Deep Unified Cross-Modality Hashing by Pairwise Data Alignment

Yimu Wang, **Bo Xue**, **Quan Cheng**, **Yuhui Chen** and **Lijun Zhang** *
National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China
{wangym, xueb, chengq, chenyuhui, zhanglj}@lamda.nju.edu.cn

Abstract

With the increasing amount of multimedia data, cross-modality hashing has made great progress as it achieves sub-linear search time and low memory space. However, due to the huge discrepancy between different modalities, most existing cross-modality hashing methods cannot learn unified hash codes and functions for modalities at the same time. The gap between separated hash codes and functions further leads to bad search performance. In this paper, to address the issues above, we propose a novel end-to-end Deep Unified Cross-Modality Hashing method named DUCMH, which is able to jointly learn unified hash codes and unified hash functions by alternate learning and data alignment. Specifically, to reduce the discrepancy between image and text modalities, DUCMH utilizes data alignment to learn an auxiliary image to text mapping under the supervision of image-text pairs. For text data, hash codes can be obtained by unified hash functions, while for image data, DUCMH first maps images to texts by the auxiliary mapping, and then uses the mapped texts to obtain hash codes. DUCMH utilizes alternate learning to update unified hash codes and functions. Extensive experiments on three representative image-text datasets demonstrate the superiority of our DUCMH over several state-of-the-art cross-modality hashing methods.

1 Introduction

With the increasing multimedia data, approximate nearest neighbor (ANN) search has been a fundamental problem in the information retrieval area. Among several ANN search methods, hashing [Wang *et al.*, 2018; Chen *et al.*, 2019; Wang *et al.*, 2020] has attracted extensive attention. It maps data points to binary codes with hash functions by preserving the similarity in the original space of data points. Due to binary codes, the storage cost can be dramatically reduced with sub-linear search complexity. In many applications, such as search engines, systems have to handle data of multi-modalities. It requires researchers to support cross-modality retrieval that can return relevant results of one modality when querying another modality, *e.g.*, retrieving images with text queries. Due to its low storage cost and search time, cross-modality hashing (CMH) area receives more and more attention recently.

Existing CMH methods can be roughly divided into shallow CMH methods [Zhai *et al.*, 2013; Wu *et al.*, 2015; Xie *et al.*, 2016; Ye and Peng, 2018] and deep CMH methods [Jiang and Li, 2017; Yan *et al.*, 2017; Li *et al.*, 2018; Shi *et al.*, 2019; Sun *et al.*, 2019; Xu *et al.*, 2019]. Most shallow methods capture the semantic relevance in a common Hamming space and learn hash functions that map hand-crafted features to hash codes, which means the feature extraction procedure is independent of the hash code learning procedure. That may block the way to achieve satisfying performance, as the hand-crafted features might not be optimal for the hash code learning procedure. Recently, deep learning [Zagoruyko and Komodakis, 2016] has shown its superiority of representation learning in various applications, such as image recognition [He *et al.*, 2016]. Deep CMH methods leverage the power of deep learning by integrating feature learning and hash code learning into a single framework. As a result, deep methods capture non-linear correlations among cross-modality instances and achieve better performance than shallow methods. The key to CMH is to bridge the modality gap, as it makes original data distributions and feature representations of modalities different. Thus, it is necessary to explore semantic relevance between different modalities in sufficient details to reduce that gap and further improve search performance. Recently, some supervised deep methods [Li *et al.*, 2018; Zhan *et al.*, 2020] exploit semantic labels or relevance information, thereby better cross-modality correlations.

However, previous work [Jiang and Li, 2017; Li *et al.*, 2018; Sun *et al.*, 2019] mainly aligns different modalities in the level of representation using pre-defined loss functions and derives independent hash functions and codes for different modalities. Representation (level) alignment may reduce the modality gap in some intermediate procedures of independent hash functions, but inputs, other layers and final outputs (hash codes) of deep hash functions are usually of different distributions. Actually, high-dimensional modality-specific original data contain abundant information that enables us to bridge the modality gap by the data (level) alignment. As we have paired data of different modalities, *e.g.*, image-text paired data, leveraging that data to learn mappings between

*Lijun Zhang is the corresponding author.

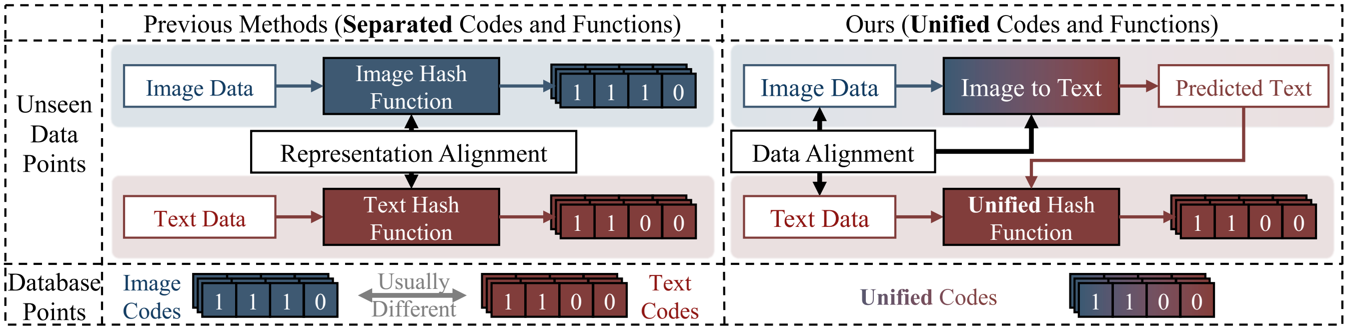

Figure 1: The comparison of previous cross-modality hashing methods and ours. Previous methods tend to align different modalities in the level of representations, while ours directly align modalities in the level of original data under the supervision of image-text pairs. Besides, most of the previous methods learn hash functions and hash codes for different modalities separately, while ours simultaneously learn unified hash functions and unified hash codes for bridging the gap between different modalities and further improve search performance.

modalities (data alignment) is possible and may bridge the modality gap better than representation alignment. Besides, with the mappings between modalities, we can derive unified hash functions and learn unified hash codes for correlated data for different modalities at the same time, while existing work cannot. The difference between data alignment (**unified** hash codes and functions) and representation alignment (**separated** hash codes and functions) is presented in Figure 1.

In this paper, to better bridge the modality gap and derive unified hash functions and hash codes for different modalities, we propose a novel Deep Unified Cross-Modality Hashing (DUCMH) method. Specifically, DUCMH alternatively learns unified hash functions and unified hash codes, while aligning different modalities by data alignment under the supervision of image-text pairs at hand. Our unified hash function maps texts to hash codes, while with an auxiliary image to text mapping trained under the supervision of paired data, the unified hash function is also able to map images to hash codes. The auxiliary image to text mapping maximizes the semantic relevance and distribution consistency between different modalities, while the unified hash function is employed for unifying hash codes of different modalities and further reducing modality gap. To empirically evaluate our DUCMH, we conduct extensive experiments on three commonly used image-text datasets, showing the superiority of our method over several state-of-the-art methods.

The main contributions of DUCMH are summarized below:

- To the best of our knowledge, DUCMH is the first deep method that derives unified hash codes for database instances and unified hashing functions for unseen query points at the same time.
- DUCMH is the first deep method to apply data alignment and learn the mapping between modalities. Data alignment can carefully preserve semantic relevance and distribution consistency across modalities, while effectively bridging the modality gap.
- Extensive experiments conducted on three image-text benchmarks show the superiority of our DUCMH over several state-of-the-art cross-modality hashing methods, including both shallow and deep methods.

2 Related Work

Existing cross-modality hashing work can be divided into unsupervised and supervised methods. For unsupervised methods, the intra- and inter-modality relations are exploited to generate hash codes without any supervised information [Hotelling, 1992]. On the other side, with more semantic information, supervised methods are able to capture correlations better and thus achieve superior performance [Ding *et al.*, 2014; Zhang and Li, 2014; Wang *et al.*, 2015; Luo *et al.*, 2018]. Semantic Correlation Maximization (SCM) [Zhang and Li, 2014] reduces the modality gap by utilizing label information to learn a modality-specific transformation, and preserves the maximal correlation between modalities. Collective Matrix Factorization Hashing (CMFH) [Ding *et al.*, 2014] employs the latent factor model to learn hash codes by collective matrix factorization from different modalities. Semantic Topic Multimodal Hashing (STMH) [Wang *et al.*, 2015] utilizes latent semantic information among different modalities to learn hash codes. Supervised Discrete Manifold-Embedded Cross-Modality Hashing (SDMCH) [Luo *et al.*, 2018] generates binary hash codes by exploiting the non-linear manifold structure of data and constructs the correlations among heterogeneous multiple modalities with semantic information.

The above methods utilize hand-crafted features to generate hash codes, which might not be optimally compatible with the hash-code learning procedure. With the advances of deep learning, more and more deep supervised CMH work has been proposed. Deep methods [Jiang and Li, 2017; Li *et al.*, 2018; Shi *et al.*, 2019; Zhan *et al.*, 2020] integrate feature learning and hash-code learning into a unified architecture with promising performance. Deep Cross-modality Hashing (DCMH) [Jiang and Li, 2017] is the first deep framework that performs feature learning and hash-code learning simultaneously. Self-Supervised Adversarial Hashing (SSAH) [Li *et al.*, 2018] incorporates a self-supervised semantic network coupled with multi-label information, and carries out adversarial learning to maximize the semantic relevance and feature distribution consistency between different modalities. Equally-Guided Discriminative Hashing (EGDH) [Shi *et al.*, 2019] takes semantic structure and discriminability into consideration to learn better hash functions and hash codes. Supervised

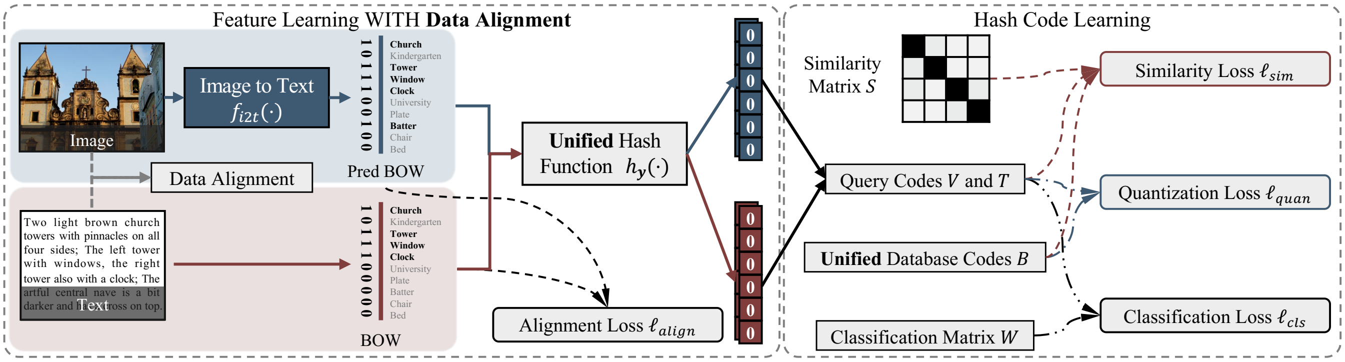
Feature Learning WITH **Data Alignment**
Image
Image to Text $f_{i2t}(\cdot)$
1011100100
Church
Kindergarten
Tower
Window
Clock
University
Plate
Batter
Chair
Bed
Pred BOW
Data Alignment
Two light brown church towers with pinnacles on all four sides; The left tower with windows, the right tower also with a clock; The artful central nave is a bit darker and has a cross on top.
Text
1011100000
Church
Kindergarten
Tower
Window
Clock
University
Plate
Batter
Chair
Bed
BOW
Unified Hash Function $h_y(\cdot)$
Alignment Loss ℓ_{align}
0
0
0
0
0
0
Hash Code Learning
Similarity Matrix S
Similarity Loss ℓ_{sim}
Query Codes V and T
Quantization Loss ℓ_{quan}
Unified Database Codes B
Classification Matrix W
Classification Loss ℓ_{cls}

Figure 2: The overall framework of our proposed DUCMH on Image and Text (Tags) modalities. DUCMH is the first deep method that employs data alignment with the image-text paired data to achieve unified hash functions and codes simultaneously. Data alignment reduces the modality gap better than the representation alignment previous work uses. Feature learning part includes an auxiliary image to text mapping $f_{i2t}(\cdot)$ and a unified hash function $h_{\boldsymbol{y}}(\cdot)$ mapping texts to hash codes. Hash code learning part employs different losses to learn unified hash codes and further improve search performance.

Hierarchical Deep Hashing (SHDCH) [Zhan *et al.*, 2020] explores the power of hierarchical labels to further boost the learning procedure.

While previous methods achieve success in performance by representation alignment, they ignore image-text pairs at hand, and never leverage that information to reduce the modality gap by data alignment or derive unified hash functions and codes simultaneously. To the best of our knowledge, DUCMH is the first deep method to employ data alignment by fully utilizing the image-text paired data for better reducing the modality gap and deriving unified hash functions and codes simultaneously.

3 DUCMH

In this section, we present the details about our deep unified cross-modality hashing (DUCMH) method, including model formulation and learning algorithm. DUCMH, shown in Figure 2, is an end-to-end learning framework consisting two parts: the feature learning part and the hash-code learning part. We employ asymmetric learning to alternatively update each part and data alignment to bridge the modality gap.

3.1 Problem and Notation

Scalars are defined by lowercase letters, such as w. Vectors and matrices are denoted by boldface lowercase letters and uppercase letters, *e.g.*, $\boldsymbol{w}$ and W. The i-th element of $\boldsymbol{w}$ and i-th row, j-column element of W are represented by subscripts as $\boldsymbol{w}_i$ and W_{ij}. Besides, W_{*j} and W_{i*} represent the j-th column and i-th row of matrix W, while $\tilde{W}_{*j}$ and $\tilde{W}_{i*}$ represent the matrices of matrix W without j-th column and i-th row.

Without loss of generality, we focus on CMH using image-text paired data. Assume that we have n training data points, in which each instance has two modalities of features, *i.e.*, image and text. $O = \{\boldsymbol{o}_i\}_{i=1}^n$ is a cross-modality dataset with n instance, while for i-th instance $\boldsymbol{o}_i = (\boldsymbol{x}_i, \boldsymbol{y}_i, \boldsymbol{l}_i)$, $\boldsymbol{x}_i$, $\boldsymbol{y}_i$, and $\boldsymbol{l}_i = [\boldsymbol{l}_{i,1}, \ldots, \boldsymbol{l}_{i,clsnum}]$ are the corresponding image, text and label, and $clsnum$ is the number of classes. If $\boldsymbol{o}_i$ belongs to the j-th class, $\boldsymbol{l}_{i,j} = 1$, otherwise $\boldsymbol{l}_{i,j} = 0$. Besides, we denote $L = [\boldsymbol{l}_1, \ldots, \boldsymbol{l}_n] \in \{0,1\}^{n \times clsnum}$. The similarity matrix $S \in \{-1,+1\}^{n \times n}$ is generated by labels. We let $S_{ij} = 1$ if $\boldsymbol{o}_i$ and $\boldsymbol{o}_j$ are semantically similar, in another word, share at least one label, otherwise $S_{ij} = -1$.

Given the above training information O and S, the goal of cross-modality hashing is to learn two hash functions $h_{\boldsymbol{x}}(\cdot) \in \{-1,+1\}^c$ for the image modality and $h_{\boldsymbol{y}}(\cdot) \in \{-1,+1\}^c$ for the text modality, where c is code length. Meanwhile, the similarity-preserving hash codes $B = [\boldsymbol{b}_1^\top, \ldots, \boldsymbol{b}_n^\top]^\top \in \{-1,+1\}^{n \times c}$ should also be derived. Two hash functions are used for generating hash codes of the unseen data points while hash codes should preserve the cross-modality similarity in S. Specifically, when $S_{ij} = 1$, the Hamming distance between the binary codes $\boldsymbol{b}_i$ and $\boldsymbol{b}_j$ should be small. Otherwise, the distance should be large.

3.2 Our Model – Learning by Paired Data

Feature Learning – Unified Hash Functions

As aforementioned, previous work [Li *et al.*, 2018; Shi *et al.*, 2019] usually employs *representation alignment*, which only reduces the modality gap in some intermediate procedure of algorithms. On the other side, as the image-text pairs of database are at hand, it is possible to employ *data alignment* to directly bridge the modality gap by learning a mapping from image to text or from text to image.

DUCMH chooses to learn the mapping $f_{i2t}(\cdot)$ from image to text. There are two reasons: first, it is easier to learn the mapping from image to text than another, as the image always contains more information, such as spatial information and styles, which are hard to be pictured by a sentence. Second, images might be influenced a lot by the lights, angles, and other conditions, while texts are more steady and the light, angle, spatial information of an image can be easily represented by a simple word. That mapping $f_{i2t}(\cdot)$ can be learned under the supervision of image-text pairs.

With the mapping $f_{i2t}(\cdot)$, we can represent an image $\boldsymbol{x}$ as a text $f_{i2t}(\boldsymbol{x})$. By employing an alignment loss presented in the following section, the modality gap between image and text can be better reduced than the representation alignment previous work employs.

For the real text $\boldsymbol{y}$ and predicted text $f_{i2t}(\boldsymbol{x})$, we use a unified hash function $h_{\boldsymbol{y}}(\cdot)$ mapping them to hash codes as,

$$\begin{aligned}\boldsymbol{b}^{\boldsymbol{x}} =& \text{sign}(h_{\boldsymbol{x}}(\boldsymbol{x})) = \text{sign}(h_{\boldsymbol{y}}(f_{i2t}(\boldsymbol{x})))\,,\\ \boldsymbol{b}^{\boldsymbol{y}} =& \text{sign}(h_{\boldsymbol{y}}(\boldsymbol{y}))\,.\end{aligned}$$

To generate discrete binary hash codes, the sign function $\text{sign}(\cdot)$ is attached after the last layer of our unified hash function $h_{\boldsymbol{y}}(\cdot)$. However, as the gradients of $\text{sign}(\cdot)$ are 0 at most of points, we use $\tanh(\cdot)$ instead to better train the unified hash function $h_{\boldsymbol{y}}(\cdot)$ and the image to text mapping $f_{i2t}(\cdot)$.

Hash Code Learning – Unified Hash Codes

In this section, we present our objective function. As unified hash codes B should preserve semantic information and also be capable of classification, we design three corresponding losses shown below. Besides, to bridge the modality gap, we also employ an alignment loss to align different modalities for better search performance.

First, DUCMH leverages image-text pairs $\{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{n}$ to learn a precise image to text mapping $f_{i2t}(\cdot)$ for data alignment. Thus, to fully bridge the modality gap on the database points, we use alignment loss to model the modality gap, which is shown below,

$$\ell_{align} = \epsilon \sum_{i=1}^{n} \|f_{i2t}(\boldsymbol{x}_i) - \boldsymbol{y}_i\|_2\,, \tag{1}$$

where ϵ is a hyper-parameter.

Second, as the learned database hash codes and the hash codes generated by the hashing function should preserve semantic similarity, we employ the similarity loss [Zhang and Li, 2014]. Similarity loss requires that for any two hash codes, if they are semantically similar, the Hamming distance between them should be 0. Otherwise the Hamming distance should be c. Therefore, this loss we minimize is as:

$$\begin{aligned}\ell_{sim} =& \sum_{i=1}^{n}\sum_{j=1}^{n} \left\|h_x(\boldsymbol{x}_i)\boldsymbol{b}_j^{\top} - cS_{ij}\right\|^2\\ &+ \sum_{i=1}^{n}\sum_{j=1}^{n} \left\|h_y(\boldsymbol{y}_i)\boldsymbol{b}_j^{\top} - cS_{ij}\right\|^2\\ &+ \alpha\sum_{i=1}^{n}\sum_{j=1}^{n} \left\|h_x(\boldsymbol{x}_i)h_y(\boldsymbol{y}_j)^{\top} - cS_{ij}\right\|^2\,,\end{aligned} \tag{2}$$

where α is a hyper-parameter, $\boldsymbol{b}_i$ and $\boldsymbol{b}_j$ are the hash codes of i-th and j-th data point learned by the alternative learning algorithm shown in the following section.

Third, as the hash space is small, *i.e.*, 2^c points, we expect hash codes are able to preserve discriminative semantic information and recover labels with simple transformations. We use $h_*(\cdot)W$ and $\boldsymbol{b}W$ to represent the predicted label recovered from hash codes, where $W \in \mathbb{R}^{c\times clsnum}$. To simplify the formulation of this loss, we employ Mean Squared Error to measure the label information contained in hash codes. Now, the classification loss is defined as follows:

$$\begin{aligned}\ell_{cls} =& \alpha\sum_{i=1}^{n}\left(\|h_x(\boldsymbol{x}_i)W - \boldsymbol{l}_i\|^2 + \|h_y(\boldsymbol{y}_i)W - \boldsymbol{l}_i\|^2\right)\\ &+ \alpha\sum_{i=1}^{n}\|\boldsymbol{b}_iW - \boldsymbol{l}_i\|^2 + \|W\|_F^2\,,\end{aligned} \tag{3}$$

where α is a hyper-parameter defined above.

Last but not the least, as we replace $\text{sign}(\cdot)$ with $\tanh(\cdot)$ after the last layer of our unified hash function $h_{\boldsymbol{y}}(\cdot)$ for better performance, hash functions is still expected to generate discrete hash codes. Intuitively, the quantization loss [Jiang and Li, 2017] is as:

$$\ell_{quan} = \rho\sum_{i=1}^{n}\left(\|h_x(\boldsymbol{x}_i) - \boldsymbol{b}_i\|^2 + \|h_y(\boldsymbol{y}_i) - \boldsymbol{b}_i\|^2\right)\,, \tag{4}$$

where ρ is a hyper-parameter.

Now, by minimizing four losses, our objective function can be formulated as:

$$\min_{\Theta,B,W} \ell = \ell_{align} + \ell_{sim} + \ell_{cls} + \ell_{quan}\,, \tag{5}$$

where Θ is the parameter of the unified hash function $h_{\boldsymbol{y}}(\cdot)$ and the image to text mapping $f_{i2t}(\cdot)$. Hyper-parameters ϵ, α and ρ are empirically set to $5000, 50$ and 200 for scaling the order of each loss.

3.3 Optimization

In this part, we present an alternative learning algorithm to learn Θ, B and W in Eq. (5), which means we update one parameter with others fixed. This algorithm is also summarized in Algorithm 1, while details are presented below.

Normally, we are only given database points O and the pairwise supervised information S between them. To accelerate the training procedure, we can learn hash-codes and hash functions by sampling a subset of O as the query set O_{train} for training, *i.e.*, $O_{train} \subseteq O$ in each step. Thus, the training complexity can be reduced to $\mathcal{O}(nm)$ from $\mathcal{O}(n^2)$, where $\mathcal{O}$ hides variables irrelevant to the size of dataset, m and n represent the size of datasets O and O_{train}.

Learn Θ with B and W fixed

When B and W are fixed, we learn and update the parameter Θ by back-propagation algorithm.

Learn B with Θ and W fixed

To simplify the calculation, when fixing Θ and W, we rewrite Eq. (5) in the matrix form as:

$$\begin{aligned}\min_{B} \ell =& \|VB^{\top} - cS\|_F^2 + \|TB^{\top} - cS\|_F^2\\ &+ \rho\left(\|V - B\|_F^2 + \|T - B\|_F^2\right)\\ &+ \alpha\|BW - L\|_F^2 + \text{const}\,,\end{aligned}$$

where const is the constant independent of B, $V = [h_{\boldsymbol{x}}(\boldsymbol{x}_1)^{\top}, \ldots, h_{\boldsymbol{x}}(\boldsymbol{x}_n)^{\top}]^{\top} \in \mathbb{R}^{n\times c}$, $T = [h_{\boldsymbol{y}}(\boldsymbol{y}_1)^{\top}, \ldots, h_{\boldsymbol{y}}(\boldsymbol{y}_n)^{\top}]^{\top} \in \mathbb{R}^{n\times c}$.

With little algebra, we can derive,

$$\begin{aligned}\min_{B} \ell =& \|VB^{\top}\|_F^2 + \|TB^{\top}\|_F^2 + \alpha\|BW\|_F^2\\ &- 2c\text{tr}\left(BV^{\top}S\right) - 2c\text{tr}\left(BT^{\top}S\right)\\ &- 2\alpha\text{tr}\left(BWL^{\top}\right) - 2\rho\text{tr}\left(B(T+V)^{\top}\right) + \text{const}\\ =& \text{tr}(B(Q+R)) + \text{const}\,,\end{aligned}$$

where $Q = V^{\top}BV^{\top} + T^{\top}BT^{\top} + \alpha WW^{\top}B^{\top}$ and $R = -2cV^{\top}S - 2cT^{\top}S - 2\alpha WL^{\top} - 2\rho(T+V)^{\top}$, and $\text{tr}(\cdot)$ is the trace of a matrix.

Algorithm 1 The alternative learning algorithm

```
Input: $O = \{\boldsymbol{o}_i\}_{i=1}^{n}$ : $n$ data points. $S \in \{-1,+1\}^{n\times n}$ is the similarity matrix. $c$ is the target binary code length. Mini-batch size $batchsize$, sample size $m$ and iteration number $T_{out}, T_{in}$.
Output: $\Theta$ is the parameter of the networks $h_{\boldsymbol{y}}(\cdot)$ and $f_{i2t}(\cdot)$. $B$ is the binary hash codes for data points.
Initialization: initialize $\Theta$, $B$, and $W$.
for $t_o = 1 \to T_{out}$ do
    Generate training set $O_{trian} \subseteq O$ and $S_{trian}$ by randomly indexing.
    for $t_i = 1 \to T_{in}$ do
        for $batch = 1 \to m/batchsize$ do
            Construct the mini-batch $O_{trian}^{batch}$ by randomly sample $batchsize$ points from $O_{trian}$.
            Calculate the binary hash codes of mini-batch $O_{trian}^{batch}$ by inferencing.
            Calculate the gradient by the chain rule and update $\Theta \leftarrow \Theta - \eta\frac{\partial \ell}{\partial \Theta}$ by back propagation.
        end for
    end for
    for $k = 1 \to c$ do
        Update $B_{*k}$ as Eq. (7).
    end for
    Update $W$ as Eq. (8).
end for
return $\Theta$ and $B$.
```

As this problem is NP-hard, to ease this problem, inspired by SDH [Shen *et al.*, 2015], we update one bit a time using the discrete cyclic coordinate descent method. We alternatively update a column of B with other columns fixed. Hence, this bit by bit problem at bit k becomes:

$$\min_{B_{*k}} \ell = \text{tr}(B_{*k}(Q_{k*} + R_{k*})) + \text{const}\,, \tag{6}$$

where $Q_{k*} = V_{*k}^{\top}\tilde{B}_{*k}\tilde{V}_{*k}^{\top} + T_{*k}^{\top}\tilde{B}_{*k}\tilde{T}_{*k}^{\top} + \alpha W_{k*}\tilde{W}_{k*}^{\top}\tilde{B}_{*k}^{\top}$, $R_{k*} = -2cV_{*k}^{\top}S - 2cT_{*k}^{\top}S - 2\alpha W_{k*}L^{\top} - 2\rho(T_{*k} + V_{*k})^{\top}$. Next, the optimal solution of problem (6) is as follows:

$$B_{*k} = -\text{sign}(Q_{k*} + R_{k*})^{\top}\,. \tag{7}$$

Learn W with Θ and B fixed

When fixing Θ and B, we rewrite Eq. (5) as:

$$\begin{aligned}\min_{W} \ell =& \alpha\left(\|VW - L\|_F^2 + \|TW - L\|_F^2\right)\\ &+ \alpha\|BW - L\|_F^2 + \|W\|_F^2 + \text{const}\,,\end{aligned}$$

where const is the constant independent of W. It is easy to solve W by the regularized least squares problem, while the closed-form solution is as:

$$W = P^{-1}(\alpha V + \alpha T + \alpha B)^{\top} L\,, \tag{8}$$

where $P = (\alpha V^{\top}V + \alpha T^{\top}T + \alpha B^{\top}B + \mathcal{I})$ and $\mathcal{I}$ is the diagonal with elements being 1.

4 Experiment

In this section, we carry out experiments to empirically evaluate the performance of DUCMH on three image-text benchmarks, and then compare it to state-of-the-art approaches.

4.1 Datasets and Evaluation Protocols

Three datasets, MIRFLICKR-25K [Huiskes and Lew, 2008], IAPR TC-12 [Escalante *et al.*, 2010], and NUS-WIDE [Chua *et al.*, 2009] are used for evaluation. **MIRFLICKR-25K** consists of 25,000 images collected from Flickr website. Each image is associated with several textual tags and annotated by one of the 24 unique labels. We select the points which have at least 20 textual tags for experiments. The text is represented by a 1386-dimensional bag-of-words (BOW) vector. **IAPR TC-12** consists of 20,000 image-text (images-sentences) pairs which are annotated using 255 labels. The text for each point is represented as a 2912-dimensional BOW vector from sentences following DCMH [Jiang and Li, 2017]. **NUS-WIDE** contains 260,648 web images 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 BOW vector. For MIRFLICKR-25K and IAPR TC-12, 2,000 data points are randomly sampled as the test (query) set, while for NUS-WIDE, 2,100 data points are selected. The remaining points as the retrieval set (database).

The retrieval performance is evaluated by one of the most widely used criteria, Mean Average Precision (mAP), which is the average of average precision for all the queries. All the data are reported with average values running five times.

4.2 Implementation Details and Comparison Methods

Our DUCMH method is implemented based on PyTorch [Paszke *et al.*, 2019] with eight NVIDIA V100 GPUs and optimized by the mini-batch SGD with the size of 64 and weight decay. The learning rate is initialized as 0.0001 for the image to text mapping $f_{i2t}(\cdot)$ and 0.004 for the unified hash function $h_{\boldsymbol{y}}(\cdot)$. For two neural network architectures, we use the following model: **Image to text mapping** $f_{i2t}(\cdot)$: It is based on CNN-F neural networks [Chatfield *et al.*, 2014]. We reserve the first seven layers, which are the same as those in CNN-F. Following this, a middle full-connected layer with 512 nodes and final full-connected layer which has nodes equal to the number of tags are framed. **The unified hash function** $h_{\boldsymbol{y}}(\cdot)$: It is built by a two convolutional layers followed by the $\tanh(\cdot)$ activation. The first convolutional layer, followed by the ReLu activation and dropout with 0.5, has the input channel, output channel, kernel size and stride of 1, 10240, (the number of tags, 1) and $(1, 1)$. The second convolutional layer has the input channel, output channel, kernel size and stride of 10240, c, $(1, 1)$ and $(1, 1)$.

We compare our method with several state-of-the-art (SOTA) hashing methods, including shallow methods, *i.e.*, CCA [Hotelling, 1992], CMFH [Ding *et al.*, 2014], SCM [Zhang and Li, 2014], STMH [Wang *et al.*, 2015], SDMCH [Luo *et al.*, 2018], and deep supervised methods, *i.e.*, DCMH [Jiang and Li, 2017], SSAH [Li *et al.*, 2018], EGDH [Shi *et al.*, 2019], SHDCH [Zhan *et al.*, 2020]. For our method and other deep hashing methods, the raw image is resized to 224×224 pixels as inputs, and the text inputs are BoW vectors. For traditional shallow methods, we extract 4096-dimensional deep features by the CNN-F model

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
T → I	CCA	0.5714	0.5733	0.5700	0.3489	0.3481	0.3491	0.3614	0.3498	0.3386
	CMFH	0.6365	0.6399	0.6429	0.4168	0.4212	0.4277	0.5031	0.5187	0.5225
	SCM	0.6939	0.7012	0.7060	0.3453	0.3410	0.3470	0.5344	0.5412	0.5484
	STMH	0.6074	0.6153	0.6217	0.3687	0.3897	0.4044	0.4471	0.4677	0.4780
	SDMCH	0.7692	0.7832	0.8102	0.5501	0.5660	0.5829	0.7307	0.7462	0.7659
	DCMH	0.7827	0.7900	0.7932	0.5185	0.5378	0.5468	0.6389	0.6511	0.6571
	SSAH	0.7818	0.7916	0.8006	0.5427	0.5515	0.5845	0.6384	0.6492	0.6489
	EGDH	0.7351	0.7598	0.7816	0.5256	0.5405	0.5599	0.6317	0.6462	0.6494
	SHDCH	0.7744	0.8001	0.8130	0.5716	0.6011	0.6109	0.6248	0.6793	0.6902
	DUCMH	**0.8379**	**0.8424**	**0.8455**	**0.6007**	**0.6579**	**0.6880**	**0.7507**	**0.7715**	**0.7880**
I → T	CCA	0.5718	0.5690	0.5661	0.3422	0.3367	0.3391	0.3578	0.3681	0.3587
	CMFH	0.6377	0.6418	0.6451	0.4189	0.4234	0.4251	0.4900	0.5053	0.5097
	SCM	0.6851	0.6921	0.7003	0.3692	0.3666	0.3802	0.5409	0.5485	0.5553
	STMH	0.6132	0.6219	0.6274	0.3775	0.4002	0.4130	0.4710	0.4684	0.4942
	SDMCH	0.6883	0.7089	0.7210	0.4839	0.4828	0.4951	0.6296	0.6235	0.6393
	DCMH	0.7410	0.7465	0.7485	0.4526	0.4732	0.4844	0.5903	0.6031	0.6093
	SSAH	0.7789	0.7912	0.7990	0.5393	0.5682	0.5812	0.6401	0.6671	0.6700
	EGDH	0.7695	0.7823	0.7854	0.5294	0.5321	0.5639	0.6159	0.6389	0.6331
	SHDCH	0.7788	0.7885	0.7881	0.5731	0.5703	0.5907	0.6604	0.6631	0.6703
	DUCMH	**0.8684**	**0.8756**	**0.8760**	**0.6392**	**0.6908**	**0.7104**	**0.6957**	**0.7041**	**0.7061**

Table 1: Comparison of mAP w.r.t. different number of bits on three datasets, *MIRFLICKR-25K*, *IAPR TC-12* and *NUS-WIDE*. Best in bold.

Task	Method	16 bits	32 bits	64 bits
T → I	DUCMH with tags	0.6007	0.6579	0.6880
	DUCMH with stcs	0.5824	0.5909	0.6140
I → T	DUCMH with tags	0.6392	0.6908	0.7104
	DUCMH with stcs	0.6106	0.6479	0.6659

Table 2: Comparison of mAP w.r.t. different number of bits on *IAPR TC-12*. "stcs" represents sentences.

pre-trained on ImageNet to conduct fair comparisons. Besides, for all the SOTA methods, we employ the hyper-parameters introduced in their papers.

4.3 Results on Image and Text (Tags)

Following the experiment settings of previous work [Ding *et al.*, 2014; Jiang and Li, 2017; Li *et al.*, 2018; Shi *et al.*, 2019], we first test the power of data alignment between text (tags) and image modalities.

The mAP searching results are presented in Table 1. In all datasets and two tasks, *i.e.*, text to image (T → I) and image to text (I → T), our proposed DUCMH outperforms all the methods with the power of data alignment and unified hash functions and codes. Specifically, in MIRFLICKR-25K, DUCMH outperforms SOTAs by 0.0461, 0.0423, 0.0325 of 16, 32, and 64 bits on the task of T → I, respectively, while on the task of I → T, our DUCMH outperforms SOTAs by 0.0895, 0.0844, 0.0770 of 16, 32, and 64 bits. Similar results can be found in other datasets on two tasks.

4.4 Results on Image and Text (Sentences)

As the original data of text modality in IAPR TC-12 are sentences, we conduct experiments on IAPR TC-12 to test the power of data alignment between text (sentences) and image modalities. We replace the image to text mapping $f_{i2t}(\cdot)$ with a classical image caption method, *i.e.*, show and tell [Vinyals *et al.*, 2015], and the unified hash function $h_{\boldsymbol{y}}(\cdot)$ with a classical sentence classification method, *i.e.*, TextCNN [Kim, 2014] (represented by "DUCMH with stcs" in Table 2). We still use the hyper-parameters introduced in the original papers, while other hyper-parameters ϵ, α and ρ are still the same.

The results are shown in Table 2. As tags are pure semantic information without noises extracted from sentences and learning to predict conceptual tags is much easier than learning to caption images, it is reasonable that the model using tags ("DUCMH with tags" in Table 2) outperforms the model using sentences ("DUCMH with stcs" in Table 2).

5 Conclusion

In this paper, we presented a novel cross-modality hashing method named DUCMH. DUCMH was the first deep method that learned unified hash functions and hash codes simultaneously. The key contribution was leveraging the power of data alignment and image-text data pairs at hand to better reduce modality gap than the representation alignment previous work employed. Extensive experiments showed the superiority of our method over several state-of-the-art hashing methods. In the future, we would like to explore the potential of data alignment on the task of paired unsupervised and unpaired supervised cross-modality hashing.

Acknowledgments

This work was partially supported by the NSFC (61976112), JiangsuSF (BK20200064), and Open Research Projects of Zhejiang Lab (NO. 2021KB0AB02).

References

[Chatfield *et al.*, 2014] Ken Chatfield, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Return of the Devil in the Details: Delving Deep into Convolutional Nets. In *BMVC*, 2014.

[Chen *et al.*, 2019] Tian-Yi Chen, Lan Zhang, Shi-cong Zhang, Zi-long Li, and Bai-chuan Huang. Extensible Cross-Modal Hashing. In *IJCAI*, pages 2109–2115, 2019.

[Chua *et al.*, 2009] Tat-Seng Chua, Jinhui Tang, Richang Hong, Haojie Li, Zhiping Luo, and Yantao Zheng. NUS-WIDE: a real-world web image database from National University of Singapore. In *CIVR*, pages 1–9, 2009.

[Ding *et al.*, 2014] Guiguang Ding, Yuchen Guo, and Jile Zhou. Collective Matrix Factorization Hashing for Multimodal Data. In *CVPR*, pages 2083–2090, 2014.

[Escalante *et al.*, 2010] Hugo Jair Escalante, Carlos A. Hernández, Jesus A. Gonzalez, A. López-López, Manuel Montes, Eduardo F. Morales, L. Enrique Sucar, Luis Villaseñor, and Michael Grubinger. The segmented and annotated IAPR TC-12 benchmark. *CVIR*, 114(4):419 – 428, 2010.

[He *et al.*, 2016] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.

[Hotelling, 1992] Harold Hotelling. Relations between two sets of variates. In *Breakthroughs in statistics*, pages 162–190. 1992.

[Huiskes and Lew, 2008] Mark J. Huiskes and Michael S. Lew. The MIR Flickr Retrieval Evaluation. In *MIR*, pages 39–43, 2008.

[Jiang and Li, 2017] Qing-Yuan Jiang and Wu-Jun Li. Deep Cross-Modal Hashing. In *CVPR*, pages 3270–3278, 2017.

[Kim, 2014] Yoon Kim. Convolutional Neural Networks for Sentence Classification. In *EMNLP*, pages 1746–1751, 2014.

[Li *et al.*, 2018] Chao Li, Cheng Deng, Ning Li, Wei Liu, Xinbo Gao, and Dacheng Tao. Self-Supervised Adversarial Hashing Networks for Cross-Modal Retrieval. In *CVPR*, pages 4242–4251, 2018.

[Luo *et al.*, 2018] Xin Luo, Xiao-Ya Yin, Liqiang Nie, Xuemeng Song, Yongxin Wang, and Xin-Shun Xu. SDMCH: Supervised Discrete Manifold-Embedded Cross-Modal Hashing. In *IJCAI*, pages 2518–2524, 2018.

[Paszke *et al.*, 2019] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *NeurIPS*, pages 8024–8035. 2019.

[Shen *et al.*, 2015] Fumin Shen, Chunhua Shen, Wei Liu, and Heng Tao Shen. Supervised discrete hashing. In *CVPR*, pages 37–45, 2015.

[Shi *et al.*, 2019] Yufeng Shi, Xinge You, Feng Zheng, Shuo Wang, and Qinmu Peng. Equally-Guided Discriminative Hashing for Cross-modal Retrieval. In *IJCAI*, pages 4767–4773, 2019.

[Sun *et al.*, 2019] Changchang Sun, Xuemeng Song, Fuli Feng, Wayne Xin Zhao, Hao Zhang, and Liqiang Nie. Supervised Hierarchical Cross-Modal Hashing. In *SIGIR*, pages 725–734, 2019.

[Vinyals *et al.*, 2015] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan. Show and tell: A neural image caption generator. In *CVPR*, pages 3156–3164, 2015.

[Wang *et al.*, 2015] Di Wang, Xinbo Gao, Xiumei Wang, and Lihuo He. Semantic Topic Multimodal Hashing for Cross-Media Retrieval. In *IJCAI*, pages 3890–3896, 2015.

[Wang *et al.*, 2018] Jingdong Wang, Ting Zhang, Jingkuan Song, Nicu Sebe, and Heng Tao Shen. A Survey on Learning to Hash. *TPAMI*, 40(4):769–790, 2018.

[Wang *et al.*, 2020] Yimu Wang, Shiyin Lu, and Lijun Zhang. Searching privately by imperceptible lying: A novel private hashing method with differential privacy. In *ACM MM*, page 2700–2709, 2020.

[Wu *et al.*, 2015] Botong Wu, Qiang Yang, Wei-Shi Zheng, Yizhou Wang, and Jingdong Wang. Quantized Correlation Hashing for Fast Cross-Modal Search. In *IJCAI*, pages 3946–3952, 2015.

[Xie *et al.*, 2016] Liang Xie, Jialie Shen, and Lei Zhu. Online Cross-Modal Hashing for Web Image Retrieval. In *AAAI*, pages 294–300, 2016.

[Xu *et al.*, 2019] Ruiqing Xu, Chao Li, Junchi Yan, Cheng Deng, and Xianglong Liu. Graph Convolutional Network Hashing for Cross-Modal Retrieval. In *IJCAI*, pages 982–988, 2019.

[Yan *et al.*, 2017] Xinyu Yan, Lijun Zhang, and Wu-Jun Li. Semi-supervised deep hashing with a bipartite graph. In *IJCAI*, pages 3238–3244, 2017.

[Ye and Peng, 2018] Zhaoda Ye and Yuxin Peng. Multi-Scale Correlation for Sequential Cross-modal Hashing Learning. In *ACM MM*, pages 852–860, 2018.

[Zagoruyko and Komodakis, 2016] Sergey Zagoruyko and Nikos Komodakis. Wide Residual Networks. In *BMVC*, pages 87.1–87.12, 2016.

[Zhai *et al.*, 2013] Deming Zhai, Hong Chang, Yi Zhen, Xianming Liu, Xilin Chen, and Wen Gao. Parametric Local Multimodal Hashing for Cross-View Similarity Search. In *IJCAI*, pages 2754–2760, 2013.

[Zhan *et al.*, 2020] Yu-Wei Zhan, Xin Luo, Yongxin Wang, and Xin-Shun Xu. Supervised Hierarchical Deep Hashing for Cross-Modal Retrieval. In *ACM MM*, pages 3386–3394, 2020.

[Zhang and Li, 2014] Dongqing Zhang and Wu-Jun Li. Large-Scale Supervised Multimodal Hashing with Semantic Correlation Maximization. In *AAAI*, pages 2177–2183, 2014.