Deep Cross-Modal Hashing

Qing-Yuan Jiang and Wu-Jun Li
National Key Laboratory for Novel Software Technology
Collaborative Innovation Center of Novel Software Technology and Industrialization
Department of Computer Science and Technology, Nanjing University, P. R. China
jiangqy@lamda.nju.edu.cn, liwujun@nju.edu.cn

Abstract

Due to its low storage cost and fast query speed, cross-modal hashing (CMH) has been widely used for similarity search in multimedia retrieval applications. However, most existing CMH methods are based on hand-crafted features which might not be optimally compatible with the hash-code learning procedure. As a result, existing CMH methods with hand-crafted features may not achieve satisfactory performance. In this paper, we propose a novel CMH method, called deep cross-modal hashing (DCMH), by integrating feature learning and hash-code learning into the same framework. DCMH is an end-to-end learning framework with deep neural networks, one for each modality, to perform feature learning from scratch. Experiments on three real datasets with image-text modalities show that DCMH can outperform other baselines to achieve the state-of-the-art performance in cross-modal retrieval applications.

1. Introduction

Approximate nearest neighbor (ANN) search [1] plays a fundamental role in machine learning and related applications like information retrieval. Due to its low storage cost and fast retrieval speed, hashing has recently attracted much attention from the ANN research community [17, 34, 9, 15, 26, 10, 29, 21, 28, 4]. The goal of hashing is to map the data points from the original space into a Hamming space of binary codes where the similarity in the original space is preserved in the Hamming space. By using binary hash codes to represent the original data, the storage cost can be dramatically reduced. Furthermore, we can achieve a constant or sub-linear time complexity for search by using hash codes to construct an index [15]. Hence, hashing has become more and more popular for ANN search in large-scale datasets.

In many applications, the data can have multi-modalities. For example, besides the image content, there also exists text information like tags for the images in Flickr and many other social websites. This kind of data is always called multi-modal data. With the rapid growth of multi-modal data in real applications, especially multimedia applications, multi-modal hashing (MMH) has recently been widely used for ANN search (retrieval) on multi-modal datasets.

Existing MMH methods can be divided into two main categories: *mutli-source hashing* (MSH) [30, 36, 32, 14] and *cross-modal hashing* (CMH) [18, 35, 7, 22, 3]. The goal of MSH is to learn hash codes by utilizing all the information from multiple modalities. Hence, MSH requires that all the modalities should be observed for all data points including query points and those in database. In practice, the application of MSH is limited because in many cases it is difficult to acquire all modalities of all data points. On the contrary, the application scenarios of CMH are more flexible than those of MSH. In CMH, the modality of a query point is different from the modality of the points in database. Furthermore, typically the query point has only one modality and the points in the database can have one or more modalities. For example, we can use text queries to retrieve images in the database, and we can also use image queries to retrieve texts in the database. Due to its wide application, CMH has gained more attention than MSH.

Many CMH methods have recently been proposed. Existing representative methods include cross modality similarity sensitive hashing (CMSSH) [2], cross view hashing (CVH) [18], multi-modal latent binary embedding (MLBE) [39], co-regularized hashing (CRH) [38], semantic correlation maximization (SCM) [35], collective matrix factorization hashing (CMFH) [7], semantic topic multi-modal hashing (STMH) [33] and semantics preserving hashing (SePH) [22]. Almost all these existing CMH methods are based on hand-crafted features. One shortcoming of these hand-crafted feature based methods is that the feature extraction procedure is independent of the hash-code learning procedure, which means that the hand-crafted features might not be optimally compatible with the hash-code learning procedure. Hence, these existing CMH methods with hand-crafted features may not achieve satisfactory performance in real applications.

Recently, deep learning with neural networks [19, 16] has been widely used to perform feature learning from scratch with promising performance. There also exist some methods which adopt deep learning for uni-modal hashing [37, 23, 20, 40, 24]. These methods show that end-to-end deep learning architecture is more compatible for hashing learning. For the CMH setting, there also appears one method, called deep visual-semantic hashing (DVSH) [3], with deep neural networks for feature learning[1]. However, DVSH can only be used for a special CMH case where one of the modalities have to be temporal dynamics.

In this paper, we propose a novel CMH method, called deep cross-modal hashing (DCMH), for cross-modal retrieval applications. The main contributions of DCMH are outlined as follows:

- DCMH is an end-to-end learning framework with deep neural networks, one for each modality, to perform feature learning from scratch.
- The hash-code learning problem is essentially a discrete learning problem, which is difficult to learn. Hence, most existing CMH methods solve this problem by relaxing the original discrete learning problem into a continuous learning problem. This relaxation procedure may deteriorate the accuracy of the learned hash codes [25]. Unlike these relaxation-based methods, DCMH directly learns the discrete hash codes without relaxation.
- Experiments on real datasets with image-text modalities show that DCMH can outperform other baselines to achieve the state-of-the-art performance in cross-modal retrieval applications.

The rest of this paper is organized as follows. Section 2 introduces the problem definition of this paper. We present our DCMH method in Section 3, including the model formulation and learning algorithm. Experiments are shown in Section 4. At last, we conclude our work in Section 5.

2. Problem Definition

2.1. Notation

Boldface lowercase letters like $\mathbf{w}$ are used to denote vectors. Boldface uppercase letters like $\mathbf{W}$ are used to denote matrices, and the element in the ith row and jth column of $\mathbf{W}$ is denoted as W_{ij}. The ith row of $\mathbf{W}$ is denoted as $\mathbf{W}_{i*}$, and the jth column of $\mathbf{W}$ is denoted as $\mathbf{W}_{*j}$. $\mathbf{W}^T$ is the transpose of $\mathbf{W}$. We use $\mathbf{1}$ to denote a vector with all elements being 1. $\mathrm{tr}(\cdot)$ and $\|\cdot\|_F$ denote the trace of a matrix and the Frobenius norm of a matrix, respectively. $\mathrm{sign}(\cdot)$ is an element-wise sign function defined as follows:

$$\mathrm{sign}(x) = \begin{cases} 1 & x \geq 0, \\ -1 & x < 0. \end{cases}$$

2.2. Cross-Modal Hashing

Although the method proposed in this paper can be easily adapted to cases with more than two modalities, we only focus on the case with two modalities here.

Assume that we have n training entities (data points), each of which has two modalities of features. Without loss of generality, we use image-text datasets for illustration in this paper, which means that each training point has both text modality and image modality. We use $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^n$ to denote the image modality, where $\mathbf{x}_i$ can be the hand-crafted features or the raw pixels of image i. Moreover, we use $\mathbf{Y} = \{\mathbf{y}_i\}_{i=1}^n$ to denote the text modality, where $\mathbf{y}_i$ is typically the tag information related to image i. In addition, we are also given a cross-modal similarity matrix $\mathbf{S}$. $S_{ij} = 1$ if image $\mathbf{x}_i$ and text $\mathbf{y}_j$ are similar, and $S_{ij} = 0$ otherwise. Here, the similarity is typically defined by some semantic information such as class labels. For example, we can say that image $\mathbf{x}_i$ and text $\mathbf{y}_j$ are similar if they share the same class label. Otherwise, image $\mathbf{x}_i$ and text $\mathbf{y}_j$ are dissimilar if they are from different classes.

Given the above training information $\mathbf{X}$, $\mathbf{Y}$ and $\mathbf{S}$, the goal of cross-modal hashing is to learn two hash functions for the two modalities: $h^{(x)}(\mathbf{x}) \in \{-1, +1\}^c$ for the image modality and $h^{(y)}(\mathbf{y}) \in \{-1, +1\}^c$ for the text modality, where c is the length of binary code. These two hash functions should preserve the *cross-modal similarity* in $\mathbf{S}$. More specifically, if $S_{ij} = 1$, the Hamming distance between the binary codes $\mathbf{b}_i^{(x)} = h^{(x)}(\mathbf{x}_i)$ and $\mathbf{b}_j^{(y)} = h^{(y)}(\mathbf{y}_j)$ should be small. Otherwise, if $S_{ij} = 0$, the corresponding Hamming distance should be large.

Here, we assume that both modalities of features for each point in the *training set* are observed although our method can also be easily adapted to other settings where some *training points* have only one modality of features being observed. Please note that we only make this assumption for training points. After we have trained the model, we can use the learned model to generate hash codes for query and database points of either one modality or two modalities, which exactly matches the setting of cross-modal retrieval applications.

[1]The first version of our DCMH method has been submitted to arXiv [13] before this CVPR submission, which actually appeared earlier than DVSH in public literature.

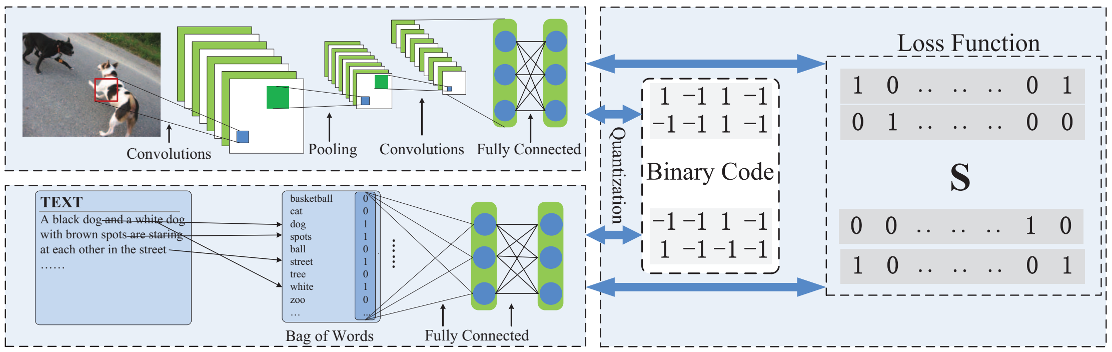


Figure 1. The end-to-end deep learning framework of our DCMH model.

Table 1. Configuration of the CNN for image modality.

Layer	Configuration
conv1	f. $64 \times 11 \times 11$; st. 4×4, pad 0, LRN,$\times 2$ pool
conv2	f. $265 \times 5 \times 5$; st. 1×1, pad 2, LRN,$\times 2$ pool
conv3	f. $265 \times 3 \times 3$; st. 1×1, pad 1
conv4	f. $265 \times 3 \times 3$; st. 1×1, pad 1
conv5	f. $265 \times 3 \times 3$; st. 1×1, pad 1,$\times 2$ pool
full6	4096
full7	4096
full8	Hash code length c

3. Deep Cross-Modal Hashing

In this section, we present the details about our deep CMH (DCMH) method, including model formulation and learning algorithm.

3.1. Model

The whole DCMH model is shown in Figure 1, which is an end-to-end learning framework by seamlessly integrating two parts: the feature learning part and the hash-code learning part. During learning, each part can give feedback to the other part.

3.1.1 Feature Learning Part

The feature learning part contains two deep neural network-s, one for image modality and the other for text modality.

The deep neural network for image modality is a convolutional neural network (CNN) adapted from [5]. There are eight layers in this CNN model. The first seven layers are the same as those in CNN-F of [5]. The eighth layer is a fully-connected layer with the output being the learned image features.

Table 1 shows the detailed configuration of the CN-N for image modality. More specifically, eight layers are divided into five convolutional layers and three fully-connected layers, which are denoted as "conv1 - conv5" and "full6 - full8" in Table 1, respectively. Each convolutional layer is described by several aspects:

- "f. $num \times size \times size$" denotes the number of convolution filters and their receptive field size.
- "st" denotes the convolution stride.
- "pad" denotes the number of pixels to add to each size of the input.
- "LRN" denotes whether Local Response Normalization (LRN) [16] is applied or not.
- "pool" denotes the down-sampling factor.
- The number in the fully connected layers, such as "4096", denotes the number of nodes in that layer. It is also the dimensionality of the output at that layer.

All the first seven layers use the Rectified Linear Unit (Re-LU) [16] as activation function. For the eighth layer, we choose identity function as the activation function.

To perform feature learning from text, we first represent each text $\mathbf{y}_j$ as a vector with bag-of-words (BOW) representation. And then the bag-of-words vectors are used as the input to a deep neural network with two fully-connected layers, denoted as "full1 - full2". The detailed configuration of the deep neural network for text is shown in Table 2, where the configuration shows the number of nodes in each layer. The activation function for the first layer is ReLU, and that for the second layer is the identity function.

Table 2. Configuration of the deep neural network for text modality.

Layer	Configuration
full1	8192
full2	Hash code length c

Please note that the main goal of this paper is to show that it is possible to design an end-to-end learning framework for cross-modal hashing by using deep neural networks for feature learning from scratch. But how to design different neural networks is not the focus of this paper. Other deep neural networks might also be used to perform feature learning for our DCMH model, which will be leaved for future study.

3.1.2 Hash-Code Learning Part

Let $f(\mathbf{x}_i;\theta_x) \in \mathbb{R}^c$ denote the learned image feature for point i, which corresponds to the output of the CNN for image modality. Furthermore, let $g(\mathbf{y}_j;\theta_y) \in \mathbb{R}^c$ denote the learned text feature for point j, which corresponds to the output of the deep neural network for text modality. Here, θ_x is the network parameter of the CNN for image modality, and θ_y is the network parameter of the deep neural network for text modality.

The objective function of DCMH is defined as follows:

$$\begin{aligned} \min_{\mathbf{B}^{(x)},\mathbf{B}^{(y)},\theta_x,\theta_y} \quad & \mathcal{J} = -\sum_{i,j=1}^{n}(S_{ij}\Theta_{ij} - \log(1+e^{\Theta_{ij}})) \\ & + \gamma(\|\mathbf{B}^{(x)}-\mathbf{F}\|_F^2 + \|\mathbf{B}^{(y)}-\mathbf{G}\|_F^2) \\ & + \eta(\|\mathbf{F1}\|_F^2 + \|\mathbf{G1}\|_F^2) \qquad (1) \\ s.t. \quad & \mathbf{B}^{(x)} \in \{-1,+1\}^{c\times n}, \\ & \mathbf{B}^{(y)} \in \{-1,+1\}^{c\times n}, \end{aligned}$$

where $\mathbf{F} \in \mathbb{R}^{c\times n}$ with $\mathbf{F}_{*i} = f(\mathbf{x}_i;\theta_x)$, $\mathbf{G} \in \mathbb{R}^{c\times n}$ with $\mathbf{G}_{*j} = g(\mathbf{y}_j;\theta_y)$, $\Theta_{ij} = \frac{1}{2}\mathbf{F}_{*i}^T\mathbf{G}_{*j}$, $\mathbf{B}_{*i}^{(x)}$ is the binary hash code for image $\mathbf{x}_i$, $\mathbf{B}_{*j}^{(y)}$ is the binary hash code for text $\mathbf{y}_j$, γ and η are hyper-parameters.

The first term $-\sum_{i,j=1}^{n}(S_{ij}\Theta_{ij} - \log(1+e^{\Theta_{ij}}))$ in (1) is the negative log likelihood of the cross-modal similarities with the likelihood function defined as follows:

$$p(S_{ij}|\mathbf{F}_{*i},\mathbf{G}_{*j}) = \begin{cases} \sigma(\Theta_{ij}) & S_{ij}=1 \\ 1-\sigma(\Theta_{ij}) & S_{ij}=0 \end{cases}$$

where $\Theta_{ij} = \frac{1}{2}\mathbf{F}_{*i}^T\mathbf{G}_{*j}$ and $\sigma(\Theta_{ij}) = \frac{1}{1+e^{-\Theta_{ij}}}$.

It is easy to find that minimizing this negative log likelihood, which is equivalent to maximizing the likelihood, can make the similarity (inner product) between $\mathbf{F}_{*i}$ and $\mathbf{G}_{*j}$ be large when $S_{ij}=1$ and be small when $S_{ij}=0$. Hence, optimizing the first term in (1) can preserve the cross-modal similarity in $\mathbf{S}$ with the image feature representation $\mathbf{F}$ and text feature representation $\mathbf{G}$.

By optimizing the second term $\gamma(\|\mathbf{B}^{(x)}-\mathbf{F}\|_F^2+\|\mathbf{B}^{(y)}-\mathbf{G}\|_F^2)$ in (1), we can get $\mathbf{B}^{(x)} = \text{sign}(\mathbf{F})$ and $\mathbf{B}^{(y)} = \text{sign}(\mathbf{G})$. Hence, we can consider $\mathbf{F}$ and $\mathbf{G}$ to be the continuous surrogate of $\mathbf{B}^{(x)}$ and $\mathbf{B}^{(y)}$, respectively. Because $\mathbf{F}$ and $\mathbf{G}$ can preserve the cross-modal similarity in $\mathbf{S}$, the binary hash codes $\mathbf{B}^{(x)}$ and $\mathbf{B}^{(y)}$ can also be expected to preserve the cross-modal similarity in $\mathbf{S}$, which exactly matches the goal of cross-modal hashing.

The third term $\eta(\|\mathbf{F1}\|_F^2 + \|\mathbf{G1}\|_F^2)$ in (1) is used to make each bit of the hash code be balanced on all the training points. More specifically, the number of $+1$ and that of -1 for each bit on all the training points should be almost the same. This constraint can be used to maximize the information provided by each bit.

In our experiment, we find that better performance can be achieved if the binary codes from the two modalities are set to be the same for the same training points. Hence, we set $\mathbf{B}^{(x)} = \mathbf{B}^{(y)} = \mathbf{B}$. Then, the problem in (1) can be transformed to the following formulation:

$$\begin{aligned} \min_{\mathbf{B},\theta_x,\theta_y} \quad & \mathcal{J} = -\sum_{i,j=1}^{n}(S_{ij}\Theta_{ij} - \log(1+e^{\Theta_{ij}})) \\ & + \gamma(\|\mathbf{B}-\mathbf{F}\|_F^2 + \|\mathbf{B}-\mathbf{G}\|_F^2) \\ & + \eta(\|\mathbf{F1}\|_F^2 + \|\mathbf{G1}\|_F^2) \qquad (2) \\ s.t. \quad & \mathbf{B} \in \{-1,+1\}^{c\times n}. \end{aligned}$$

This is the final objective function of our DCMH for learning.

From (2), we can find that the parameters of the deep neural networks (θ_x and θ_y) and the binary hash code ($\mathbf{B}$) are learned from the same objective function. That is to say, DCMH integrates both feature learning and hash-code learning into the same deep learning framework.

Please note that we only make $\mathbf{B}^{(x)} = \mathbf{B}^{(y)}$ for the *training* points. After we have learned the problem in (2), we still need to generate different binary codes $\mathbf{b}_i^{(x)} = h^{(x)}(\mathbf{x}_i)$ and $\mathbf{b}_i^{(y)} = h^{(y)}(\mathbf{y}_i)$ for the two different modalities of the same point i if point i is a query point or a point from the database rather than a training point. This will be further illustrated in Section 3.3.

3.2. Learning

We adopt an alternating learning strategy to learn θ_x, θ_y and $\mathbf{B}$. Each time we learn one parameter with the other parameters fixed. The whole alternating learning algorithm for DCMH is briefly outlined in Algorithm 1, and the detailed derivation will be introduced in the following content of this subsection.

3.2.1 Learn θ_x, with θ_y and B Fixed

When θ_y and $\mathbf{B}$ are fixed, we learn the CNN parameter θ_x of the image modality by using a back-propagation (BP) algorithm. As most existing deep learning methods [16], we utilize stochastic gradient descent (SGD) to learn θ_x with the BP algorithm. More specifically, in each iteration we sample a mini-batch of points from the training set and then carry out our learning algorithm based on the sampled data.

In particular, for each sampled point $\mathbf{x}_i$, we first compute the following gradient:

$$\frac{\partial \mathcal{J}}{\partial \mathbf{F}_{*i}} = \frac{1}{2}\sum_{j=1}^{n}(\sigma(\Theta_{ij})\mathbf{G}_{*j} - S_{ij}\mathbf{G}_{*j}) + 2\gamma(\mathbf{F}_{*i} - \mathbf{B}_{*i}) + 2\eta\mathbf{F}\mathbf{1}. \quad (3)$$

Then we can compute $\frac{\partial \mathcal{J}}{\partial \theta_x}$ with $\frac{\partial \mathcal{J}}{\partial \mathbf{F}_{*i}}$ by using the chain rule, based on which BP can be used to update the parameter θ_x.

3.2.2 Learn θ_y, with θ_x and $\mathbf{B}$ Fixed

When θ_x and $\mathbf{B}$ are fixed, we also learn the neural network parameter θ_y of the text modality by using SGD with a BP algorithm. More specifically, for each sampled point $\mathbf{y}_j$, we first compute the following gradient:

$$\frac{\partial \mathcal{J}}{\partial \mathbf{G}_{*j}} = \frac{1}{2}\sum_{i=1}^{n}(\sigma(\Theta_{ij})\mathbf{F}_{*i} - S_{ij}\mathbf{F}_{*i}) + 2\gamma(\mathbf{G}_{*j} - \mathbf{B}_{*j}) + 2\eta\mathbf{G}\mathbf{1}. \quad (4)$$

Then we can compute $\frac{\partial \mathcal{J}}{\partial \theta_y}$ with $\frac{\partial \mathcal{J}}{\partial \mathbf{G}_{*j}}$ by using the chain rule, based on which BP can be used to update the parameter θ_y.

3.2.3 Learn $\mathbf{B}$, with θ_x and θ_y Fixed

When θ_x and θ_y are fixed, the problem in (2) can be reformulated as follows:

$$\max_{\mathbf{B}} \operatorname{tr}(\mathbf{B}^T(\gamma(\mathbf{F}+\mathbf{G}))) = \operatorname{tr}(\mathbf{B}^T\mathbf{V}) = \sum_{i,j} B_{ij}V_{ij}$$
$$s.t. \quad \mathbf{B} \in \{-1,+1\}^{c\times n},$$

where $\mathbf{V} = \gamma(\mathbf{F}+\mathbf{G})$.

It is easy to find that the binary code B_{ij} should keep the same sign as V_{ij}. Therefore, we have:

$$\mathbf{B} = \operatorname{sign}(\mathbf{V}) = \operatorname{sign}(\gamma(\mathbf{F}+\mathbf{G})). \quad (5)$$

3.3. Out-of-Sample Extension

For any point which is not in the training set, we can obtain its hash code as long as one of its modalities (image or text) is observed. In particular, given the image modality $\mathbf{x}_q$ of point q, we can adopt forward propagation to generate the hash code as follows:

$$\mathbf{b}_q^{(x)} = h^{(x)}(\mathbf{x}_q) = \operatorname{sign}(f(\mathbf{x}_q;\theta_x)).$$

Similarly, if point q only has the text modality $\mathbf{y}_q$, we can also generate the hash code $\mathbf{b}_q^{(y)}$ as follows:

$$\mathbf{b}_q^{(y)} = h^{(y)}(\mathbf{y}_q) = \operatorname{sign}(g(\mathbf{y}_q;\theta_y)).$$

Hence, our DCMH model can be used for cross-modal search where the query points have one modality and the points in database have the other modality.

Algorithm 1 The learning algorithm for DCMH.

```
Input: Image set $\mathbf{X}$, text set $\mathbf{Y}$, and cross-modal similarity matrix $\mathbf{S}$.
Output: Parameters $\theta_x$ and $\theta_y$ of the deep neural networks, and binary code matrix $\mathbf{B}$.
  Initialization
  Initialize neural network parameters $\theta_x$ and $\theta_y$, mini-batch size $N_x = N_y = 128$, and iteration number $t_x = \lceil n/N_x \rceil, t_y = \lceil n/N_y \rceil$.
  repeat
    for $iter = 1, 2, \cdots, t_x$ do
      Randomly sample $N_x$ points from $\mathbf{X}$ to construct a mini-batch.
      For each sampled point $\mathbf{x}_i$ in the mini-batch, calculate $\mathbf{F}_{*i} = f(\mathbf{x}_i;\theta_x)$ by forward propagation.
      Calculate the derivative according to (3).
      Update the parameter $\theta_x$ by using back propagation.
    end for
    for $iter = 1, 2, \cdots, t_y$ do
      Randomly sample $N_y$ points from $\mathbf{Y}$ to construct a mini-batch.
      For each sampled point $\mathbf{y}_j$ in the mini-batch, calculate $\mathbf{G}_{*j} = g(\mathbf{y}_j;\theta_y)$ by forward propagation.
      Calculate the derivative according to (4).
      Update the parameter $\theta_y$ by using back propagation.
    end for
    Learn $\mathbf{B}$ according to (5).
  until a fixed number of iterations
```

4. Experiment

We carry out experiments on image-text datasets to verify the effectiveness of DCMH. DCMH is implemented with the open source deep learning toolbox MatConvNet [31] on a NVIDIA K80 GPU server.

4.1. Datasets

Three datasets, *MIRFLICKR-25K* [12], *IAPR TC-12* [8] and *NUS-WIDE* [6], are used for evaluation.

The original *MIRFLICKR-25K* dataset [12] consists of 25,000 images collected from Flickr website. Each image is associated with several textual tags. Hence, each point is a image-text pair. We select those points which have at least 20 textual tags for our experiment. The text for each point is represented as a 1386-dimensional bag-of-words vector. For the hand-crafted feature based method, each image is represented by a 512-dimensional GIST feature vector. Furthermore, each point is manually annotated with one of the 24 unique labels.

The *IAPR TC-12* dataset [8] consists of 20,000 image-text pairs which are annotated using 255 labels. We use the entire dataset for our experiment. The text for each point is represented as a 2912-dimensional bag-of-words vector. For the hand-crafted feature based method, each image is represented by a 512-dimensional GIST feature vector.

The *NUS-WIDE* dataset [6] contains 260,648 web images, and some images are associated with textual tags. It is a multi-label dataset where each point is annotated with one or multiple labels from 81 concept labels. We select 195,834 image-text pairs that belong to the 21 most frequent concepts. The text for each point is represented as a 1000-dimensional bag-of-words vector. The hand-crafted feature for each image is a 500-dimensional bag-of-visual words (BOVW) vector.

For all datasets, the image i and text j are considered to be similar if point i and point j share at least one common label. Otherwise, they are considered to be dissimilar.

4.2. Evaluation Protocol and Baseline

4.2.1 Evaluation Protocol

For *MIRFLICKR-25K* and *IAPR TC-12* datasets, we randomly sample 2,000 data points as the test (query) set and the remaining points as the retrieval set (database). For *NUS-WIDE* dataset, we take 2,100 data points as the test set and the rest as the retrieval set. Moreover, we sample 10,000 data points from the retrieval set as training set for *MIRFLICKR-25K* and *IAPR TC-12*. For *NUS-WIDE* dataset, we sample 10,500 data points from the retrieval set as training set. The ground-truth neighbors are defined as those image-text pairs which share at least one common label.

For hashing-based retrieval, *Hamming ranking* and *hash lookup* are two widely used retrieval protocols [25]. We also adopt these two protocols to evaluate our method and other baselines. The Hamming ranking protocol ranks the points in the database (retrieval set) according to their Hamming distances to the given query point, in an increasing order. Mean average precision (MAP) [25] is the widely used metric to measure the accuracy of the Hamming ranking protocol. The hash lookup protocol returns all the points within a certain Hamming radius away from the query point. The precision-recall curve is the widely used metric to measure the accuracy of the hash lookup protocol.

4.2.2 Baseline

Six state-of-the-art cross-modal hashing methods are adopted as baselines for comparison, including SePH [22], STMH [33], SCM [35], CMFH [7], CCA [11] and DVSH [3]. Since DVSH can only be used for a special CMH case where one of the modalities have to be temporal dynamics, we compare DCMH with DVSH only on *IAPR TC-12* dataset where the original texts are sentences which can be treated as temporal dynamics. The texts in *MIRFLICKR-25K* and *NUS-WIDE* are tags which are not suitable for DVSH. Please note that the texts are represented as BOW vectors for all the evaluated methods except DVSH.

Source codes of SePH, STMH and SCM are kindly provided by the corresponding authors. While for CMFH and CCA whose codes are not available, we implement them carefully by ourselves. SePH is a kernel-based method, for which we use RBF kernel and take 500 randomly selected points as kernel bases by following its authors' suggestion. In SePH, the authors propose two strategies to construct the hash codes for retrieval (database) points according to whether both modalities of a point are observed or not. However, in this paper we only use one modality for the database (retrieval) points[2], because the focus of this paper is on cross-modal retrieval. All the other parameters for all baselines are set according to the suggestion of the original papers of these baselines.

For DCMH, we use a validation set to choose the hyper-parameters γ and η, and find that good performance can be achieved with $\gamma = \eta = 1$. Hence, we set $\gamma = \eta = 1$ for DCMH. We exploit the CNN-F network [5] pre-trained on ImageNet dataset [27] to initialize the first seven layers of the CNN for image modality. All the other parameters of the deep neural networks in DCMH are randomly initialized. The input for the image modality is the raw pixels, and that for the text modality is the BOW vectors. We fix the mini-batch size to be 128 and set the iteration number of the outer-loop in Algorithm 1 to be 500. The learning rate is chosen from 10^{-6} to 10^{-1} with a validation set. All experiments are run for five times, and the average performance is reported.

4.3. Accuracy

4.3.1 Hamming Ranking

The MAP results for DCMH and other baselines with hand-crafted features on *MIRFLICKR-25K*, *IAPR TC-12* and *NUS-WIDE* datasets are reported in Table 3. Here, "$I \to T$" denotes the case where the query is image and the database is text, and "$T \to I$" denotes the case where the query is text and the database is image. We can find that DCMH can outperform all the other baselines with hand-crafted features.

To further verify the effectiveness of DCMH, we exploit the CNN-F deep network [5] pre-trained on ImageNet dataset, which is the same as the initial CNN of the image modality in DCMH, to extract CNN features. All the baselines are trained based on these CNN features. The MAP results for DCMH and other baselines with CNN features on three datasets are reported in Table 4. We can find that DCMH can outperform all the other baselines except SePH for image to text retrieval on *NUS-WIDE*.

[2]For both SePH and DCMH, the accuracy by using both modalities for database points is typically higher than that by using only one modality for database points. DCMH can still outperform SePH for cases with both modalities for database points. This result is omitted in this paper due to space limitation.

Table 3. MAP. The best accuracy is shown in boldface. The baselines are based on hand-crafted features.

Task	Method	MIRFLICKR-25K			IAPR TC-12			NUS-WIDE		
		16 bits	32 bits	64 bits	16 bits	32 bits	64 bits	16 bits	32 bits	64 bits
$I \rightarrow T$	DCMH	**0.7410**	**0.7465**	**0.7485**	**0.4526**	**0.4732**	**0.4844**	**0.5903**	**0.6031**	**0.6093**
	SePH	0.6573	0.6603	0.6616	0.4112	0.4158	0.4203	0.4787	0.4869	0.4888
	STMH	0.5921	0.5950	0.5980	0.3580	0.3732	0.3819	0.3973	0.4082	0.4153
	SCM	0.6290	0.6404	0.6480	0.3833	0.3898	0.3878	0.4650	0.4714	0.4822
	CMFH	0.5818	0.5808	0.5805	0.3683	0.3734	0.3786	0.3568	0.3624	0.3661
	CCA	0.5695	0.5663	0.5641	0.3345	0.3254	0.3193	0.3414	0.3336	0.3282
$T \rightarrow I$	DCMH	**0.7827**	**0.7900**	**0.7932**	**0.5185**	**0.5378**	**0.5468**	**0.6389**	**0.6511**	**0.6571**
	SePH	0.6480	0.6521	0.6545	0.4024	0.4074	0.4131	0.4489	0.4539	0.4587
	STMH	0.5802	0.5846	0.5855	0.3445	0.3570	0.3690	0.3607	0.3738	0.3842
	SCM	0.6195	0.6302	0.6366	0.3698	0.3734	0.3696	0.4370	0.4428	0.4504
	CMFH	0.5787	0.5774	0.5784	0.3619	0.3687	0.3769	0.3623	0.3670	0.3723
	CCA	0.5690	0.5659	0.5639	0.3340	0.3255	0.3197	0.3392	0.3320	0.3272

Table 4. MAP. The best accuracy is shown in boldface. The baselines are based on CNN-F features.

Task	Method	MIRFLICKR-25K			IAPR TC-12			NUS-WIDE		
		16 bits	32 bits	64 bits	16 bits	32 bits	64 bits	16 bits	32 bits	64 bits
$I \rightarrow T$	DCMH	**0.7410**	**0.7465**	**0.7485**	**0.4526**	**0.4732**	**0.4844**	0.5903	0.6031	0.6093
	SePH	0.7123	0.7194	0.7232	0.4442	0.4563	0.4639	**0.6037**	**0.6136**	**0.6211**
	STMH	0.6132	0.6219	0.6274	0.3775	0.4002	0.4130	0.4710	0.4864	0.4942
	SCM	0.6851	0.6921	0.7003	0.3692	0.3666	0.3802	0.5409	0.5485	0.5553
	CMFH	0.6377	0.6418	0.6451	0.4189	0.4234	0.4251	0.4900	0.5053	0.5097
	CCA	0.5719	0.5693	0.5672	0.3422	0.3361	0.3300	0.3604	0.3485	0.3390
$T \rightarrow I$	DCMH	**0.7827**	**0.7900**	**0.7932**	**0.5185**	**0.5378**	**0.5468**	**0.6389**	**0.6511**	**0.6571**
	SePH	0.7216	0.7261	0.7319	0.4423	0.4562	0.4648	0.5983	0.6025	0.6109
	STMH	0.6074	0.6153	0.6217	0.3687	0.3897	0.4044	0.4471	0.4677	0.4780
	SCM	0.6939	0.7012	0.7060	0.3453	0.3410	0.3470	0.5344	0.5412	0.5484
	CMFH	0.6365	0.6399	0.6429	0.4168	0.4212	0.4277	0.5031	0.5187	0.5225
	CCA	0.5742	0.5713	0.5691	0.3493	0.3438	0.3378	0.3614	0.3494	0.3395

4.3.2 Hash Lookup

In the hash lookup protocol, we can compute the precision and recall for the returned points given any Hamming radius. By varying the Hamming radius from 0 to c with a stepsize 1, we can get the precision-recall curve.

Figure 2 shows the precision-recall curve with code length 16 on three datasets, where the first two sub-figures are based on hand-crafted features and the last two sub-figures are based on CNN-F features for baselines in each row of the figures. We can find that DCMH can dramatically outperform the baselines for both hand-crafted features and CNN-F features. Our DCMH can also achieve the best performance on other cases with different values of code length, such as 32 bits and 64 bits. Those results are omitted due to space limitation.

4.4. Comparison with DVSH

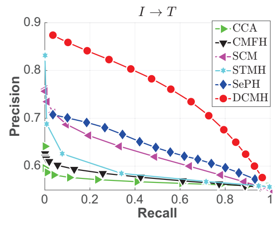
Since the source code of DVSH is not publicly available and it is also difficult to re-implement DVSH, we adopt the same experimental setting as that in DVSH [3] to evaluate DCMH and directly use the result in DVSH [3] for comparison. The top-500 MAP result on *IAPR TC-12* dataset is listed in Table 5. Please note that the text input for DVSH is sentences and we represent the sentences as BOW vectors for DCMH. We can find that DCMH can outperform DVSH in most cases.

Table 5. Top-500 MAP on *IAPR TC-12* dataset.

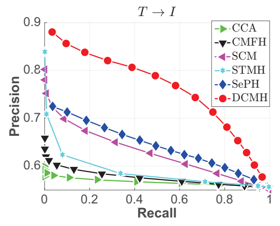
Task	Method	16 bits	32 bits	64 bits
$I \rightarrow T$	DCMH	**0.5780**	0.6061	0.6310
	DVSH	0.5696	**0.6321**	**0.6964**
$T \rightarrow I$	DCMH	**0.6594**	**0.6744**	**0.6905**
	DVSH	0.6037	0.6395	0.6806

4.5. Sensitivity to Parameters

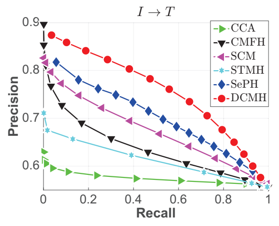
We explore the influence of the hyper-parameters γ and η. Figure 3 shows the MAP results on *MIRFLICKR-25K* dataset with different values of γ and η, where the code length is 16 bits. We can see that DCMH is not sensitive to γ and η with $0.01 < \gamma < 2$ and $0.01 < \eta < 2$.



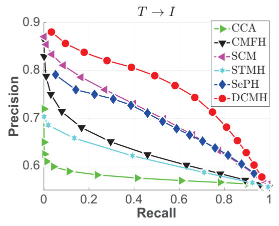
(a) *MIRFLICKR-25K* @Hand-crafted feature



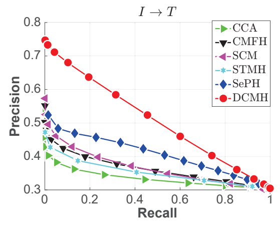
(b) *MIRFLICKR-25K* @Hand-crafted feature



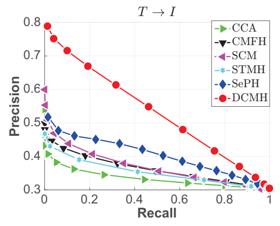
(c) *MIRFLICKR-25K* @CNN-F feature



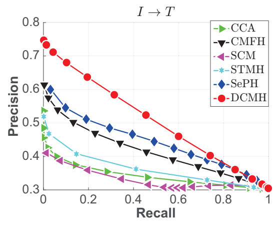
(d) *MIRFLICKR-25K* @CNN-F feature



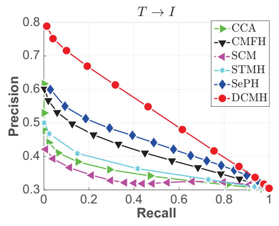
(e) *IAPR TC-12* @Hand-crafted feature



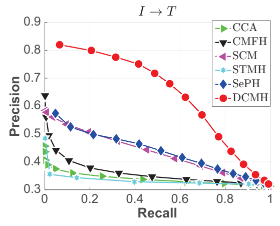
(f) *IAPR TC-12* @Hand-crafted feature



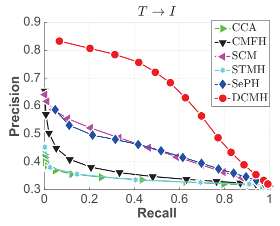
(g) *IAPR TC-12* @CNN-F feature



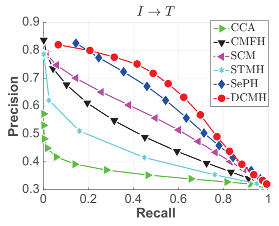
(h) *IAPR TC-12* @CNN-F feature



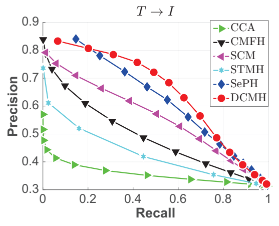
(i) *NUS-WIDE* @Hand-crafted feature



(j) *NUS-WIDE* @Hand-crafted feature



(k) *NUS-WIDE* @CNN-F feature



(l) *NUS-WIDE* @CNN-F feature

Figure 2. Precision-recall curves on three datasets. The code length is 16.

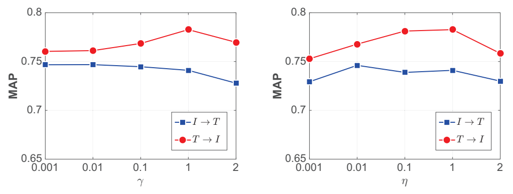


Figure 3. The influence of hyper-parameters.

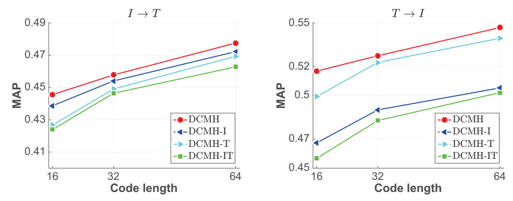


Figure 4. MAP on *IAPR TC-12*.

4.6. Further Analysis

To further verify the effectiveness of feature learning, we evaluate some variants of DCMH, *i.e*., DCMH-I, DCMH-T, DCMH-IT. DCMH-I denotes the variant without image feature learning, in which we fix the parameters of the first seven layers for image modality during training. DCMH-T denotes the variant without text feature learning, in which we replace the deep neural network of text modality as a linear projection. DCMH-IT denotes the variant without both image and text feature learning.

Figure 4 reports the MAP results on *IAPR TC-12*. We can find that DCMH can achieve higher accuracy than DCMH-I, DCMH-T and DCMH-IT, which demonstrates the importance of simultaneous hash-code learning and feature learning.

5. Conclusion

In this paper, we have proposed a novel hashing method, called DCMH, for cross-modal retrieval applications. DCMH is an end-to-end deep learning framework which can perform feature learning and hash-code learning simultaneously. Experiments on three datasets show that DCMH can significantly outperform other baselines to achieve the state-of-the-art performance in real applications.

6. Acknowledgements

This work is supported by the NSFC (61472182) and a fund from Tencent.

References

[1] A. Andoni and P. Indyk. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. *Commun. ACM*, 51(1):117–122, 2008.

[2] M. M. Bronstein, A. M. Bronstein, F. Michel, and N. Paragios. Data fusion through cross-modality metric learning using similarity-sensitive hashing. In *CVPR*, pages 3594–3601, 2010.

[3] Y. Cao, M. Long, J. Wang, Q. Yang, and P. S. Yu. Deep visual-semantic hashing for cross-modal retrieval. In *SIGKDD*, pages 1445–1454, 2016.

[4] M. Á. Carreira-Perpiñán and R. Raziperchikolaei. Hashing with binary autoencoders. In *CVPR*, pages 557–566, 2015.

[5] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman. Return of the devil in the details: Delving deep into convolutional nets. In *BMVC*, 2014.

[6] 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 *CIVR*, 2009.

[7] G. Ding, Y. Guo, and J. Zhou. Collective matrix factorization hashing for multimodal data. In *CVPR*, pages 2083–2090, 2014.

[8] H. J. Escalante, C. A. Hernández, J. A. González, A. López-López, M. Montes-y-Gómez, E. F. Morales, L. E. Sucar, L. V. Pineda, and M. Grubinger. The segmented and annotated IAPR TC-12 benchmark. *CVIU*, 114(4):419–428, 2010.

[9] Y. Gong and S. Lazebnik. Iterative quantization: A procrustean approach to learning binary codes. In *CVPR*, pages 817–824, 2011.

[10] J.-P. Heo, Y. Lee, J. He, S.-F. Chang, and S.-E. Yoon. Spherical hashing. In *CVPR*, pages 2957–2964, 2012.

[11] H. Hotelling. Relations between two sets of variates. *Biometrika*, pages 321–377, 1936.

[12] M. J. Huiskes and M. S. Lew. The MIR flickr retrieval evaluation. In *ACM MM*, pages 39–43, 2008.

[13] Q.-Y. Jiang and W.-J. Li. Deep cross-modal hashing. *CoRR*, abs/1602.02255, 2016.

[14] Y. Kang, S. Kim, and S. Choi. Deep learning to hash with multiple representations. In *ICDM*, pages 930–935, 2012.

[15] W. Kong and W.-J. Li. Isotropic hashing. In *NIPS*, pages 1655–1663, 2012.

[16] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, pages 1106–1114, 2012.

[17] B. Kulis and K. Grauman. Kernelized locality-sensitive hashing for scalable image search. In *ICCV*, pages 2130–2137, 2009.

[18] S. Kumar and R. Udupa. Learning hash functions for cross-view similarity search. In *IJCAI*, pages 1360–1365, 2011.

[19] Y. LeCun, B. E. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. E. Hubbard, and L. D. Jackel. Handwritten digit recognition with a back-propagation network. In *NIPS*, pages 396–404, 1989.

[20] W.-J. Li, S. Wang, and W.-C. Kang. Feature learning based deep supervised hashing with pairwise labels. In *IJCAI*, pages 1711–1717, 2016.

[21] G. Lin, C. Shen, Q. Shi, A. van den Hengel, and D. Suter. Fast supervised hashing with decision trees for high-dimensional data. In *CVPR*, pages 1971–1978, 2014.

[22] Z. Lin, G. Ding, M. Hu, and J. Wang. Semantics-preserving hashing for cross-view retrieval. In *CVPR*, pages 3864–3872, 2015.

[23] V. E. Liong, J. Lu, G. Wang, P. Moulin, and J. Zhou. Deep hashing for compact binary codes learning. In *CVPR*, pages 2475–2483, 2015.

[24] H. Liu, R. Wang, S. Shan, and X. Chen. Deep supervised hashing for fast image retrieval. In *CVPR*, pages 2064–2072, 2016.

[25] W. Liu, C. Mu, S. Kumar, and S.-F. Chang. Discrete graph hashing. In *NIPS*, pages 3419–3427, 2014.

[26] W. Liu, J. Wang, R. Ji, Y.-G. Jiang, and S.-F. Chang. Supervised hashing with kernels. In *CVPR*, pages 2074–2081, 2012.

[27] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. S. Bernstein, A. C. Berg, and F.-F. Li. Imagenet large scale visual recognition challenge. *CoRR*, abs/1409.0575, 2014.

[28] F. Shen, C. Shen, W. Liu, and H. T. Shen. Supervised discrete hashing. In *CVPR*, pages 37–45, 2015.

[29] F. Shen, C. Shen, Q. Shi, A. van den Hengel, and Z. Tang. Inductive hashing on manifolds. In *CVPR*, pages 1562–1569, 2013.

[30] J. Song, Y. Yang, Z. Huang, H. T. Shen, and R. Hong. Multiple feature hashing for real-time large scale near-duplicate video retrieval. In *ACM MM*, pages 423–432, 2011.

[31] A. Vedaldi and K. Lenc. Matconvnet: Convolutional neural networks for MATLAB. In *ACM MM*, pages 689–692, 2015.

[32] D. Wang, P. Cui, M. Ou, and W. Zhu. Deep multimodal hashing with orthogonal regularization. In *IJCAI*, pages 2291–2297, 2015.

[33] D. Wang, X. Gao, X. Wang, and L. He. Semantic topic multimodal hashing for cross-media retrieval. In *IJCAI*, pages 3890–3896, 2015.

[34] J. Wang, O. Kumar, and S.-F. Chang. Semi-supervised hashing for scalable image retrieval. In *CVPR*, pages 3424–3431, 2010.

[35] D. Zhang and W.-J. Li. Large-scale supervised multimodal hashing with semantic correlation maximization. In *AAAI*, pages 2177–2183, 2014.

[36] D. Zhang, F. Wang, and L. Si. Composite hashing with multiple information sources. In *SIGIR*, pages 225–234, 2011.

[37] F. Zhao, Y. Huang, L. Wang, and T. Tan. Deep semantic ranking based hashing for multi-label image retrieval. In *CVPR*, pages 1556–1564, 2015.

[38] Y. Zhen and D.-Y. Yeung. Co-regularized hashing for multimodal data. In *NIPS*, pages 1385–1393, 2012.

[39] Y. Zhen and D.-Y. Yeung. A probabilistic model for multimodal hash function learning. In *SIGKDD*, pages 940–948, 2012.

[40] B. Zhuang, G. Lin, C. Shen, and I. D. Reid. Fast training of triplet-based deep binary embedding networks. In *CVPR*, pages 5955–5964, 2016.