Deep Cross-Modal Hashing

Qing-Yuan Jiang JIANGQY@LAMDA.NJU.EDU.CN
Wu-Jun Li LIWUJUN@NJU.EDU.CN
National Key Laboratory for Novel Software Technology,
Department of Computer Science and Technology,
Nanjing University, Nanjing, China

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, almost all 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 cross-modal hashing 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 two real datasets with text-image 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 (Andoni & Indyk, 2008; Andoni & Razenshteyn, 2015) 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 (Weiss et al., 2008; Raginsky & Lazebnik, 2009; Wang et al., 2010; Liu et al., 2011; Norouzi & Fleet, 2011; Norouzi et al., 2012; Rastegari et al., 2013; Yu et al., 2014; Liu et al., 2014; Shrivastava & Li, 2014; Andoni et al., 2015; Neyshabur & Srebro, 2015; Leng et al., 2015). 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. 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 roughly divided into two main categories: *mutli-source hashing* (MSH) (Song et al., 2011; Zhang et al., 2011) and *cross-modal hashing* (CMH) (Kumar & Udupa, 2011; Ding et al., 2014; Zhang & Li, 2014; Lin et al., 2015). 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 the query points and those in database. In practice, the application of MSH is limited because in many cases it is difficult to acquire all the 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 the 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. Representative methods include cross modality similarity sensitive hashing (CMSSH) (Bronstein et al., 2010), cross view hashing (CVH) (Kumar & Udupa, 2011), multi-modal latent binary embedding (MLBE) (Zhen & Yeung, 2012a), co-regularized hashing (CRH) (Zhen & Yeung, 2012b), semantic correlation maximization (SCM) (Zhang & Li, 2014), collective matrix factorization hashing (CMFH) (Ding et al., 2014), semantic topic multi-modal hashing (STMH) (Wang et al., 2015) and semantics preserving hashing (SePH) (Lin et al., 2015). 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 (LeCun et al., 1989; Krizhevsky et al., 2012) 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 (Zhao et al., 2015; Liong et al., 2015). However, to the best of our knowledge, there has not appeared any deep CMH methods which can perform simultaneous feature learning and hash-code learning in the same framework.

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.
- To the best of our knowledge, DCMH is the first CMH method which integrates both feature learning and hash-code learning into the same deep learning framework.
- The hash-code learning problem is essentially a discrete optimization problem, which is difficult to learn. Hence, most existing CMH methods typically 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 (Liu et al., 2014). Unlike these relaxation-based methods, DCMH directly learns the discrete hash codes without relaxation.
- Experiments on real datasets with text-image 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

In this section, we introduce the notation and problem definition of this paper.

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 (i, j)th element 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 text-image 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.

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 with Deep Neural Networks

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

The deep neural network for image modality is a CNN model adapted from (Chatfield et al., 2014). There are eight layers in this CNN model. The first seven layers are the same as those in CNN-F of (Chatfield et al., 2014). The eightth layer is a fully-connected layer with the output being the learned image features.

Table 1 shows the detailed configuration of the CNN 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.

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

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

Layer	Configuration
full1	Length of BOW vector
full2	4096
full3	Hash code length c

- "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) (Krizhevsky et al., 2012) 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 (ReLU) (Krizhevsky et al., 2012) 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 three fully-connected layers, denoted as "full1 - full3". 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 two layers is ReLU, and that for the third layer is the identity function.

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.

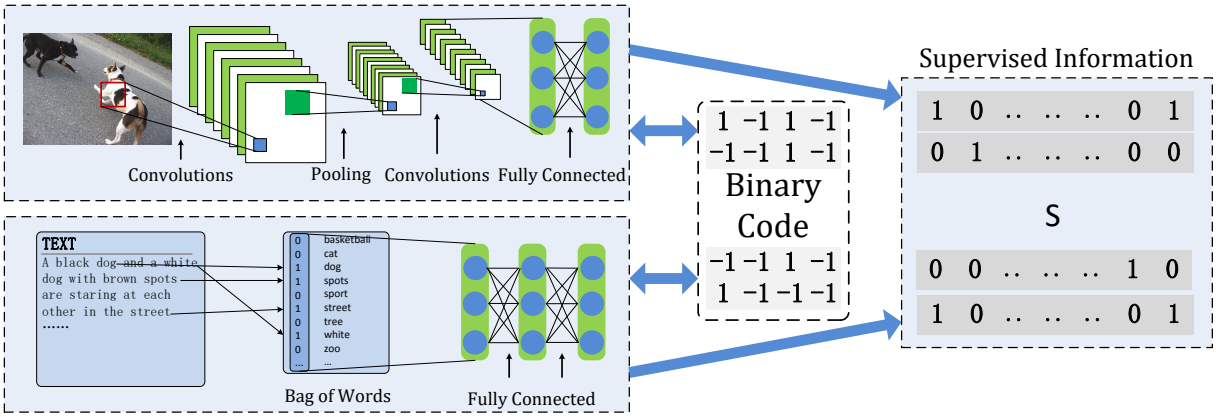


Figure 1. The end-to-end deep learning framework of our DCMH model.

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:

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

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 $\eta(||\mathbf{F}\mathbf{1}||_F^2 + ||\mathbf{G}\mathbf{1}||_F^2)$ 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 training points. Hence, we add another constraint $\mathbf{B} = \mathbf{B}^{(x)} = \mathbf{B}^{(y)}$ in (1). With this constraint, the problem in (1) can be equivalently transformed to the following reduced formulation:

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

where $\mathbf{F} = f(\mathbf{X};\theta_x)$ means $\mathbf{F}_{*i} = f(\mathbf{x}_i;\theta_x)$, $\mathbf{G} = g(\mathbf{Y};\theta_y)$ means $\mathbf{G}_{*j} = g(\mathbf{y}_j;\theta_y)$. 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 optimize 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. Fix θ_y and $\mathbf{B}$, optimize θ_x

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 (Krizhevsky et al., 2012), 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:

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

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. Fix $\mathbf{B}$ and θ_x, optimize θ_y

When $\mathbf{B}$ and θ_x 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:

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 θ_x and θ_y of the deep neural networks, and binary code matrix $\mathbf{B}$.
Initialization
Initialize neural network parameters θ_x and θ_y, mini-batch size $N_x = N_y = 128$, and iteration number $t_x = n/N_x, t_y = n/N_y$.
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 mini-batch, calculate $\mathbf{F}_{*i} = f(\mathbf{x}_i; \theta_x)$ by forward propagation.
    Calculate the derivative according to (3).
    Update the parameter θ_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 mini-batch, calculate $\mathbf{G}_{*j} = g(\mathbf{y}_j; \theta_y)$ by forward propagation.
    Calculate the derivative according to (4).
    Update the parameter θ_y by using back propagation.
  end for
  Optimize $\mathbf{B}$ according to (5).
until a fixed number of iterations

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

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. Fix θ_x and θ_y, optimize $\mathbf{B}$

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:

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

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:

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

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:

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

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.

4. Experiment

We carry out experiments on text-image datasets to verify the effectiveness of DCMH. DCMH is implemented with the open source deep learning toolbox MatConvNet (Vedaldi & Lenc, 2015) on a NVIDIA K40 GPU server.

4.1. Datasets

Two datasets, *MIRFLICKR-25K* (Huiskes & Lew, 2008) and *NUS-WIDE* (Chua et al., 2009), are used for evaluation.

The original *MIRFLICKR-25K* dataset (Huiskes & Lew, 2008) consists of 25,000 images collected from Flickr website. Each image is associated with several textual tags. Hence, each point is a text-image pair. We select those points which have at least 20 textual tags for our experiment, and subsequently we get 20,015 points 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 SIFT feature vector. Furthermore, each point is manually annotated with one of the 24 unique labels. The image i and text j are considered to be similar if point i and point j share the same label. Otherwise, they are considered to be dissimilar.

The *NUS-WIDE* dataset (Chua et al., 2009) 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 186,577 text-image pairs that belong to the 10 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. The image i and text j are considered to be similar if point i and point j share at least one concept label. Otherwise, they are considered to be dissimilar.

4.2. Evaluation Protocol and Baseline

4.2.1. Evaluation Protocol

For *MIRFLICKR-25K* dataset, we take 2000 data points as the test (query) set and the remaining points as the retrieval set (database). For *NUS-WIDE* dataset, we take 1% of the dataset as the test (query) set and the rest as the retrieval set. Moreover, we take 5000 data points from the retrieval set to construct the training set for both *MIRFLICKR-25K* and *NUS-WIDE*. The ground-truth neighbors are defined as those text-image pairs which share at least one semantic label.

For hashing-based retrieval, *Hamming ranking* and *hash lookup* are two widely used retrieval procedures (Liu et al., 2014). We also adopt these two procedures to evaluate our method and other baselines. The Hamming ranking procedure 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) (Liu et al., 2014) is the widely used metric to measure the accuracy of the Hamming ranking procedure. The hash lookup procedure returns all the points within a certain Hamming radius away from the query point. The precision-recall curve and F-measure (Liu et al., 2014) are widely used metrics to measure the accuracy of the hash lookup procedure.

4.2.2. Baseline

Five state-of-the-art cross-modal hashing methods are adopted as baselines for comparison, including SePH (Lin et al., 2015), STMH (Wang et al., 2015), SCM (Zhang & Li, 2014), CMFH (Ding et al., 2014) and CCA (Hotelling, 1936). 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 can only use one modality for the database (retrieval) points, 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 our DCMH, we use a validation set to choose the hyper-parameter γ and η, and find that good performance can be achieved with $\gamma = \eta = 1$. Hence, we set $\gamma = \eta = 1$ for all our experiments. We exploit the CNN-F network (Chatfield et al., 2014) pre-trained on ImageNet dataset (Russakovsky

et al., 2014) to initialize the first seven layers of the CNN for image modality, and 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.

4.3. Accuracy

We report the accuracy for both Hamming ranking procedure and hash lookup procedure.

4.3.1. Hamming Ranking

The MAP results for DCMH and other baselines with hand-crafted features on *MIRFLICKR-25K* and *NUS-WIDE* are reported in Table 3 and Table 4, respectively. We can find that DCMH can outperform all the other baselines with hand-crafted features.

Table 3. Comparison to baselines with hand-crafted features on *MIRFLICKR-25K* in terms of MAP. The best accuracy is shown in boldface.

Task	Method	code length		
		16 bits	32 bits	64 bits
Image Query v.s. Text Database	DCMH	**0.7127**	**0.7203**	**0.7303**
	SePH	0.6441	0.6492	0.6508
	STMH	0.5876	0.5951	0.5942
	SCM	0.6153	0.6279	0.6288
	CMFH	0.5804	0.5790	0.5797
	CCA	0.5634	0.5630	0.5626
Text Query v.s. Image Database	DCMH	**0.7504**	**0.7574**	**0.7704**
	SePH	0.6455	0.6474	0.6506
	STMH	0.5763	0.5877	0.5826
	SCM	0.6102	0.6184	0.6192
	CMFH	0.5782	0.5778	0.5779
	CCA	0.5639	0.5631	0.5627

Table 4. Comparison to baselines with hand-crafted features on *NUS-WIDE* in terms of MAP. The best accuracy is shown in boldface.

Task	Method	code length		
		16 bits	32 bits	64 bits
Image Query v.s. Text Database	DCMH	**0.6249**	**0.6355**	**0.6438**
	SePH	0.5314	0.5340	0.5429
	STMH	0.4344	0.4461	0.4534
	SCM	0.4904	0.4945	0.4992
	CMFH	0.3825	0.3858	0.3890
	CCA	0.3742	0.3667	0.3617
Text Query v.s. Image Database	DCMH	**0.6791**	**0.6829**	**0.6906**
	SePH	0.5086	0.5055	0.5170
	STMH	0.3845	0.4089	0.4181
	SCM	0.4595	0.4650	0.4691
	CMFH	0.3915	0.3944	0.3990
	CCA	0.3731	0.3661	0.3613

To further verify the effectiveness of DCMH, we exploit the CNN-F deep network (Chatfield et al., 2014) 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 *MIRFLICKR-25K* and *NUS-WIDE* are reported in Table 5 and Table 6, respectively. We can find that DCMH can outperform all the other baselines except SePH. For SePH, DCMH can outperform it in most cases except the image to text retrieval on NUS-WIDE. Please note that SePH is a kernel-based method, which constructs kernels based on the CNN-F image features and text features. However, our DCMH can be seen as a linear method with deep features because the final layers of both modalities are fully-collected ones with identity activation functions. We find that the better performance of SePH mainly comes from the kernel features of SePH, which is verified by the worse results of a linear variant of SePH without kernels called "SePH-linear" in Table 6. DCMH can outperform SePH with linear features in all cases. And even for SePH with kernel features, DCMH can outperform it for most cases. Hence, compared with these baselines with CNN-F features, the better accuracy of DCMH verifies that integrating both feature learning and hash-code learning into the same framework may improve the performance.

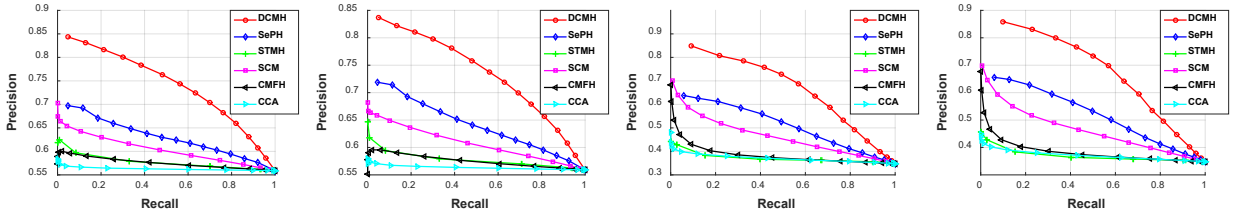
Table 5. Comparison to baselines with CNN-F features on *MIRFLICKR-25K* in terms of MAP. The best accuracy is shown in boldface.

Task	Method	code length		
		16 bits	32 bits	64 bits
Image Query v.s. Text Database	DCMH	**0.7150**	**0.7203**	**0.7303**
	SePH	0.7090	0.7110	0.7169
	STMH	0.6242	0.6294	0.6314
	SCM	0.6281	0.6286	0.6367
	CMFH	0.5761	0.5798	0.5807
	CCA	0.5619	0.5616	0.5616
Text Query v.s. Image Database	DCMH	**0.7545**	**0.7574**	**0.7704**
	SePH	0.7127	0.7261	0.7309
	STMH	0.6137	0.6196	0.6221
	SCM	0.6068	0.6089	0.6108
	CMFH	0.5776	0.5792	0.5834
	CCA	0.5628	0.5630	0.5630

4.3.2. Hash Lookup

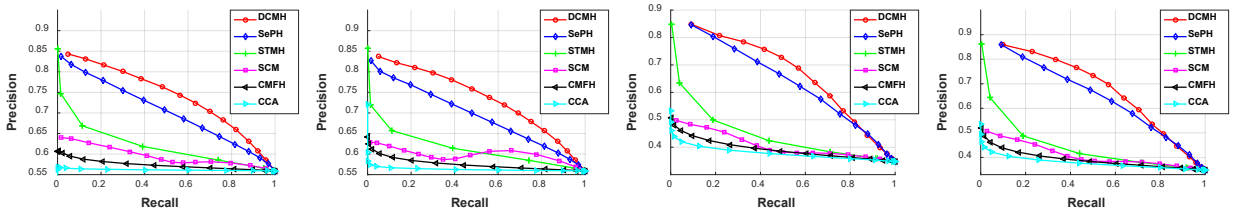
In the hash lookup procedure, we can compute the precision, recall and F-measure for the returned points given any Hamming radius. The Hamming radius can take the values in $\{0, 1, \ldots, c\}$. By varying the Hamming radius from 0 to c with a stepsize 1, we can get the precision-recall curve.

Figure 2 shows the precision-recall curve with code length 16 on two datasets, where the baselines use hand-drafted features. Here, "Image $\rightarrow$ Text" denotes the case where the query is image and the database is text, and similar notations are used for other cases. We can find that DCMH can dramatically outperform the baselines.



(a) Image → Text @MIRFLICKR-25K (b) Text → Image @MIRFLICKR-25K (c) Image → Text @NUS-WIDE (d) Text → Image @NUS-WIDE

Figure 2. Precision-recall curves. The baselines are based on hand-crafted features. The code length is 16.



(a) Image → Text @MIRFLICKR-25K (b) Text → Image @MIRFLICKR-25K (c) Image → Text @NUS-WIDE (d) Text → Image @NUS-WIDE

Figure 3. Precision-recall curves. The baselines are based on CNN-F features. The code length is 16.

Table 6. Comparison to baselines with CNN-F features on *NUS-WIDE* in terms of MAP. The best accuracy is shown in boldface.

Task	Method	code length		
		16 bits	32 bits	64 bits
Image Query v.s. Text Database	DCMH	0.6249	0.6355	0.6438
	SePH	**0.6538**	**0.6668**	**0.6720**
	SePH-linear	0.6179	0.6306	0.6411
	STMH	0.5054	0.5277	0.5248
	SCM	0.4867	0.4958	0.4724
	CMFH	0.3997	0.3889	0.3972
	CCA	0.3783	0.3704	0.3649
Text Query v.s. Image Database	DCMH	**0.6791**	**0.6829**	**0.6906**
	SePH	0.6600	0.6676	0.6854
	SePH-linear	0.6369	0.6470	0.6551
	STMH	0.4717	0.5013	0.4993
	SCM	0.4280	0.4374	0.4090
	CMFH	0.3902	0.3870	0.3910
	CCA	0.3788	0.3705	0.3651

Figure 3 shows the precision-recall curve with code length 16 on two datasets, where the baselines use CNN-F features. We can also find that DCMH can outperform all the other baselines with CNN-F features.

We select the best three methods and report their precision, recall and F-measure with Hamming radius $r = 0, 1, 2$ in Table 7 on MIRFLICKR-25K when the code length is 16, where "I" denotes image and "T" denotes text. We can find that in all cases our DCMH can achieve the best recall and F-measure within Hamming radius $r = 0, 1, 2$. For precision, DCMH outperforms SePH in all cases, but is outperformed by STMH in most cases. However, this does not mean that STMH is better than DCMH, because the recall of STMH is very poor. For example, assume there are 10,000 ground-truth similar points for a query on MIRFLICKR-25K. If we use an image query to retrieve text database with a Hamming radius 0, STMH only returns 3 points. However, our DCMH method can return nearly 580 points and 487 of them are ground-truth similar points. Hence, DCMH is more practical than STMH in real applications. From this perspective, F-measure is a more meaningful metric than precision and recall in the hash lookup procedure, and our DCMH achieves the best F-measure on all cases.

Please note that we only report the results when the code length is 16 due to space limitation. Our DCMH can also achieve the best performance on other cases with different number of code length. Furthermore, our DCMH is not sensitive to hyper-parameters γ and η when they are from the range $[0.5, 2]$. All these experiments can be found in the supplementary materials.

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 learning framework which can perform feature learning from scratch. To the best of our knowledge, DCMH is the first cross-modal hashing method which can perform simultaneous feature learning and hash-code learning in the same framework. Experiments on two datasets show that DCMH can outperform other baselines to achieve the state-of-the-art performance in real applications.

Table 7. Precision, recall and F-measure on *MIRFLICKR-25K* with CNN-F features. The best F-measure is shown in boldface.

Task	Method	Metric	Hamming Radius		
			$r = 0$	$r = 1$	$r = 2$
I → T	DCMH	Precision	0.8434	0.8316	0.8166
		Recall	0.0487	0.1281	0.2121
		F-measure	**0.0920**	**0.2220**	**0.3367**
	SePH	Precision	0.8373	0.8182	0.7985
		Recall	0.0166	0.0620	0.1286
		F-measure	0.0325	0.1153	0.2215
	STMH	Precision	0.8560	0.8560	0.7473
		Recall	0.0003	0.0003	0.0147
		F-measure	0.0007	0.0007	0.0287
T → I	DCMH	Precision	0.8370	0.8220	0.8106
		Recall	0.0504	0.1329	0.2164
		F-measure	**0.0951**	**0.2288**	**0.3416**
	SePH	Precision	0.8273	0.8014	0.7853
		Recall	0.0151	0.0566	0.1193
		F-measure	0.0297	0.1058	0.2072
	STMH	Precision	0.8578	0.8578	0.7194
		Recall	0.0003	0.0003	0.0134
		F-measure	0.0006	0.0006	0.0264

References

Andoni, Alexandr and Indyk, Piotr. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. *Communication of ACM*, 51(1):117–122, 2008.

Andoni, Alexandr and Razenshteyn, Ilya. Optimal data-dependent hashing for approximate near neighbors. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing*, pp. 793–801, 2015.

Andoni, Alexandr, Indyk, Piotr, Laarhoven, Thijs, Razenshteyn, Ilya P., and Schmidt, Ludwig. Practical and optimal LSH for angular distance. In *Proceedings of the Advances in Neural Information Processing Systems*, 2015.

Bronstein, Michael M., Bronstein, Alexander M., Michel, Fabrice, and Paragios, Nikos. Data fusion through cross-modality metric learning using similarity-sensitive hashing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3594–3601, 2010.

Chatfield, Ken, Simonyan, Karen, Vedaldi, Andrea, and Zisserman, Andrew. Return of the devil in the details: Delving deep into convolutional nets. In *Proceedings of the British Machine Vision Conference*, 2014.

Chua, Tat-Seng, Tang, Jinhui, Hong, Richang, Li, Haojie, Luo, Zhiping, and Zheng, Yantao. NUS-WIDE: a real-world web image database from national university of singapore. In *Proceedings of the 8th ACM International Conference on Image and Video Retrieval*, 2009.

Ding, Guiguang, Guo, Yuchen, and Zhou, Jile. Collective matrix factorization hashing for multimodal data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2083–2090, 2014.

Hotelling, Harold. Relations between two sets of variates. *Biometrika*, pp. 321–377, 1936.

Huiskes, Mark J. and Lew, Michael S. The mir flickr retrieval evaluation. In *Proceedings of the ACM International Conference on Multimedia Information Retrieval*, 2008.

Krizhevsky, Alex, Sutskever, Ilya, and Hinton, Geoffrey E. Imagenet classification with deep convolutional neural networks. In *Proceedings of the Advances in Neural Information Processing Systems*, pp. 1106–1114, 2012.

Kumar, Shaishav and Udupa, Raghavendra. Learning hash functions for cross-view similarity search. In *Proceedings of the 22th International Joint Conference on Artificial Intelligence*, pp. 1360–1365, 2011.

LeCun, Yann, Boser, Bernhard E., Denker, John S., Henderson, Donnie, Howard, R. E., Hubbard, Wayne E., and Jackel, Lawrence D. Handwritten digit recognition with a back-propagation network. In *Proceedings of the Advances in Neural Information Processing Systems*, pp. 396–404, 1989.

Leng, Cong, Wu, Jiaxiang, Cheng, Jian, Zhang, Xi, and Lu, Hanqing. Hashing for distributed data. In *Proceedings of the 32nd International Conference on Machine Learning*, pp. 1642–1650, 2015.

Lin, Zijia, Ding, Guiguang, Hu, Mingqing, and Wang, Jianmin. Semantics-preserving hashing for cross-view retrieval. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3864–3872, 2015.

Liong, Venice Erin, Lu, Jiwen, Wang, Gang, Moulin, Pierre, and Zhou, Jie. Deep hashing for compact binary codes learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2475–2483, 2015.

Liu, Wei, Wang, Jun, Kumar, Sanjiv, and Chang, Shih-Fu. Hashing with graphs. In *Proceedings of the 28th International Conference on Machine Learning*, pp. 1–8, 2011.

Liu, Wei, Mu, Cun, Kumar, Sanjiv, and Chang, Shih-Fu. Discrete graph hashing. In *Proceedings of the Advances in Neural Information Processing Systems*, pp. 3419–3427, 2014.

Neyshabur, Behnam and Srebro, Nathan. On symmetric and asymmetric lshs for inner product search. In *Proceedings of the 32nd International Conference on Machine Learning*, pp. 1926–1934, 2015.

Norouzi, Mohammad and Fleet, David J. Minimal loss hashing for compact binary codes. In *Proceedings of the 28th International Conference on Machine Learning*, pp. 353–360, 2011.

Norouzi, Mohammad, Fleet, David J., and Salakhutdinov, Ruslan. Hamming distance metric learning. In *Proceedings of the Advances in Neural Information Processing Systems*, pp. 1070–1078, 2012.

Raginsky, Maxim and Lazebnik, Svetlana. Locality-sensitive binary codes from shift-invariant kernels. In *Proceedings of the Advances in Neural Information Processing Systems*, pp. 1509–1517, 2009.

Rastegari, Mohammad, Choi, Jonghyun, Fakhraei, Shobeir, III, Hal Daumé, and Davis, Larry S. Predictable dual-view hashing. In *Proceedings of the 30th International Conference on Machine Learning*, pp. 1328–1336, 2013.

Russakovsky, Olga, Deng, Jia, Su, Hao, Krause, Jonathan, Satheesh, Sanjeev, Ma, Sean, Huang, Zhiheng, Karpathy, Andrej, Khosla, Aditya, Bernstein, Michael S., Berg, Alexander C., and Li, Fei-Fei. Imagenet large scale visual recognition challenge. *CoRR*, abs/1409.0575, 2014.

Shrivastava, Anshumali and Li, Ping. Densifying one permutation hashing via rotation for fast near neighbor search. In *Proceedings of the 31th International Conference on Machine Learning*, pp. 557–565, 2014.

Song, Jingkuan, Yang, Yi, Huang, Zi, Shen, Heng Tao, and Hong, Richang. Multiple feature hashing for real-time large scale near-duplicate video retrieval. In *Proceedings of the 19th ACM Conference on Multimedia*, pp. 423–432, 2011.

Vedaldi, Andrea and Lenc, Karel. Matconvnet: Convolutional neural networks for MATLAB. In *Proceedings of the ACM Conference on Multimedia*, pp. 689–692, 2015.

Wang, Di, Gao, Xinbo, Wang, Xiumei, and He, Lihuo. Semantic topic multimodal hashing for cross-media retrieval. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence*, pp. 3890–3896, 2015.

Wang, Jun, Kumar, Sanjiv, and Chang, Shih-Fu. Sequential projection learning for hashing with compact codes. In *Proceedings of the 27th International Conference on Machine Learning*, pp. 1127–1134, 2010.

Weiss, Yair, Torralba, Antonio, and Fergus, Robert. Spectral hashing. In *Proceedings of the Advances in Neural Information Processing Systems*, pp. 1753–1760, 2008.

Yu, Felix X., Kumar, Sanjiv, Gong, Yunchao, and Chang, Shih-Fu. Circulant binary embedding. In *Proceedings of the 31th International Conference on Machine Learning*, pp. 946–954, 2014.

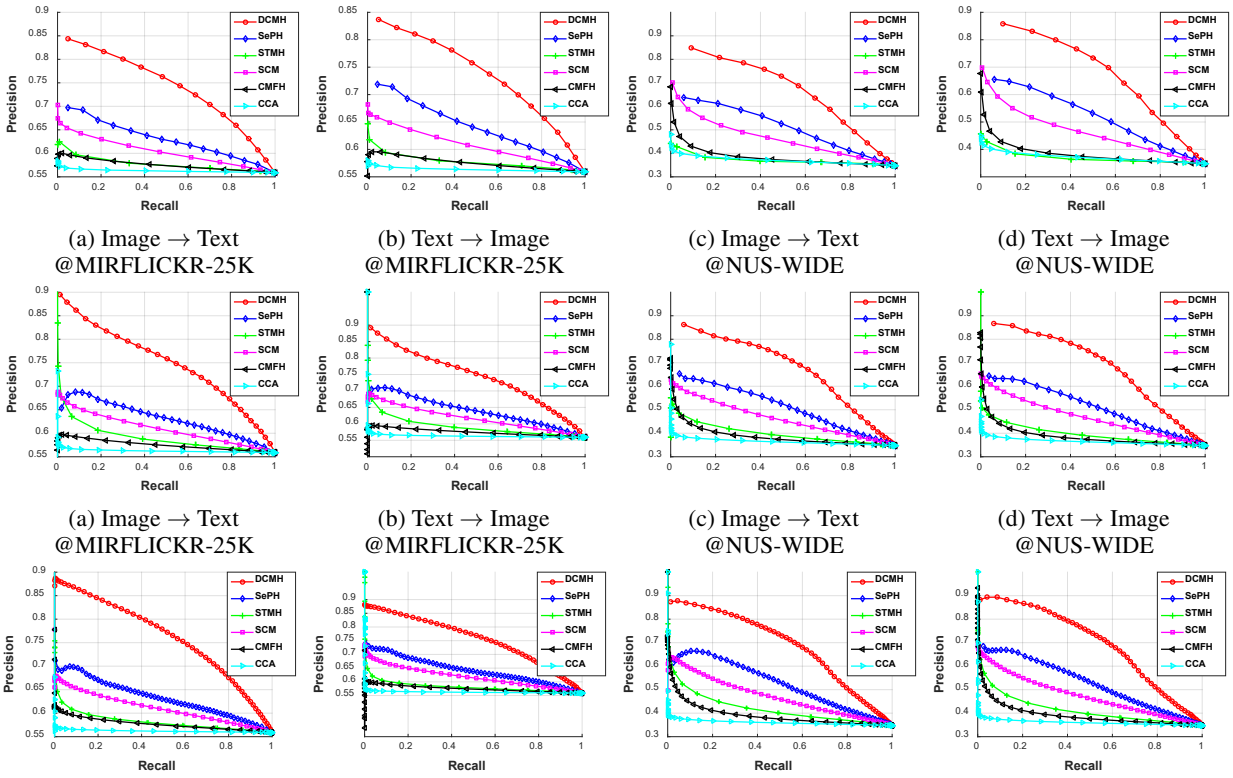
Zhang, Dan, Wang, Fei, and Si, Luo. Composite hashing with multiple information sources. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 225–234, 2011.

Zhang, Dongqing and Li, Wu-Jun. Large-scale supervised multimodal hashing with semantic correlation maximization. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence*, pp. 2177–2183, 2014.

Zhao, Fang, Huang, Yongzhen, Wang, Liang, and Tan, Tieniu. Deep semantic ranking based hashing for multi-label image retrieval. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1556–1564, 2015.

Zhen, Yi and Yeung, Dit-Yan. A probabilistic model for multimodal hash function learning. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 940–948, 2012a.

Zhen, Yi and Yeung, Dit-Yan. Co-regularized hashing for multimodal data. In *Proceedings of the Advances in Neural Information Processing Systems*, pp. 1385–1393, 2012b.

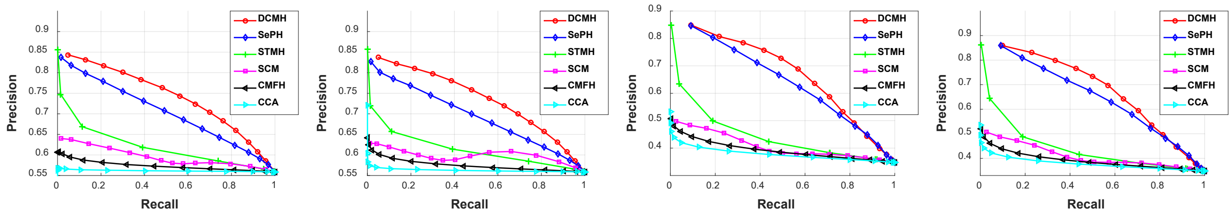


(a) Image → Text @MIRFLICKR-25K (b) Text → Image @MIRFLICKR-25K (c) Image → Text @NUS-WIDE (d) Text → Image @NUS-WIDE

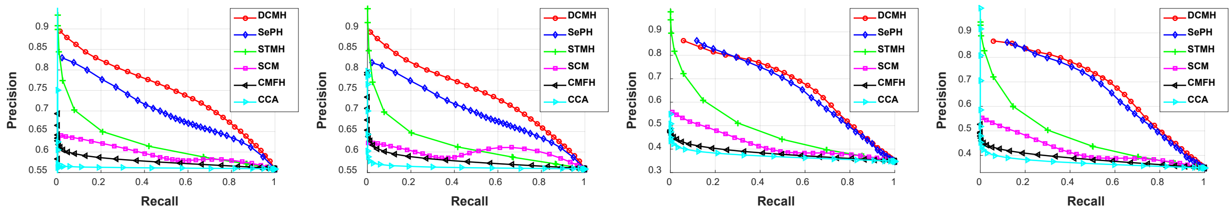
(a) Image → Text @MIRFLICKR-25K (b) Text → Image @MIRFLICKR-25K (c) Image → Text @NUS-WIDE (d) Text → Image @NUS-WIDE

(a) Image → Text @MIRFLICKR-25K (b) Text → Image @MIRFLICKR-25K (c) Image → Text @NUS-WIDE (d) Text → Image @NUS-WIDE

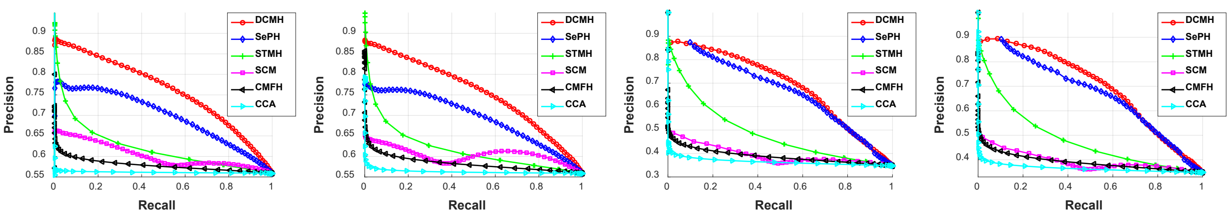
Figure 4. Precision-recall curves. The baselines are based on hand-crafted features. The first row is for 16 bits, the second row is for 32 bits, and the third row is for 64 bits.



(a) Image → Text @MIRFLICKR-25K (b) Text → Image @MIRFLICKR-25K (c) Image → Text @NUS-WIDE (d) Text → Image @NUS-WIDE

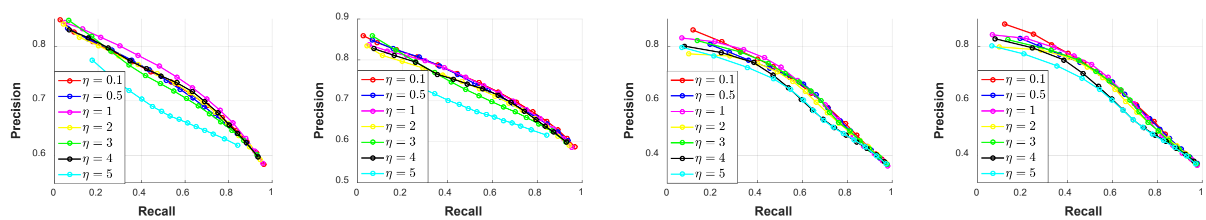


(a) Image → Text @MIRFLICKR-25K (b) Text → Image @MIRFLICKR-25K (c) Image → Text @NUS-WIDE (d) Text → Image @NUS-WIDE



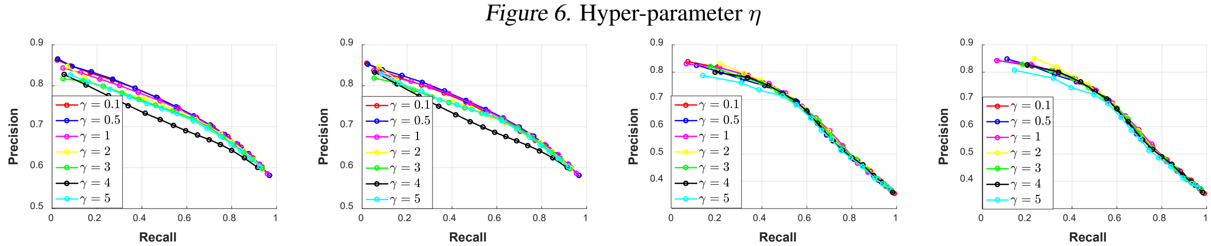
(a) Image → Text @MIRFLICKR-25K (b) Text → Image @MIRFLICKR-25K (c) Image → Text @NUS-WIDE (d) Text → Image @NUS-WIDE

Figure 5. Precision-recall curves. The baselines are based on CNN-F features. The first row is for 16 bits, the second row is for 32 bits, and the third row is for 64 bits.



(a) Image → Text @MIRFLICKR-25K (b) Text → Image @MIRFLICKR-25K (c) Image → Text @NUS-WIDE (d) Text → Image @NUS-WIDE

Figure 6. Hyper-parameter η



(a) Image → Text @MIRFLICKR-25K (b) Text → Image @MIRFLICKR-25K (c) Image → Text @NUS-WIDE (d) Text → Image @NUS-WIDE

Figure 7. Hyper-parameter γ