPR TC-12, MIRFLICKR-25K and NUS-WIDE datasets with the code length being 64-bits. More specially, Fig. 7 (a), (f) and (k) show the affect of the hyper-parameter α over the three datasets with the value between 1 and 600. Fig. 7 (b), (g) and (i) show the affect of the hyper-parameter γ over the three datasets with the value between 1 and 600. Fig. 7 (c), (h) and (m) show the affect of the hyper-parameter β over the three datasets with the value between 10^{-3} and 10. Fig. 7 (d), (i) and (n) show the affect of the hyper-parameter η over the three datasets with the value between 1 and 600. Fig. 7 (e), (j) and (o) show the affect of the hyper-parameter μ over the three datasets with the value between 1 and 600. It can be found that DCHUC is not sensitive to $\alpha, \gamma, \beta, \eta$ and μ . For instance, DCHUC can achieve good performance on all the three datasets in the range of 1 to 600 for the hyper-parameters α, γ and η , and also can achieve good performance on all the three datasets with $1 \leq \beta \leq 300$. Furthermore, DCHUC can get the high MAP values with different β from the range of 10^{-3} to 10.

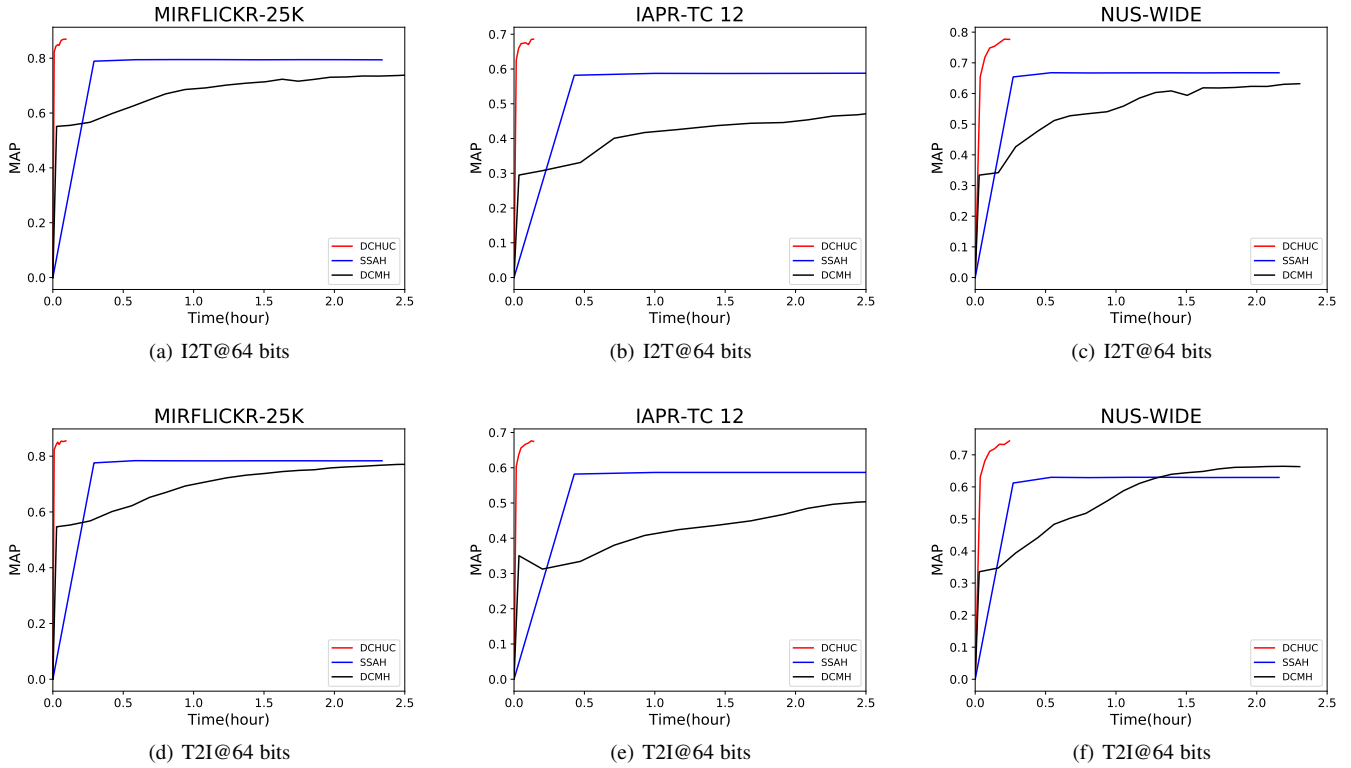


Fig. 6. Training Efficiency of DCHUC, SSAH and DCMH on Three Datasets.

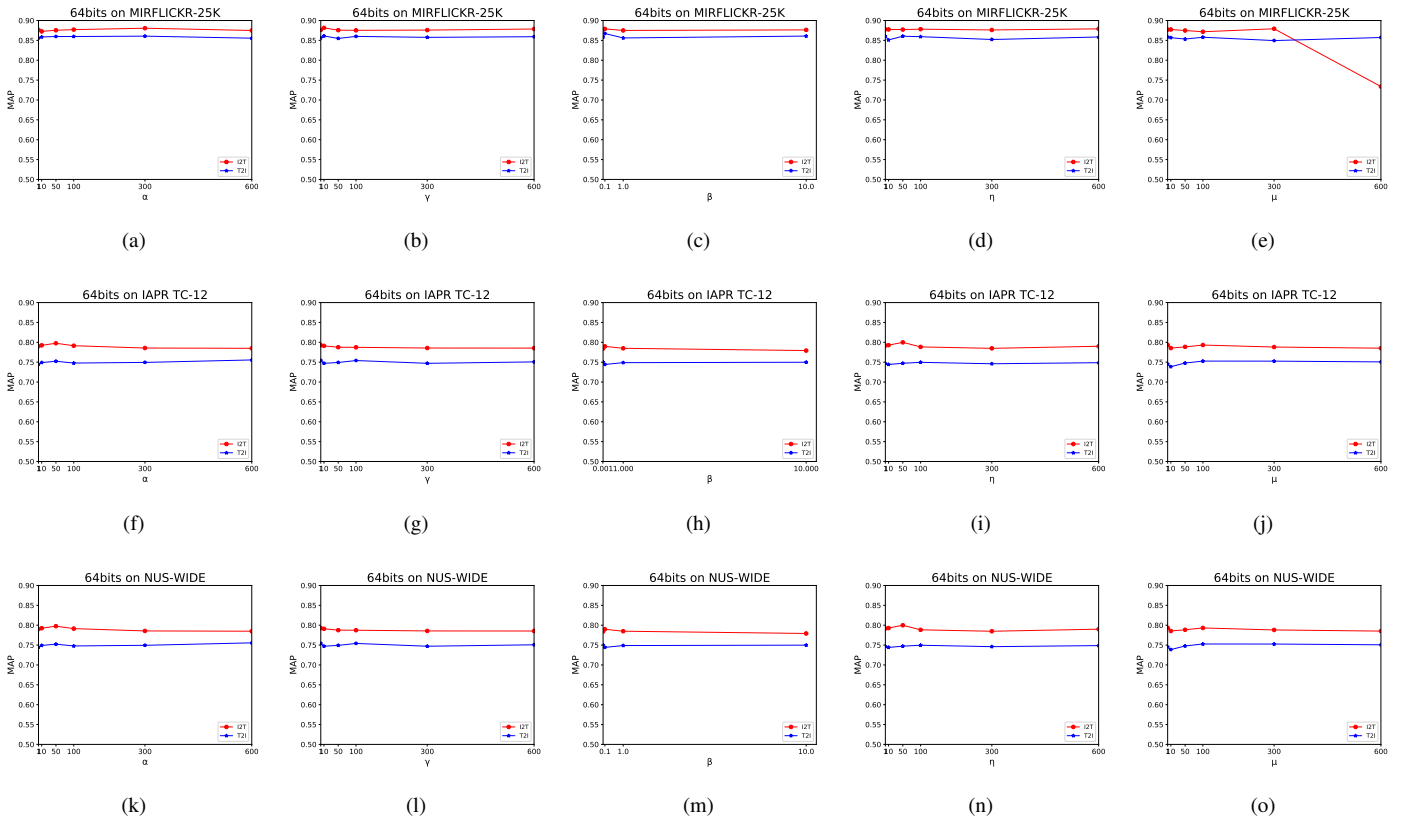


Fig. 7. MAP values with different parameters on three datasets.

5 CONCLUSION

In this paper, we have proposed a novel cross-modal deep hashing method for cross-modal data, called DCHUC. To the best of our knowledge, DCHUC is the first deep method to jointly learn unified hash codes for database instances and hashing functions for unseen query points in an end-to-end framework. Extensive experiments on three real-world public datasets have shown that the proposed DCHUC method outperforms the state-of-the-art cross-modal hashing methods.

ACKNOWLEDGMENT

The work is supported by SFSMBRP(2018YFB1005100), BIGKE(No. 20160754021), NSFC (No. 61772076 and 61751201), NSFB (No. Z181100008918002), Major Project of Zhijiang Lab (No. 2019DH0ZX01), CETC (No. w-2018018) and OPBKLICDD (NO. ICDD201901).

REFERENCES

- [1] Z. Cao, M. Long, J. Wang, and S. Y. Philip, "Hashnet: Deep learning to hash by continuation," in *JCCV*, 2017, pp. 5609–5618.
- [2] X. Liu, X. Nie, W. Zeng, C. Cui, L. Zhu, and Y. Yin, "Fast discrete cross-modal hashing with regressing from semantic labels," in *2018 ACM Multimedia Conference on Multimedia Conference*. ACM, 2018, pp. 1662–1669.
- [3] T. Zhang and J. Wang, "Collaborative quantization for cross-modal similarity search," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2036–2045.
- [4] M. M. Bronstein, A. M. Bronstein, F. Michel, and N. Paragios, "Data fusion through cross-modality metric learning using similarity-sensitive hashing," in *2010 IEEE computer society conference on computer vision and pattern recognition*. IEEE, 2010, pp. 3594–3601.
- [5] Z. Jin, C. Li, Y. Lin, and D. Cai, "Density sensitive hashing," *IEEE Trans. Cybernetics*, vol. 44, no. 8, pp. 1362–1371, 2014.
- [6] S. Huang, Y. Xiong, Y. Zhang, and J. Wang, "Unsupervised triplet hashing for fast image retrieval," in *Proceedings of the on the Thematic Workshops of ACM Multimedia 2017*. ACM, 2017, pp. 84–92.
- [7] K. Ghasedi Dizaji, F. Zheng, N. Sadoughi, Y. Yang, C. Deng, and H. Huang, "Unsupervised deep generative adversarial hashing network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 3664–3673.
- [8] D. Wang, H. Huang, C. Lu, B.-S. Feng, L. Nie, G. Wen, and X.-L. Mao, "Supervised deep hashing for hierarchical labeled data," pp. 7388–7395, 2018.
- [9] Z. Qiu, Y. Pan, T. Yao, and T. Mei, "Deep semantic hashing with generative adversarial networks," in *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 2017, pp. 225–234.
- [10] H. Liu, R. Ji, Y. Wu, F. Huang, and B. Zhang, "Cross-modality binary code learning via fusion similarity hashing," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 7380–7388.
- [11] V. Erin Liong, J. Lu, Y.-P. Tan, and J. Zhou, "Cross-modal deep variational hashing," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 4077–4085.
- [12] F. Feng, X. Wang, and R. Li, "Cross-modal retrieval with correspondence autoencoder," in *Proceedings of the 22nd ACM international conference on Multimedia*. ACM, 2014, pp. 7–16.
- [13] C. Deng, X. Tang, J. Yan, W. Liu, and X. Gao, "Discriminative dictionary learning with common label alignment for cross-modal retrieval," *IEEE Transactions on Multimedia*, vol. 18, no. 2, pp. 208–218, 2016.
- [14] B. Wang, Y. Yang, X. Xu, A. Hanjalic, and H. T. Shen, "Adversarial cross-modal retrieval," in *Proceedings of the 25th ACM international conference on Multimedia*. ACM, 2017, pp. 154–162.
- [15] E. Yang, C. Deng, W. Liu, X. Liu, D. Tao, and X. Gao, "Pairwise relationship guided deep hashing for cross-modal retrieval," in *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.
- [16] W. Liu, C. Mu, S. Kumar, and S.-F. Chang, "Discrete graph hashing," in *Advances in neural information processing systems*, 2014, pp. 3419–3427.
- [17] W. Liu, J. Wang, R. Ji, Y.-G. Jiang, and S.-F. Chang, "Supervised hashing with kernels," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2012, pp. 2074–2081.
- [18] X. Luo, X.-Y. Yin, L. Nie, X. Song, Y. Wang, and X.-S. Xu, "Sdmch: Supervised discrete manifold-embedded cross-modal hashing," in *IJCAI*, 2018, pp. 2518–2524.
- [19] Q.-Y. Jiang and W.-J. Li, "Discrete latent factor model for cross-modal hashing," *IEEE Transactions on Image Processing*, 2019.
- [20] Y. Cao, M. Long, J. Wang, and H. Zhu, "Correlation autoencoder hashing for supervised cross-modal search," in *Proceedings of the 2016 ACM on International Conference on Multimedia Retrieval*. ACM, 2016, pp. 197–204.
- [21] Y. Cao, M. Long, J. Wang, Q. Yang, and P. S. Yu, "Deep visual-semantic hashing for cross-modal retrieval," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2016, pp. 1445–1454.
- [22] C. Li, C. Deng, N. Li, W. Liu, X. Gao, and D. Tao, "Self-supervised adversarial hashing networks for cross-modal retrieval," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4242–4251.
- [23] Y. Shen, L. Liu, L. Shao, and J. Song, "Deep binaries: Encoding semantic-rich cues for efficient textual-visual cross retrieval," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 4097–4106.
- [24] N. Rasiwasia, J. Costa Pereira, E. Coviello, G. Doyle, G. R. Lanckriet, R. Levy, and N. Vasconcelos, "A new approach to cross-modal multimedia retrieval," in *Proceedings of the 18th ACM international conference on Multimedia*. ACM, 2010, pp. 251–260.
- [25] J. Zhou, G. Ding, and Y. Guo, "Latent semantic sparse hashing for cross-modal similarity search," in *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*. ACM, 2014, pp. 415–424.
- [26] G. Ding, Y. Guo, and J. Zhou, "Collective matrix factorization hashing for multimodal data," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 2075–2082.
- [27] Q.-Y. Jiang and W.-J. Li, "Asymmetric deep supervised hashing," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.
- [28] D. Zhang and W.-J. Li, "Large-scale supervised multimodal hashing with semantic correlation maximization," in *Twenty-Eighth AAAI Conference on Artificial Intelligence*, 2014.
- [29] S. Kumar and R. Udupa, "Learning hash functions for cross-view similarity search," in *Twenty-Second International Joint Conference on Artificial Intelligence*, 2011.
- [30] Z. Lin, G. Ding, M. Hu, and 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.
- [31] X. Xu, F. Shen, Y. Yang, H. T. Shen, and X. Li, "Learning discriminative binary codes for large-scale cross-modal retrieval," *IEEE Transactions on Image Processing*, vol. 26, no. 5, pp. 2494–2507, 2017.
- [32] Q.-Y. Jiang and W.-J. Li, "Deep cross-modal hashing," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 3232–3240.
- [33] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.
- [34] F. Shen, C. Shen, W. Liu, and H. Tao Shen, "Supervised discrete hashing," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 37–45.
- [35] M. J. Huiskes and M. S. Lew, "The mir flickr retrieval evaluation," in *Proceedings of the 1st ACM international conference on Multimedia information retrieval*. ACM, 2008, pp. 39–43.
- [36] H. J. Escalante, C. A. Hernández, J. A. Gonzalez, A. López-López, M. Montes, E. F. Morales, L. E. Sucar, L. Villaseñor, and M. Grubinger, "The segmented and annotated iapr tc-12 benchmark," *Computer Vision and Image Understanding*, vol. 114, no. 4, pp. 419–428, 2010.
- [37] T.-S. Chua, J. Tang, R. Hong, H. Li, Z. Luo, and Y. Zheng, "Nus-wide: a real-world web image database from national university of singapore," in *Proceedings of the ACM international conference on image and video retrieval*. ACM, 2009, p. 48.
- [38] Y. Gong, S. Lazebnik, A. Gordo, and F. Perronnin, "Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 12, pp. 2916–2929, 2013.
- [39] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet large scale visual recognition challenge," *International journal of computer vision*, vol. 115, no. 3, pp. 211–252, 2015.

- [40] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, 2010, pp. 249–256.



Rong-Cheng Tu received the bachelor's degree from Beijing Institute of Technology, Chain, in 2018. He is currently working toward the master's degree in the Department of Computer Science and Technology, Beijing Institute of Technology, Chain. His research interests are in deep learning and learning to hash.



Xian-Ling Mao received the Ph.D. science degree from Peking University, Chain, in 2012. He is currently an Associate Professor with the Department of Computer Science and Technology, Beijing Institute of Technology, Chain. His major research interests include deep learning, machine learning, information retrieval, natural language processing, artificial intelligence and network data mining.



Wei wei received the Ph.D. degree from the Huazhong University of Science and Technology, Chain, in 2012. He is currently an Associate Professor with School of Computer Science and Technology and the Director of Cognitive Computing and Intelligent Information Processing (CCIIP) Laboratory in Huazhong University of Science and Technology, Chain. His major research interests include information retrieval, natural language processing, artificial intelligence, data mining (text mining), statistics

machine learning, social media analysis and mining recommender system.



Heyan Huang received the bachelor's degree from Wuhan University of Surveying and Mapping, Chain, in 1983, the master's degree from National University of Defense Technology, Chain, in 1986, and the Ph.D. degree from the Institute of Computing Technology, Chinese Academy of Sciences, Chain, in 1989. She is currently a professor and the Dean with the Department of Computer Science and Technology, Beijing Institute of Technology, Chain. Her major research interests include natural language processing, information content security, intelligent application system.

processing, information content security, intelligent application system.