Asymmetric Deep Cross-modal Hashing

Jingzi Gu[1,2], JinChao Zhang[1](✉), Zheng Lin[1], Bo Li[1], Weiping Wang[1], and Dan Meng[1]

[1] Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
{gujingzi,zhangjinchao,linzheng,libo,wangweiping,mengdan}@iie.ac.cn
[2] School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China

Abstract. Cross-modal retrieval has attracted increasing attention in recent years. Deep supervised hashing methods have been widely used for cross-modal similarity retrieval on large-scale datasets, because the deep architectures can generate more discriminative feature representations. Traditional hash methods adopt a symmetric way to learn the hash function for both query points and database points. However, those methods take an immense amount of work and time for model training, which is inefficient with the explosive growth of data volume. To solve this issue, an Asymmetric Deep Cross-modal Hashing (ADCH) method is proposed to perform more effective hash learning by simultaneously preserving the semantic similarity and the underlying data structures. More specifically, ADCH treats the query points and database points in an asymmetric way. Furthermore, to provide more similarity information, a detailed definition for cross-modal similarity matrix is also proposed. The training of ADCH takes less time than traditional symmetric deep supervised hashing methods. Extensive experiments on two widely used datasets show that the proposed approach achieves the state-of-the-art performance in cross-modal retrieval.

Keywords: Asymmetric hashing · Cross-modal · Retrieval

1 Introduction

A tremendous amount of data in heterogeneous modalities are being generated every day on the Internet, including image, text, audio, etc. Multimedia retrieval has been an essential technique in many applications. However, essential retrieval methods mainly focus on single-modal scenarios [1,2]. For example, image retrieval or text retrieval are homogeneous-modal, in which the query and result are from the same modality. These methods cannot directly measure the similarity between different modalities. Thus, effective retrieval of such massive amounts of media data from heterogeneous sources poses a great challenge.

Retrieval across multimedia data [3] is a relatively new paradigm. Recently, deep hashing methods [5] have been used in cross-modal retrieval. Hashing methods map high-dimensional representations in the original space to short binary codes in the Hamming space, which can bridge the “heterogeneity gap” between multimedia. For example, deep cross-modal hashing (DCMH) [7] integrates feature learning and hash-code learning into the same framework. Besides, collective deep quantization for efficient cross-modal retrieval (CDQ) [8] introduces quantization in end-to-end deep architecture for cross-modal retrieval.

However, the related previous methods adopt a symmetric strategy to learn deep hash function for both query points and database points. On one hand, the training of these symmetric deep hashing methods are typically time-consuming. The storage and computation of these data cost even more time. To make the training feasible, most deep hashing methods choose simple small datasets or subsets of a large dataset, which make it difficult to utilize the information adequately. On the other hand, these traditional supervised deep hashing methods measure the similarity of image-text using the semantic-level labels and define the similarity in a certain way. However, such similarity definition cannot reflect the similarity in detail on multiple labels datasets.

To solve these problems, an Asymmetric Deep Cross-modal Hashing (ADCH) method is proposed which treats the query points and database points in an asymmetric way. The binary hash codes of query points can be obtained from a deep hash function which is learned in this method, while the binary hash codes of database points are directly learned. The training of ADCH takes less time than traditional symmetric deep supervised hashing methods, because the training points are only query points that are much fewer than all of the databases points. Hence, the whole set of database points can be used for training even if the database is large. Furthermore, a detailed definition is proposed that the similarities between points are quantified into a percentage, which can provide more label information. In this paper, the main contributions of ADCH are outlined as follows:

(1) A novel method ADCH is proposed which learns cross-modal hash codes in an asymmetric way. ADCH generates hash codes for database points directly, while hash functions are only for query points. Therefore, it takes less training time than symmetric deep supervised hash methods.
(2) A detailed definition is proposed for cross-modal similarity matrix to make ADCH use more similarity information. The detailed definition of similarity matrix quantifies the similarity into a percentage with the normalized semantic labels, which provide more fine-grained information of labels in the loss function. Thus, it can improve the retrieval accuracy.
(3) ADCH takes less time and achieves high accuracy than the traditional symmetric deep supervised hashing methods. Experiments on two large-scale datasets show that ADCH can achieve the state-of-the-art performance.

2 Related Work

There are many traditional hashing methods [9,11] in cross-modal retrieval, which do not use deep networks. However, deep learning has shown its strength in modeling nonlinear correlation, and has achieved state-of-the-art performance in single-modal scenarios. Therefore, cross-modal hashing with deep methods has been proposed to meet retrieval demands in large-scale cross-modal databases.

Deep cross-modal hashing methods can be further divided into the following two categories: unsupervised and supervised hashing. Unsupervised cross-modal hashing methods map unlabeled input data into hash codes by learning a hash function. Zhang et al. make full use of Generative Adversarial Networks for unsupervised representation learning to exploit the underlying manifold structure of cross-modal data [4]. Even though unsupervised methods can get good performance, they still cannot satisfy the demanded accuracy of image retrieval. Therefore, lots of supervised methods were proposed to improve retrieval accuracy. Specifically, supervised cross-modal deep hashing methods [5,23] learn the hash function with supervised information.

The deep supervised architectures mainly include two ways in cross-modal retrieval. The first way is that inputs of different modal types pass through the same shared layer [12,13] to extract a unified representation that fuses modalities together, while the second way is that each modal passes through a sub-network and the output of these sub-networks are coupled by correlation constraints at the code layers [14,15]. For example, Cao et al. propose deep visual semantic hashing (DSVH) [6] model that generates compact hash codes of images and sentences in an end-to-end deep learning architecture, which captures the intrinsic cross-modal correspondences between visual data and a natural language. Wang [5] proposed an online learning method to learn the similarity function between heterogeneous modalities by preserving the relative similarity in the training data, which is modeled as a set of bidirectional hinge loss constraints on the cross-modal training triplets.

However, the related previous methods adopt a symmetric strategy to learn deep hash functions for both query points and database points. It is time-consuming to train data using these methods. Thus, most deep hashing methods choose simple small datasets or subsets of a large dataset. Unlike previous work, ADCH method is proposed which treats the query points and database points in an asymmetric way. It takes less time and achieves a better accuracy than the traditional symmetric deep supervised hashing methods.

3 Asymmetric Deep Cross-modal Hashing

3.1 Problem Definition

Although ADCH can be used in more than two modalities, we only focus on image and text for simplicity. Each point in the dataset has two feature modalities. Assume there is a query set image-text data-pair $\mathbf{P} = \{(x_i, y_i)\}_{i=1}^{m}$, in which m is the number of query data points, and a database set $\mathbf{D} = \{(x_j, y_j)\}_{j=1}^{n}$, in which n is the number of database points. In addition, we give a cross-modal similarity matrix $\mathbf{S}$ and it would be written as $\mathbf{S} = [S_{ij}]$ next. If $S_{ij} = 1$, $\{(x_i, y_i)\} \in P$ and $\{(x_j, y_j)\} \in D$ are similar; otherwise normally $S_{ij} = 0$, they are not similar. The matrix $\mathbf{S}$ is defined by semantic information such as class labels. For example, if image x_i and text y_j have the same label, then they are similar. Otherwise, they are dissimilar.

The goal of cross-modal hashing is to learn hash codes for both image and text. The hash codes preserve the semantic information of image and text. We learn two hash functions for query points: $f(x_i)$ for the image modality and $g(y_i)$ for the text modality. Additionally, $\mathbf{U} = \{(u_i^x, u_i^y)\}_{i=1}^m \in \{-1, +1\}^{m \times c}$ denotes the binary hash codes of query points, generating from $f(x_i)$ and $g(y_i)$. m is the number of query set, and c is the binary hash codes length. $\mathbf{V} = \{(v_j^x, v_j^y)\}_{j=1}^n \in \{-1, +1\}^{n \times c}$ denotes the directly learned binary hash codes of database. n is the number of database points. $\mathbf{U}$ and $\mathbf{V}$ are used to compute the similarity between the query points and the database points. The Hamming distance should be small if $S_{ij} = 1$, while the Hamming distance should be large if $S_{ij} = 0$.

In practice, there may be only database points $\mathbf{D} = \{(x_j, y_j)\}_{j=1}^n$ without query points. Hence, we randomly sample m points from the database as the query set. Set $\mathbf{P} = (\mathbf{D})^{\Omega}$, where $(\mathbf{D})^{\Omega}$ denotes the database points indexed by Ω. Here the indices of all the database points are denoted by $\Gamma = \{1, 2, ..., n\}$ and the indices of the sampled query points are denoted by $\Omega = \{i_1, i_2, ..., i_m\} \subseteq \Gamma$. Since $\Omega \subseteq \Gamma$, if a pair of points (x_i, y_i) belongs to Ω, then the points also belong to Γ. Accordingly, setting $\mathbf{S} = S^{\Omega}$, the supervised information (similarity) between pairs of all database points are denoted by $\mathbf{S} \in \{0, 1\}^{n \times n}$. $S^{\Omega} \in \{0, 1\}^{m \times n}$ denotes the submatrix formed by the rows of $\mathbf{S}$ indexed by Ω. The goal of asymmetric cross-modal hashing is that query set $\mathbf{P}$ is learned from the image and text modalities. And database set $\mathbf{D}$ is learned directly by loss function.

3.2 Model Formulation

The whole ADCH model is shown in Fig. 1, which contains the following three components: the image feature learning part, the text feature learning part and the asymmetric function part. In the image feature learning part, an appropriate image feature representation for image binary hash codes learning is extracted by using a deep convolutional neural network. Analogously, the text feature representation is extracted for the text binary hash codes learning in the text feature learning part. In the asymmetric loss part, an asymmetric loss function is proposed to train the end-to-end model.

Unlike previous methods, the query points and database points are treated in an asymmetric way in ADCH. Feature learning is only used to learn hash function for query points and not for database points, then the hash function of ADCH generates hash codes for query points; database points learn binary hash codes directly which is as a variable in the asymmetric loss. Because the training points are only query points, ADCH takes less time than traditional methods.

3.3 Feature Learning

In the ADCH model, feature learning contains the following two parts: image feature learning and text feature learning.

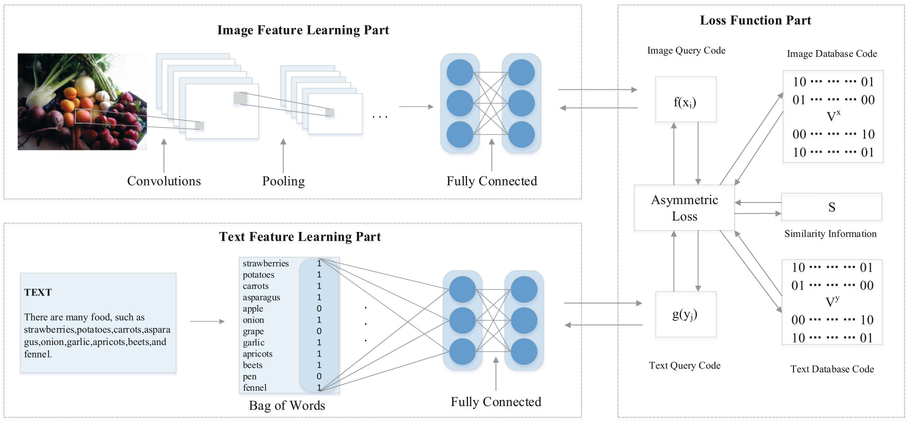

Fig. 1. The architecture of ADCH. The image feature learning part extracts image feature representations and the text feature part extracts text feature representations. The loss function preserves the similarity between query points and database points.

In the image feature learning part, the deep neural network is a convolutional neural network (CNN) model from CNN-F [16] model. In this model, there are eight layers in total, five of which are fully connected convolutional layers. The last layer of the CNN-F model is used to project the output of the first seven layers into R^c space, which represents the learned image features. The details of CNN-F can be found in [16].

In the text feature learning part, the text alternates into bag-of-words (BOW) representation. And the bag-of-words vectors are the input of a deep neural network with two fully-connected layers. The activation function of first layer is a ReLU and the identity function is used for second layers. Then the text representation can be obtained from the last layer of the deep neural network.

3.4 Asymmetric Loss

The hash codes of query points are generated from hash function, while the hash codes of database points are learned directly. Since the hash codes of query points and database points are generated in different manners, they are asymmetric. The basic idea of asymmetric cross-modal hash codes learning is to make the query point and the database point with the same labels as close as possible. In other words, the image-text pair of query points and database points are similar.

Firstly, to obtain the hashing functions $u_i^x = f(x_i)$, $u_i^y = g(y_i)$ and directly learned hashing codes v_j^x, v_j^y, we propose the J_1 loss. There are four kinds of relative similarity, i.e., the relative similarity query image to a database text, query text to a database text, query image to a database image, and query text to a database image. The object is to minimize the loss between the similarity and inner product of query-database binary codes pairs. It can be written as follows:

$$
\begin{aligned}
J_1 = \sum_{i=1}^{m}\sum_{j=1}^{n}[((u_i^x)^T v_j^x - cS_{ij})^2 + ((u_i^x)^T v_j^y - cS_{ij})^2 \\
+ ((u_i^y)^T v_j^x - cS_{ij})^2 + ((u_i^y)^T v_j^y - cS_{ij})^2] \\
s.t.\ x \in I,\ y \in T
\end{aligned} \tag{1}
$$

I is an image set and T is a text set. u_i^x and u_i^y are difficult to learn, because they are discrete data. Thus it sets $u_i^x = f(x_i) = tanh(F(x_i, \Theta_x))$ and $u_i^y = g(y_i) = tanh(G(y_i, \Theta_y))$, which are used to optimize the formulation (1). $F(x_i, \Theta_x) \in R^c$ denotes the learned image feature for point i, which is the output of the CNN-F model. Furthermore, $G(y_i, \Theta_y) \in R^c$ denotes the learned text feature for point i, which is the output of the deep neural network for text modality. Θ_x is the parameter of CNN-F model for image datasets, and Θ_y is the parameter of the deep neural network for text datasets.

Secondly, because the query point and database point are treated as asymmetrically, the hash codes of query points and database points in the same data-pair should be same. Thus, we propose J_2. There are two kinds of relative similarity, i.e., the relative similarity query image to a database image, and query text to a database text. Then J_2 can be defined as:

$$
J_2 = \gamma \sum_{i \in \Omega} [v_i^x - u_i^x]^2 + \gamma \sum_{i \in \Omega} [v_i^y - u_i^y]^2 \tag{2}
$$

where γ is a hyper-parameter.

Finally, the hash codes of image and text are from different neural networks, but they hold the same semantic information. So the hash codes should be correspondent. Thus, J_3 which can be defined as:

$$
J_3 = \eta \sum_{i \in \Omega} [u_i^x - u_i^y]^2 \tag{3}
$$

where η is a hyper-parameter.

By combining the above, we can get an asymmetric loss to perform deep hashing from similarity data by jointly preserving similarity between query points and database points. The asymmetric loss is designed as follows:

$$
\begin{aligned}
\overset{min}{\Theta,\mathbf{V}} J(\Theta, \mathbf{V}) &= J_1 + J_2 + J_3 \\
&= \sum_{i \in \Omega}\sum_{j \in \Gamma} \left((u_i^x)^T v_j^x - cS_{ij}\right)^2 + \left((u_i^x)^T v_j^y - cS_{ij}\right)^2 \\
&+ \left((u_i^y)^T v_j^x - cS_{ij}\right)^2 + \left((u_i^y)^T v_j^y - cS_{ij}\right)^2 \\
&+ \gamma \sum_{i \in \Omega} [v_i^x - (u_i^x)]^2 + \gamma \sum_{i \in \Omega} [v_i^y - (u_i^y)]^2
\end{aligned}
$$

$$+ \eta \sum_{i \in \Omega} [(u_i^x)) - (u_i^y)]^2$$
$$s.t.\ x \in I,\ y \in T, \mathbf{V} = \{v_j^x, v_j^y\}_{j=1}^n \in \{-1, +1\}^{n \times c} \qquad (4)$$

3.5 Detailed Definition of Similarity Matrix

The similarity matrix $\mathbf{S}$ is usually defined as $S_{ij} = 1$ if $\mathbf{P}_i$ and $\mathbf{D}_j$ have the same semantic label, and $S_{ij} = 0$ if $\mathbf{P}_i$ and $\mathbf{D}_j$ do not share any semantic label. This definition does not take the multi-label information into account, and leads to the loss of semantic similarity information.

Therefore, to relax the constraint of S_{ij}, we propose a detailed definition for cross-modal similarity matrix S_{ij}. The detailed definition of similarity matrix quantifies the similarity into a percentage with the normalized semantic labels, which provide more fine-grained information of labels in the loss function. The similarity of text and image can be passed into the following three cases: completely similar, partially similar, and dissimilar. Completely similar is that all the labels are the same in the compared points, partially similar is that some of the labels are same and the similarity is in percentages, and dissimilar is that all the labels are different. Thus, the detailed definition of similarity matrix can improve the retrieval accuracy. So the similarity matrix $\mathbf{S}$ is defined as:

$$S_{ij} = \begin{cases} pc_{ij}\ if\ \mathbf{P}_i\ and\ \mathbf{D}_j\ share\ some\ class\ labels \\ 0\quad otherwise \end{cases}$$

where pc_{ij} is the cosine distance of pairwise label vectors, which is formulated as:

$$pc_{ij} = \frac{< {l_i}^T, l_j >}{||l_i||||l_j||}$$

where l_i and l_j denote the semantic label vector of text and image $\mathbf{P}_i$ and $\mathbf{D}_j$, respectively, and $< {l_i}^T, l_j >$ calculates the inner product.

In the query procedure, we can use the functions $f(x_i) = sign(F(x_i, \Theta_x))$ and $g(y_i) = sign(G(y_i, \Theta_y))$ to generate binary hash codes. Then we can use hamming distance to compute the distance between query point and database point.

3.6 Learning Algorithm

We adopt an alternating learning strategy to learn $\Theta_x, \Theta_y, \mathbf{V}^x and\ \mathbf{V}^y$. That is, in each iteration, one parameter is learned, and other parameters are fixed. Then, after several iterations, the training procedure finishes. In this paper, we repeat the learning for several times, and each time, we can sample a query set from Ω. The detailed derivation will be introduced in the following content of this subsection.

Learn the Parameter of Image Modality. When Θ_y, $\mathbf{V}^x$ and $\mathbf{V}^y$ are fixed, we learn the CNN-F parameter Θ_x of the image modality by using a back-propagation (BP) algorithm. More specifically, we sample a mini-batch of the query points and update the parameter Θ_x. To simplify the formula, $z_i^x = F(x_i, \Theta_x)$, $z_i^y = G(y_i, \Theta_y)$, and $\odot$ denotes the dot product. We compute the derivative of the cost function with respect to z_i^x.

$$\frac{\partial J}{\partial z_i^x} = \left\{ \begin{array}{l} 2\sum\limits_{j\in\Gamma}\left[\left((u_i^x)^T v_j^y - cS_{ij}\right) v_j^y\right] \\ +2\sum\limits_{j\in\Gamma}\left[\left((u_i^x)^T v_j^x - cS_{ij}\right) v_j^x\right] \\ +2\gamma\left(u_i^x - v_i^x\right) + 2\eta\left(u_i^x - u_i^y\right) \end{array} \right\} \odot \left(1 - {u_i^x}^2\right) \tag{5}$$

Then we can compute $\frac{\partial J}{\partial \Theta_x}$ with $\frac{\partial J}{\partial z_i^x}$ by using the chain rule, based on BP algorithm to update the parameter Θ_x.

Learn the Parameter of Text Modality. When Θ_x, $\mathbf{V}^x$ and $\mathbf{V}^y$ are fixed, we also learn the parameter Θ_y of the text modality with a back-propagation (BP) algorithm. As simplified as above, we compute the gradient of z_i^y. We can compute $\frac{\partial J}{\partial \Theta_y}$ with $\frac{\partial J}{\partial z_i^y}$ by using the chain rule. We can get the parameter Θ_y using the same method as calculating Θ_x.

Learn the Binary Codes of Image Database. When Θ_x, Θ_y and $\mathbf{V}^y$ are fixed, we rewrite the problem (4). $\mathbf{U}^x = [u_{i1}^x, u_{i2}^x, \ldots, u_{im}^x]^T \in [-1,+1]^{m\times c}$, $\mathbf{U}^y = [u_{i1}^y, u_{i2}^y, \ldots, u_{im}^y]^T \in [-1,+1]^{m\times c}$, and $(\mathbf{V}^x)^\Omega = [v_{i1}^x, v_{i2}^x, \ldots, v_{im}^x]^T$. $(\mathbf{V}^x)^\Omega$ represents the binary codes for the database which is in set Ω.

$$\begin{aligned} \min_{\mathbf{V}^x} J(\mathbf{V}^x) = & \parallel \mathbf{U}^x(\mathbf{V}^x)^T - c\mathbf{S} \parallel_F^2 + \parallel \mathbf{U}^y(\mathbf{V}^x)^T - c\mathbf{S} \parallel_F^2 \\ & + \gamma \parallel (\mathbf{V}^x)^\Omega - \mathbf{U}^x \parallel_F^2 + L \\ = & \parallel \mathbf{U}^x(\mathbf{V}^x)^T \parallel_F^2 + \parallel \mathbf{U}^y(\mathbf{V}^x)^T \parallel_F^2 - 2ctr((\mathbf{V}^x)^T\mathbf{S}^T\mathbf{U}^x) \\ & - 2c\mathrm{tr}((\mathbf{V}^x)^T\mathbf{S}^T\mathbf{U}^y) - 2\gamma\mathrm{tr}((\mathbf{V}^x)^\Omega(\mathbf{U}^x)^T) + L \end{aligned} \tag{6}$$

where L is a constant independent of $\mathbf{V}^x$. We define $\bar{\mathbf{U}}^x = \{\bar{u}_j^x\}_{j=1}^n$, where the definition is as follows:

$$\bar{u}_j^x = \begin{cases} u_j^x \ if \ j \in \Omega \\ 0 \ otherwise \end{cases}$$

Then we can rewrite the problem (6) as follows:

$$\begin{aligned} \min_{\mathbf{V}^x} J(\mathbf{V}^x) = & \parallel \mathbf{V}^x(\mathbf{U}^x)^T \parallel_F^2 + \parallel \mathbf{V}^x(\mathbf{U}^y)^T \parallel_F^2 + L \\ & - 2\mathrm{t}r(\mathbf{V}^x[c(\mathbf{U}^x)^T\mathbf{S} + c(\mathbf{U}^y)^T\mathbf{S} + \gamma(\bar{\mathbf{U}}^x)^T]) \\ = & \parallel \mathbf{V}^x(\mathbf{U}^x)^T \parallel_F^2 + \parallel \mathbf{V}^x(\mathbf{U}^y)^T \parallel_F^2 + \mathrm{t}r(\mathbf{V}^x(\mathbf{Q}^x)^T) + L \end{aligned} \tag{7}$$

We set $\mathbf{Q}^x = -2(c\mathbf{S}^T\mathbf{U}^{\mathrm{x}} + c\mathbf{S}^T\mathbf{U}^y + \gamma(\bar{\mathbf{U}}^x))$. Every time we update one column of $\mathbf{V}^x$ when other columns are fixed. $\mathbf{V}^x_{*k}$ denotes the *k-th* column of $\mathbf{V}^x$, $\hat{\mathbf{V}}^x_k$ denotes the matrix of $\mathbf{V}^x$ excluding $\mathbf{V}^x_{*k}$. We set $\mathbf{Q}^x_{*k}$ to denote the *k-th* column of $\mathbf{Q}^x$. Let $\mathbf{U}^x_{*k}$ denotes the *k-th* column of $\mathbf{U}^x$ and $\hat{\mathbf{U}}^x_k$ denotes the matrix of $\mathbf{U}^x$ excluding $\mathbf{U}^x_{*k}$. $\mathbf{V}^x_{*k} \in \{-1, +1\}^c$. $\mathbf{U}^y_{*k}$ denotes the *k-th* column of $\mathbf{U}^y$, and $\hat{\mathbf{U}}^y_k$ denotes the matrix of $\mathbf{U}^{\mathrm{y}}$ excluding the k-th column. To optimize $\mathbf{V}^x_{*k}$, we can obtain the problem as follows:

$$\begin{aligned} J(\mathbf{V}^x_{*k}) &= \| \mathbf{V}^x(\mathbf{U}^x)^T \|_F^2 + \| \mathbf{V}^x(\mathbf{U}^y)^T \|_F^2 + tr(\mathbf{V}^x(\mathbf{Q}^x)^T) + L \\ &= tr(\mathbf{V}^x_{*k}[2(\mathbf{U}^x_{*k})^T\hat{\mathbf{U}}^x_k(\hat{\mathbf{V}}^x_k)^T + 2(\mathbf{U}^y_{*k})^T\hat{\mathbf{U}}^y_k(\hat{\mathbf{V}}^x_k)^T + (\mathbf{Q}^x_{*k})^T]) + L \end{aligned} \quad (8)$$

Then, we must minimize $J(\mathbf{V}^x_{*k})$, so we obtain the problem (9) as follows:

$$\min_{\mathbf{V}^x_{*k}} J(\mathbf{V}^x_{*k}) = tr(\mathbf{V}^x_{*k}[2(\mathbf{U}^x_{*k})^T\hat{\mathbf{U}}^x_k(\hat{\mathbf{V}}^x_k)^T + 2(\mathbf{U}^y_{*k})^T\hat{\mathbf{U}}^y_k(\hat{\mathbf{V}}^x_k)^T + (\mathbf{Q}_{*k}{}^x)^T]) + L \quad (9)$$

Then, an optimal solution of problem (10) can be get as follows:

$$J(\mathbf{V}^x_{*k}) = -sign(2(\mathbf{U}^x_{*k})^T\hat{\mathbf{U}}^x_k(\hat{\mathbf{V}}^x_k)^T + 2(\mathbf{U}^y_{*k})^T\hat{\mathbf{U}}^y_k(\hat{\mathbf{V}}^x_k)^T + (\mathbf{Q}_{*k}{}^x)^T) \quad (10)$$

Learn the Binary Codes of Text Database. As the same as $J(\mathbf{V}^x_{*k})$, we can get $J(\mathbf{V}^y_{*k})$ as follows:

$$J(\mathbf{V}^y_{*k}) = -sign(2(\mathbf{U}^x_{*k})^T\hat{\mathbf{U}}^x_k(\hat{\mathbf{V}}^y_k)^T + 2(\mathbf{U}^y_{*k})^T\hat{\mathbf{U}}^y_k(\hat{\mathbf{V}}^y_k)^T + (\mathbf{Q}_{*k}{}^y)^T) \quad (11)$$

$\mathbf{V}^y_{*k}$ denotes the *k-th* column of $\mathbf{V}^y$, and $\hat{\mathbf{V}}^y_k$ denotes the matrix of $\mathbf{V}^y$ excluding $\mathbf{V}^y_{*k}$, $\mathbf{Q}^y_{*k}$ denotes the *k*-th column of $\mathbf{Q}^y$, $\mathbf{U}^y_{*k}$ denotes the *k-th* column of $\mathbf{U}^y$ and $\hat{\mathbf{U}}^y_k$ denotes the matrix of $\mathbf{U}^y$ excluding $\mathbf{U}^y_{*k}$.

Using the above method, we can get the image and text hash functions and database hash codes. The hash functions are used to generate the hash codes of query points, and database hash codes are used to index by query points.

4 Experiment

4.1 Datasets

In the experiments, we carry out cross-modal hashing on two widely-used datasets: MIRFLICKR-25K [17] and NUS-WIDE [18]. The MIRFLICKR-25K dataset consists of 25,000 images. Following the prior work (DCMH [11]), we only select the image-text pairs that have at least 20 textual tags. We randomly select 2000 points as test set and the remaining points as database set. The NUS-WIDE is a public web image dataset which contains 260648 images. There are 81 concept labels in the dataset and one or multiple labels in each point. We select 195834 image-text pairs that belong to the 21 most frequent concepts. We randomly select 2100 points as test set and the remaining points as database set.

4.2 Evaluation Protocol and Baselines

To evaluate ADCH, we choose three metric methods. Firstly, the mean average precision (MAP) is a widely used metric to measure the accuracy of the Hamming ranking protocol. Secondly, we can compute the precision and recall for the returned points given any Hamming radius. Finally, in this paper, we also compare the training time between hashing methods.

We compare our ADCH with eight state-of-the-art methods, including several shallow-structure-based methods (CCA [9], CMFH [22], STMH [20], SCM [21], SePH [19]) and several deep-structure-based methods (DCMH [9], CHN [10], SSAH [22]). These methods are all symmetric models.

For ADCH, in dataset MIRFLICKR-25K we set $\gamma = 200$, $T_{out} = 50$, $T_{in} = 3$ and $\Omega = 2000$ and in dataset NUS-WIDE we set $\gamma = 200$, $T_{out} = 50$, $T_{in} = 3$, $\Omega = 2100$. T_{in} is the iteration number of randomly generating index set Ω from Γ, and T_{out} is the total iteration number. For ADCH, the number of query points m will be much fewer than n. We use the CNN-F network pretrained on the ImageNet dataset to initialize the image modality.

Table 1. The MAP results of baselines and ADCH on MIRFLICKE-25K and NUS-WIDE datasets.

Task	Method	MIRFLICKE-25K			NUS-WIDE		
		16 bits	24 bits	32 bits	16 bits	24 bits	32 bits
$I \rightarrow T$	CCA†	0.5719	0.5693	0.5672	0.3604	0.3485	0.3390
	CMFH†	0.6377	0.6418	0.6451	0.4900	0.5053	0.5095
	SCM†	0.6851	0.6921	0.7003	0.5409	0.5485	0.5553
	STMH†	0.6132	0.6219	0.6274	0.4710	0.4864	0.4942
	SePH†	0.7123	0.7194	0.7232	0.6037	0.6136	0.6211
	DCMH	0.7510	0.7425	0.7471	0.5922	0.6125	0.6108
	CHN	0.7531	0.7673	0.7721	0.6045	0.6187	0.6321
	SSAH	0.7732	0.7894	0.8045	0.6494	0.6312	0.6488
	ADCH	**0.8878**	**0.9011**	**0.9041**	**0.8197**	**0.8192**	**0.8196**
$T \rightarrow I$	CCA†	0.5742	0.5713	0.5691	0.3614	0.3494	0.3395
	CMFH†	0.6365	0.6399	0.6429	0.5031	0.5187	0.5225
	SCM†	0.6939	0.7012	0.7060	0.5344	0.5412	0.5484
	STMH†	0.6074	0.6153	0.6217	0.4471	0.4677	0.4780
	SePH†	0.7216	0.7261	0.7319	0.5983	0.6025	0.6109
	DCMH	0.7727	0.7800	0.7832	0.6139	0.6611	0.6671
	CHN	0.7739	0.7847	0.7956	0.6182	0.6479	0.6455
	SSAH	0.7921	0.7942	0.8067	0.6790	0.6655	0.679
	ADCH	**0.8796**	**0.8864**	**0.8864**	**0.8034**	**0.8022**	**0.8036**

Table 2. The MAP Comparison of ADCH Variants on MIRFLICKE-25K and NUS-WIDE datasets.

Task	Method	MIRFLICKE-25K			NUS-WIDE		
		16 bits	24 bits	32 bits	16 bits	24 bits	32 bits
I → T	ADCH-b	0.8710	0.8965	0.8885	0.8103	0.8131	0.8193
	ADCH	**0.8878**	**0.9011**	**0.9041**	**0.8197**	**0.8192**	**0.8196**
T → I	ADCH-b	0.8610	0.8765	0.8785	0.7903	0.7931	0.8003
	ADCH	**0.8796**	**0.8864**	**0.8864**	**0.8034**	**0.8022**	**0.8036**

4.3 Accuracy

This experiment is to investigate the performance of the ADCH method on cross-modal retrieval on given datasets. The MAP results are presented in Table 1. We compare our ADCH with eight cross-modal methods with the output dimensions of 16 bits, 24 bits and 32 bits. We can find that ADCH significantly outperforms all the other baselines, including deep hashing baselines. DCMH is a well-known method, having a good performance in cross-modal retrieval. From the experimental results, the MAP of ADCH is higher than that of DCMH by approximately fifteen percent in MIRFLICKR-25K and twenty-five percent in NUS-WIDE. I → T denotes that the query is image and the database is text, and T → I denotes that the query is text and the database is image. † denotes the result cited from [7]. The best results for MAP are shown in bold.

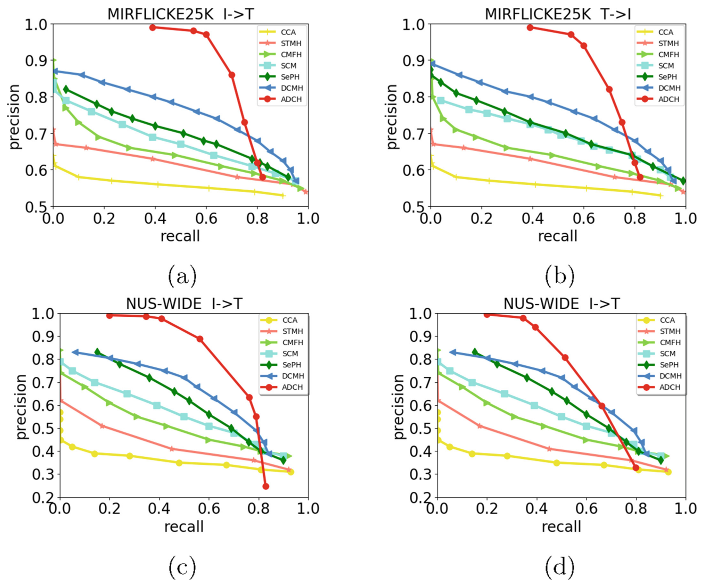

Fig. 2. The Precision-Recall curves on MIRFLICKR-25K and NUS-WIDE with 16 Bits Hash Code.

To go deeper with the effectiveness of ADCH, we design a variant of the proposed approach: ADCH-b is the ADCH variant that utilizes the cross-modal similarity matrix **S** with value 0 or 1. In Table 2, by introducing adaptive similarity matrix **S** into loss the detailed similarity is quantified into percentages, ADCH outperforms ADCH-b. The results show that the detailed definition of similarity matrix S_{ij} can provide more information of semantic similarity.

We report precision-recall curve in Fig. 2. According to the increasing of recall, the precision of ADCH is higher than other baselines. Our ADCH can also achieve the best performance on other cases with different values of codes length, such as 32 bits and 64 bits. $Precision = \frac{true\ positive\ data}{retrieval}$ data, and $Recall = \frac{true\ positive\ data}{data\ in\ database}$. With the same recall, precision of ADCH is higher than other methods.

When the precision is high, there are a large number of true positive data retrieved, and a small number of true positive data left in the database. To increase recall, we need more true positive data, so more retrieval data should be obtained. The number of retrieval data increase dramatically, but the true positive data show a slow growth. So the ADCH precision drops quickly.

4.4 Time Complexity

We compare ADCH with DCMH on the dataset MIRFLICKE-25K in Fig. 3. Y-axis is MAP of $I \rightarrow T$, and X-axis is training time. The length of hash codes is 16 bits, 24 bits, and 32 bits. We can see ADCH outperforms DCMH. With the increase of the length of hash codes, the training time increases gradually. The training time of DCMH is approximately six times than that of ADCH, and the ADCH training time is approximately one minute in dataset MIRFLICKE-25K. Hence, ADCH uses less training time.

The reasons of the training time of ADCH is much less than that of DCMH come into two aspects: Firstly, the training of deep neural networks is typically time-consuming, because they have to scan number of n database points. The computational cost of the traditional symmetric deep supervised hashing method is at least $O(n^2)$, if all database points are used for training. T_{out}, T_{in} and c are

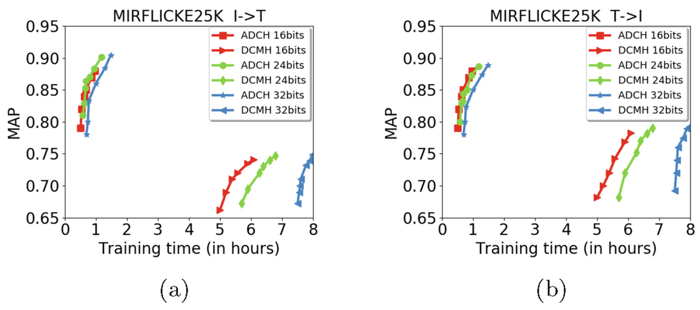

Fig. 3. Training time on MIRFLICKE-25K dataset with output dimensions 16 bits, 24 bits and 32 bits.

slightly relative to the number of database points n. Hence, the computational cost of ADCH is $O(n)$. Secondly, ADCH uses not only a direct hash code learning approach but also an asymmetric approach with the query and database points. The training time of ADCH is only the training time for the query points. Thus, the training time of ADCH is much less than that of DCMH.

In this paper, we only compare ADCH with DCMH on training time. Because the training time of directly learned hash codes methods are efficient than the methods of generating hash codes from hash function [2]. Thus, we do not compare ADCH with CHN and SSAH on training time.

4.5 Sensitivity to Parameters

As shown in Fig. 4a, the number of query points of ADCH effects the retrieval accuracy. With an increase in the number of query points on MIRFLICKE25K, the retrieval accuracy improves. When m > 2000, the MAP of I $\rightarrow$ T becomes gradually stable.

Figure 4b and c show the effect of the hyper-parameters γ and η on the MAP I $\rightarrow$ T result, and the binary codes lengths include 16 bits, 24 bits and 32 bits. We can see that the change in the hyper-parameters γ and η is not sensitive when $100 < \gamma < 500$ or $100 < \eta < 500$. In our experiments, we choose $m = 2000$, $\eta = \gamma = 200$, to get a stable and high MAP.

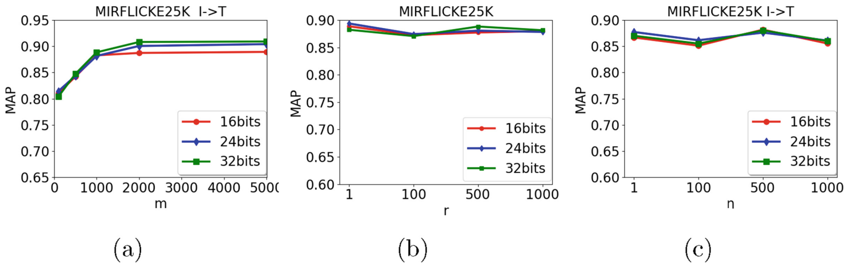

Fig. 4. The accuracy on MIRFLICKE25K of query number m, γ and η.

5 Conclusion

Effective retrieval of massive amounts of media data from heterogeneous sources is a great challenge. In this paper, we propose ADCH as a novel cross-modal deep supervised hashing method for cross-modal retrieval. ADCH uses two deep neural networks to extract image and text feature representations, and learns hash codes of query points and database points in an asymmetric way. Further, we propose a detailed definition of similarity matrix S_{ij} to improve the performance. Through experiments on real-word datasets, the results show that ADCH takes less time and achieves more accurate than traditional symmetric deep supervised hashing methods.

Acknowledgements. This work was supported by the Strategic Priority Research Program of the Chinese Academy of Sciences (XDC02050200).

References

1. Yiqun, H., Cheng, X., Chia, L.-T.: Coherent phrase model for efficient image near-duplicate retrieval. IEEE Trans. **11**, 1434–1445 (2007)
2. Jiang, Q.Y., Li, W.J.: Asymmetric deep supervised hashing. In: Proceedings of the 32nd AAAI Conference on Artificial Intelligence (2018)
3. Wang, W., Yang, X.: Effective deep learning-based multi-modal retrieval. VLDB J. **25**, 79–101 (2016)
4. Zhang, J., Peng, Y., Yuan, M.: Unsupervised Generative Adversarial Cross-modal Hashing. In: Proceedings of the 32nd AAAI Conference on Artificial Intelligence (2018)
5. Wang, Y., Wang, S., Huang, Q.: Online asymmetric similarity learning for cross-modal retrieval. In: 30th IEEE Conference on Computer Vision and Pattern Recognition (2017)
6. Cao, Y., Long, M., Wang, J.: Deep visual-semantic hashing for cross-modal retrieval. In: ACM SIGKDD Conference on Knowledge Discovery and Data Mining (2016)
7. Jiang, Q., Li, W.: Deep cross-modal hashing. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2017)
8. Cao, Y., Long, M., Wang, J., Liu, S.: Collective deep quantization for efficient cross-modal retrieval. In: AAAI Conference on Artificial Intelligence (2017)
9. Hardoon, D.R., Szedmak, S., Shawe-Taylor, J.: Canonical correlation analysis: an overview with application to learning methods. Neural Comput. **12**, 2639–2664 (2017)
10. Cao, Y., Long, M., Wang, J.: Correlation hashing network for efficient cross-modal retrieval. Neural Computation, BMVC (2017)
11. Ranjan, V., Rasiwasia, N., Jawahar, C.V.: Multi-label cross-modal retrieval. In: 2015 IEEE International Conference on Computer Vision (2015)
12. Ngiam, J., Khosla, A., Kim, M., Nam, J., Lee, H., Ng, A.Y.: Multimodal deep learning. In: ICML (2011)
13. Srivastava, N., Salakhutdinov, R.R.: Multimodal learning with deep boltzmann machines. In: Advances in Neural Information Processing Systems (2012)
14. Feng, F., Wang, X., Li, R.: Cross-modal retrieval with correspondence auto encoder. In: ACM International Conference on Multimedia (2014)
15. Feng, F., Wang, X., Li, R.: Effective multimodal retrieval based on stacked auto encoders. In: International Conference on Very Large Data Bases (2014)
16. Chatfield, K., Simonyan, K., Vedaldi, A., Zisserman, A.: Return of the devil in the details: delving deep into convolutional nets. BMVC (2014)
17. Huiskes, M.J., Lew, M.S.: The MIR flickr retrieval evaluation. In: MM. ACM (2008)
18. Chua, T.-S., Tang, J., Hong, R., Li, H., Luo, Z., Zheng, Y.: NUS-WIDE: a real-world web image database from national university of Singapore. In: CIVR (2009)
19. Lin, Z., Ding, G., Hu, M., Wang, J.: Semantics-preserving hashing for cross-view retrieval. In: CVPR (2015)
20. Wang, D., Gao, X., Wang, X., He, L.: Semantic topic multimodal hashing for cross-media retrieval. In: IJCAI (2015)
21. Zhang, D., Li, W-J.: Large-scale supervised multimodal hashing with semantic correlation maximization. In: AAAI (2014)
22. Ding, G., Guo, Y., Zhou, J.: Collective matrix factorization hashing for multimodal data. In: CVPR (2014)
23. Li, C., Deng, C., Li, N., Liu, W., Gao, X., Tao, D.: Self-supervised adversarial hashing networks for cross-modal retrieval. In: CVPR (2018)