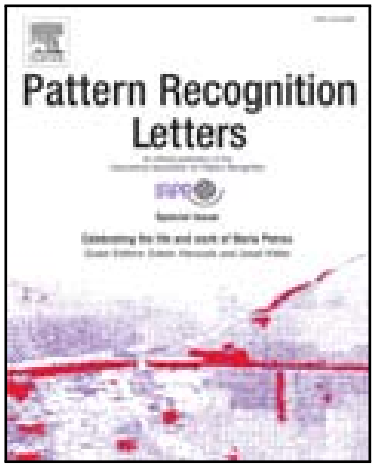
Accepted Manuscript



Deep Supervised Hashing using Symmetric Relative Entropy

Xueni Zhang, Lei Zhou, Xiao Bai, Xiushu Luan, Jie Luo,
Edwin R. Hancock

PII: S0167-8655(19)30201-6
DOI: https://doi.org/10.1016/j.patrec.2019.07.010
Reference: PATREC 7573

To appear in: *Pattern Recognition Letters*

Received date: 29 January 2019
Revised date: 3 July 2019
Accepted date: 10 July 2019

Please cite this article as: Xueni Zhang, Lei Zhou, Xiao Bai, Xiushu Luan, Jie Luo, Edwin R. Hancock, Deep Supervised Hashing using Symmetric Relative Entropy, *Pattern Recognition Letters* (2019), doi: https://doi.org/10.1016/j.patrec.2019.07.010

Highlights

- We propose a symmetric relative entropy based deep hashing method.
- Our method can decrease the information loss during hashing embedding.
- Distance similarity and distribution similarity can be simultaneously learned.
- We design a mutually optimization strategy for our deep hashing architecture.
- Extensive experiments show that our method achieves state-of-the-art performance.

Pattern Recognition Letters
journal homepage: www.elsevier.com

Deep Supervised Hashing using Symmetric Relative Entropy

Xueni Zhang[a,2], Lei Zhou[a,2], Xiao Bai[a], Xiushu Luan[b,**], Jie Luo[a], Edwin R. Hancock[c,a]

[a]*School of Computer Science and Engineering, Beijing Advanced Innovation Center for Big Data and Brain Computing, State Key Laboratory of Software Development Environment, Jiangxi Research Institute, Beihang University, Beijing, China*
[b]*Department of Anesthesiology, XuanWu Hospital, Captial Medical University, Beijing, China*
[c]*Department of Computer Science, University of York, York, UK*

ABSTRACT

By virtue of their simplicity and efficiency, hashing algorithms have achieved significant success on large-scale approximate nearest neighbor search. Recently, many deep neural network based hashing methods have been proposed to improve the search accuracy by simultaneously learning both the feature representation and the binary hash functions. Most deep hashing methods depend on supervised semantic label information for preserving the distance or similarity between local structures, which unfortunately ignores the global distribution of the learned hash codes. We propose a novel deep supervised hashing method that aims to minimize the information loss generated during the embedding process. Specifically, the information loss is measured by the Jensen-Shannon divergence to ensure that compact hash codes have a similar distribution with those from the original images. Experimental results show that our method outperforms current state-of-the-art approaches on two benchmark datasets.

1. Introduction

With the explosive growth in the number of high volume and high dimensional multimedia data sources, the potential for including images and videos in efficient large-scale visual retrieval has received increased interest. A general solution to image retrieval is approximate nearest neighbor (ANN) search, and which has been demonstrated to retrieve target images effectively and efficiently. Due to the rapid query speeds and low memory costs achievable, hashing has become one of the most widely used techniques among existing ANN methods. The key idea of hashing is to embed high-dimensional data into a set of compact binary codes while preserving the similarity of the original data in the Hamming space of the codes.

Existing hashing algorithms can be divided into a) data-independent and b) data-dependent groups. Data-independent methods usually use random projections as the hash functions to maximize the probability of "collision" between similar items. A representative data-independent method is locality sensitive hashing (LSH) (Gionis et al., 1999), which directly uses a random linear projection to learn hash codes. Although it benefits from the theoretical bounds on the approximation quality, it has been proved that the LSH method suffers in terms of poor accuracy of image retrieval.

Compared with data-independent methods, data-dependent methods on the other hand attempt to learn hash functions from training data and can achieve better performance using shorter hash codes. According to whether image labels are used or not, they can be further categorized into supervised and unsupervised methods. Unsupervised hashing methods learn hash functions by exploring the inherent structure of training data. Typical learning goals include reconstruction error minimization (Gong et al., 2013; Jegou et al., 2011) and preserving graph structure (Liu et al., 2011; Weiss et al., 2009). With the development and widespread adoption of deep learning techniques, neural networks have been used to learn the data distribution and often achieve better retrieval performance. Deep Hashing (DH) (Erin Liong et al., 2015) and several related extensions of the idea (Lin et al., 2016; Hu et al., 2017; Huang et al., 2017; Shen et al., 2018) utilize the network as the nonlinear hash function and design mapping criteria to obtain an improved hashing mapping. Most unsupervised hashing methods attempt to make

**Corresponding author
e-mail: luanxiushu@xwhosp.org (Xiushu Luan)
[2]The first two authors contribute equally to this work.

the learned hash code independent and balanced. They also attempt to construct pseudo labels using distance metrics to learn the hash function. However, all of these methods face the semantic gap dilemma. In other words, similar low-level features may have very different high-level semantic descriptions, so the results of retrieval cannot match the performance of visual perception for human observers.

Many supervised hashing methods were proposed (Liu et al., 2012; Zhang et al., 2014; Lin et al., 2014; Bai et al., 2014; Xiao et al., 2009; Yang et al., 2016; Bai et al., 2018; Su et al., 2019; Luo et al., 2018; Tang et al., 2015; Tang and Li, 2018; Li and Tang, 2015) to mitigate the semantic gap using the image labels. Recently, research on deep supervised hashing methods has shown that feature extraction and hash coding can be conducted more effectively by end-to-end learning of deep neural networks. These algorithms show the state-of-the-art results on many benchmarks (Li et al., 2018; Song et al., 2017; Wang et al., 2019; Yan et al., 2019). Most existing deep hashing methods utilize semantic labels to learn discriminative binary codes. Class-label based methods, such as DLBC (Lin et al., 2015) and DHCQ (Tang et al., 2018), expect to generate compact binary codes applicable to classification. Others pay attention to better modeling the distance between the original samples. Absolute distance is used in pairwise hashing methods, such as DQN (Cao et al., 2016), DHN (Zhu et al., 2016), DSH (Liu et al., 2016), DPSH (Li et al., 2015), DSDH (Li et al., 2017), which tries to make the Hamming distance between similar images as small as possible and vice versa. Triplet methods, such as NINH (Lai et al., 2015), DSRH (Zhao et al., 2015), DRSCH (Zhang et al., 2015), DTSH (Wang et al., 2016), consider the relative distance between images, and aim to keep the Hamming distance between dissimilar images larger than those from similar images.

Although deep learning based methods have achieved significant success on the problem of image retrieval, they mainly focus on preserving the pairwise distance relationships between images and hence ignore the global data distribution which is important for feature represention and hash coding. In our previous work (Zhang et al., 2018), we have presented a deep supervised hashing method that utilizes the Kullback-Leibler (KL) divergence to constrain the hashing codes to have a similar distribution with the original data. However, when retrieving images online using the hash codes, one may also want to minimize the information loss between the hash codes and the original images. Hence, in this paper, we propose an extended method to address this issue. It leverages a symmetric relative entropy, the Jensen-Shannon (JS) divergence, which is both symmetric and has a finite value, to minimize the distribution distance between the sources of the original image data and the hash codes. Furthermore, we analyze these two different relative entropies and demonstrate experimentally that the symmetric relative entropy based hashing method gives the best performance.

In brief, our contributions can be summarized as follows:

1. We propose a symmetric relative entropy based deep hashing method that reduces the information loss during hashing embedding by preserving similarity between the distribution of the original samples and the generated hash codes.
2. Both the distance-based similarity and the distribution-based similarity can be simultaneously learned and mutually optimized in our deep hashing architecture.
3. Extensive experiments on two image benchmarks show that our method can achieve comparable performance to state-of-the-art methods for image retrieval.

2. Related work

2.1. Hashing

A comprehensive survey on learning to hash has been given by (Wang et al., 2018). The relevant methods can be divided into a) unsupervised and b) supervised categories. Unsupervised hashing methods learn hash functions by exploring the intrinsic structure of unlabeled training data. Weiss et al. developed Spectral Hashing (SH) (Weiss et al., 2009) to formulate hash coding as a graph partition problem. The method generates binary codes by calculating the eigenvectors of the graph Laplacian. Liu et al. presented the Anchor Graph Hashing (AGH) (Liu et al., 2011) method to automatically capture the neighborhood structure inherent in a given massively large dataset. This work was extended to develop the Discrete Graph Hashing (DGH) (Liu et al., 2014) method by introducing a tractable alternating optimization method for preserving similarity in the discrete Hamming space of the hash codes. Apart from exploring the data structure based on a graph representation, reconstruction error minimization is another useful criterion to guide the learning of hash codes. The goal of Iterative Quantization (ITQ) (Gong et al., 2013) is to reduce the quantization loss from a real-valued feature vector to the vertices of a binary cube by finding a rotation of zero-centered data. The Binary Reconstructive Embedding (BRE) (Kulis and Darrell, 2009) method tries to learn hash functions by minimizing the reconstruction error directly using the coordinate-descent strategy. In a manner different from BRE that measures data similarity by Euclidean distance, the Angular Reconstructive Embedding (ARE) (Hu et al., 2018) method uses cosine similarity. Collective Reconstructive Embedding (CRE) (Hu et al., 2019), on the other hand, uses both cosine-based and Euclidean-based similarity simultaneously to address the cross-model hashing problem. In 2015, Liong et al. proposed Deep Hashing (DH) (Erin Liong et al., 2015) that first utilizes a multi-layer neural network as the hash function to preserve nonlinear neighborhood relationships. Due to the absence of label information, the performance of unsupervised hashing approaches is usually surpassed by supervised hashing.

Supervised hashing methods learn hash functions using both label information and a representation of the data. Two Step Hashing (TSH) (Lin et al., 2013) was developed to decompose the hashing learning problem into hash bit learning and hash function learning based on the learned bits, which is much easier to solve. Shen et al. introduced Supervised Discrete Hashing (SDH) (Shen et al., 2015) to optimize the discrete optimization problem directly with cyclic coordinate descent in conjunction with classification. With the aid of convolutional neural

networks and supervised information, deep supervised hashing achieves a great breakthrough in image retrieval. CNNH (Xia et al., 2014) is the first deep hashing algorithm which needs two stages to learn the high-level representation and binary codes. A drawback of this method is that the hash codes cannot be updated with the learned new image representation. To address this problem, Lai et al. propounded NINH (Lai et al., 2015) to simultaneously learn the feature representation and the hashing codes in a joint optimizing process. Then Zhao et al. described DRSH (Zhao et al., 2015) to use triplet ranking loss as multi-level similarity information to guide the learning of binary codes. As a representation of methods based on pairwise similarity, DPSH (Li et al., 2015) was developed to maximize the probability of similarity labels in the Hamming space using a Bayesian framework. Moreover, Deep Asymmetric Pairwise hashing (DAPH) (Shen et al., 2017a,b) provided the asymmetric hash forms to pairwise similarity loss for preserving more similarity information. DSH (Liu et al., 2016) designed a regularizer to encourage the binary-like outputs of neural networks to decrease the quantization loss. Later, DSDH (Li et al., 2017) was proposed by applying classification information as well as pairwise similarity to directly optimize this discrete problem. In prior work, we introduced an entropy-based deep hashing method (Zhang et al., 2018) by minimizing the information loss during the hashing process. Unlike the MFH method (Song et al., 2013) which attempts to reduce the quantization loss by minimizing the empirical error of the outputs of hash functions with respect to the learned hash codes, our method aims at preserving the original data distribution in the generated space of hash codes.

2.2. Information divergence

Information divergence is a measure of dissimilarity between probability distributions. One commonly used example is the KL divergence:

$$KL(p\|q) = \sum_{i=1}^{N} p(x_i) \log \frac{p(x_i)}{q(x_i)}$$

where p and q are two probability distributions. This is a non-negative function, which is equal to zero when $p(x) = q(x)$. Obviously, KL divergence quantifies the information loss from one distribution to another.

The JS divergence is another commonly used measure for the distance between distributions. It is defined as:

$$JS(p\|q) = \frac{1}{2} KL(p\|m) + \frac{1}{2} KL(q\|m)$$

where $m(x) = \frac{p(x)+q(x)}{2}$. This divergence can be interpreted as the average distance between each probability distribution and the average distribution, or equivalently as the diversity of two distributions with equal priors.

Compared with the KL divergence, the JS divergence has some advantages. Firstly, it is symmetric, i.e., JS divergence quantifies information loss from p to q as well as from q to p. Secondly, JS divergence has a definite upper bound. Hence, JS divergence is a better metric to measure the information loss during the hashing process.

3. Proposed Method

Suppose we have N training points $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^{N} \in \mathbb{R}^{d\times N}$ where each sample is represented as a d-dimensional feature vector $\mathbf{x}_i$. Besides the feature vectors, some pairs of points $\mathbf{x}_i$ and $\mathbf{x}_j$ are labeled with s_{ij} to indicate the pairwise similarity, where $s_{ij} = 1$ if $\mathbf{x}_i$ and $\mathbf{x}_j$ are similar and $s_{ij} = 0$ if $\mathbf{x}_i$ and $\mathbf{x}_j$ are dissimilar. These similarity labels can be either provided manually or acquired from typical semantic labels.

The goal of hashing is to learn a collection of K-bit binary codes $\mathbf{B} \in \{-1, 1\}^{K\times N}$, where the i-th column $\mathbf{b}_i \in \{-1, 1\}^K$ denotes the binary codes for the i-th sample $\mathbf{x}_i$. The binary codes are generated by the hash function $h(\cdot)$, which can be rewritten as $[h_1(\cdot), \ldots, h_c(\cdot)]$. For each image sample $\mathbf{x}_i$, its hash codes can be represented as $\mathbf{b}_i = h(\mathbf{x}_i) = [h_1(\cdot), \ldots, h_c(\cdot)]$. Generally speaking, hash coding aims to learn a hash function to project image samples to a set of binary codes.

3.1. Preserving Similarity

We commence by posing the problem of preserving pairwise similarity in a Bayesian framework. The goal is to locate the hash codes $\mathbf{B}$ that satisfy the constraints provided by the similarity labels S as closely as possible in the Hamming space.

Since there is a linear relationship between the Hamming distance and the corresponding inner product for each pair of binary codes $\mathbf{b}_i$ and $\mathbf{b}_j$, i.e., $dist_H(\mathbf{b}_i, \mathbf{b}_j) = \frac{1}{2}(K - \langle \mathbf{b}_i, \mathbf{b}_j \rangle)$, we can use the inner product to quantify pairwise similarity rather than Hamming distance. Given the binary codes $\mathbf{B} = \{\mathbf{b}_i\}_{i=1}^{n}$ for all the points, we can define the likelihood of the pairwise labels $S = \{s_{ij}\}$ as:

$$p(s_{ij} \mid \mathbf{B}) = \begin{cases} \sigma(\Omega_{ij}) & s_{ij} = 1 \\ 1 - \sigma(\Omega_{ij}) & s_{ij} = 0 \end{cases} \tag{1}$$

where $\sigma(\Omega_{ij}) = \frac{1}{1+e^{-\Omega_{ij}}}$, and $\Omega_{ij} = \frac{1}{2}\mathbf{b}_i^T \mathbf{b}_j$.

As a result, the larger the inner product $\langle \mathbf{b}_i, \mathbf{b}_j \rangle$, the smaller the corresponding $dist_H(\mathbf{b}_i, \mathbf{b}_j)$, and the larger $p(1 \mid \mathbf{b}_i, \mathbf{b}_j)$. This means that $\mathbf{b}_i$ and $\mathbf{b}_j$ should be classified as similar, and vice versa.

By taking the negative log-likelihood of the observed set of pairwise labels in S, we obtain the following optimization problem:

$$\min_{\mathbf{B}} J_1 = -\log p(S \mid \mathbf{B}) = -\sum_{s_{ij}\in S} (s_{ij}\Omega_{ij} - \log(1 + e^{\Omega_{ij}})) \tag{2}$$

This equation requires the Hamming distance of two similar points to be as small as possible, and simultaneously requires the Hamming distance between two dissimilar points to be as large as possible. This is exactly the goal of supervised hashing with pairwise similarity.

Although pairwise similarity supervision is a good way to preserve the distance similarity between the original images, the available label information is not fully exploited. Since the labels convey more information than similarity alone, as mentioned in (Lin et al., 2015), it is a reasonable assumption that good binary codes should contain enough semantic information to preserve the semantic similarity between images. In other

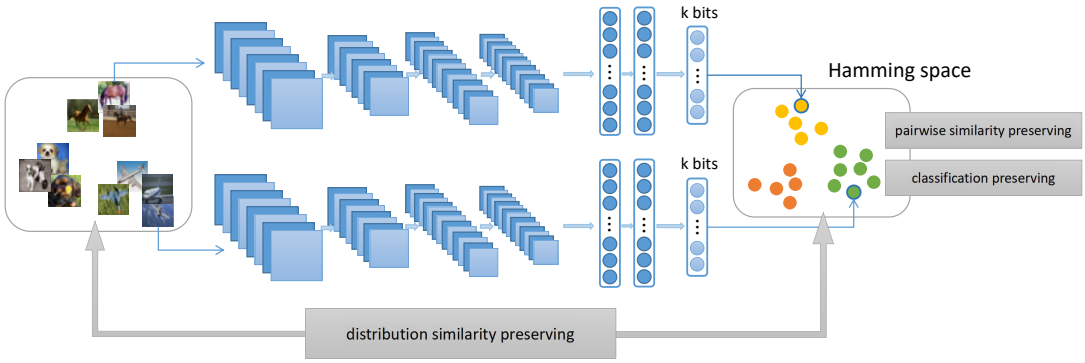


Fig. 1. The architecture of our proposed method.

words, the learned binary codes should be ideal for image classification.

Considering the binary code learning problem in the linear classification framework, the multi-class classification problem can be stated as follows:

$$\mathbf{y} = \mathbf{W}^T\mathbf{b} = [\mathbf{w}_1^T\mathbf{b}, \cdots, \mathbf{w}_C^T\mathbf{b}]^T \tag{3}$$

where $\mathbf{w}_k \in \mathbb{R}^L$, $k = 1, \cdots, C$ is the classification vector for class k and $\mathbf{y} \in \mathbb{R}^L$ is the label vector, of which the maximum item indicates the assigned class of $\mathbf{x}$. Thus, we can obtain the following optimization problem:

$$\min_{\mathbf{B},\mathbf{W}} J_2 = \sum_{i=1}^{n} L(\mathbf{y}_i, \mathbf{W}^T\mathbf{b}_i) + \lambda\|\mathbf{W}\|^2 \tag{4}$$

where λ is the regularization parameter, $\mathbf{y}_i \in \mathbb{R}^C$ is the ground truth label of $\mathbf{x}_i$, where $y_{ki} = 1$ if $\mathbf{x}_i$ belongs to class k and $y_{ki} = 0$ if not. $\|\cdot\|$ is the ℓ_2 norm for vectors and the Frobenius norm for matrices. $L(\cdot)$ is the loss function for classification. The problem can be rewritten as:

$$\min_{\mathbf{B},\mathbf{W}} J_2 = \sum_{i=1}^{n} \|\mathbf{y}_i - \mathbf{W}^T\mathbf{b}_i\|^2 + \lambda\|\mathbf{W}\|^2 \tag{5}$$

3.2. Preserving Distributions

Preserving distance and semantic similarity is an important way to generate meaningful hash codes. However, existing methods only take into account the relationship of one data point or data point-pairs. Since a good embedding needs to maintain not only the local structure but also global distribution of the data, we have utilized KL divergence as a means to constrain the distribution variation in our previous work (Zhang et al., 2018). However, as mentioned in Section 2.2, the JS divergence would be a better measure due to its symmetry and upper bound. Firstly, the symmetric property means not only that similar images should have similar binary codes, but also that similar binary codes should correspond to similar images. Secondly, the JS divergence has an upper bound which helps optimization process to converge. In fact, it is more stable, and robust to noise and the length of hash codes.

First, we define the similarity of $\mathbf{x}_i$ to $\mathbf{x}_j$ as the conditional probability $p_{j|i}$ in the original feature space where we use Euclidean distance to represent similarities between data points. This means that $\mathbf{x}_i$ would select $\mathbf{x}_j$ as its neighbor if neighbors are selected in proportion to their probability density function under the Gaussian distribution centered at $\mathbf{x}_i$. For nearby data points, $p_{j|i}$ is relatively high, whereas for widely separated data points, $p_{j|i}$ will be almost infinitesimal. This similarity measure matches the essence of retrieval. Therefore, the conditional probability can be formulated as:

$$p_{j|i} = \frac{\exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2/2\sigma_i^2)}{\sum_{k\neq i}\exp(-\|\mathbf{x}_i - \mathbf{x}_k\|^2/2\sigma_i^2)} \tag{6}$$

Furthermore, the joint probability can be derived as $p_{ij} = \frac{p_{i|j}+p_{j|i}}{2n}$.

Similarly, we can obtain the conditional probability in the low-dimensional Hamming space. Following t-SNE (Maaten and Hinton, 2008), in order to alleviate the crowding problem we use a probability distribution that has much heavier tails than a Gaussian to convert distances into probabilities. Specifically, we employ the Student t-distribution with one degree of freedom (which is the same as the Cauchy distribution) as the heavy-tailed distribution. The joint probability q_{ij} is defined as:

$$q_{ij} = \frac{(1 + \|\mathbf{b}_i - \mathbf{b}_j\|^2)^{-1}}{\sum_{k\neq l}(1 + \|\mathbf{b}_k - \mathbf{b}_l\|^2)^{-1}} \tag{7}$$

If the binary points $\mathbf{b}_i$ and $\mathbf{b}_j$ correctly model the similarity between the high-dimensional data points $\mathbf{x}_i$ and $\mathbf{x}_j$, the joint probabilities p_{ij} and q_{ij} will be equal. Therefore, our goal is to find a low-dimensional binary representation that minimizes the distance between p_{ij} and q_{ij}. The JS divergence provides a good choice for meeting this goal. It can be represented as follows:

$$J_3 = \sum JS(P_i\|Q_i) = \sum_i\sum_j (\frac{1}{2}p_{ij}\log\frac{p_{ij}}{m_{ij}} + \frac{1}{2}q_{ij}\log\frac{q_{ij}}{m_{ij}}) \tag{8}$$

where $m_{ij} = \frac{p_{ij}+q_{ij}}{2}$.

Combining equations (2), (5) and (8), we have the following formulation:

$$J = J_1 + \alpha J_2 + \beta J_3 \tag{9}$$

3.3. Optimization

In order to have a fair comparison with previous deep hashing methods, we also choose the CNN-F network to learn the feature representations and hash functions. Using pairwise-label supervision, our model consists of two separate CNNs which share the same weights. Each CNN includes 5 convolutional layers and 2 fully connected layers. The pipeline is shown in Fig. 1.

The minimization of the obtained loss function in Section 2.3 is a discrete optimization problem, which is hard to optimize directly. We solve this problem by introducing an auxiliary variable, namely the output of the last fully connected layer, $\mathbf{u}_i$ and make $\mathbf{b}_i = sgn(\mathbf{u}_i)$, which can be represented as:

$$\mathbf{u}_i = \mathbf{M}^T \phi(\mathbf{x}_i; \theta) + \mathbf{v} \tag{10}$$

where θ denotes all the parameters of the previous layers in the CNN, $\phi(\mathbf{x}_i; \theta)$ denotes the output of the penultimate fully connected layer, $\mathbf{M}$ represents the weight matrix, and $\mathbf{v}$ is the bias term. Then we can reformulate the optimization problem as the following equivalent one:

$$\begin{aligned} \min J' = & -\sum_{s_{ij}\in S}(s_{ij}\Psi_{ij} - \log(1+e^{\Psi_{ij}})) + \alpha\sum_{i=1}^{n}\|\mathbf{y}_i - \mathbf{W}^T\mathbf{u}_i\|^2 \\ & + \lambda\|\mathbf{W}\|^2 + \beta\sum_i\sum_j(\frac{1}{2}p_{ij}\log\frac{p_{ij}}{m_{ij}} + \frac{1}{2}q_{ij}\log\frac{q_{ij}}{m_{ij}}) \\ & + \eta\sum_{i=1}^{n}\|\mathbf{b}_i - \mathbf{u}_i\|_2^2 \end{aligned} \tag{11}$$

where $\Psi_{ij} = \frac{1}{2}\mathbf{u}_i{}^T\mathbf{u}_j$, and $q_{ij} = \frac{(1+\|\mathbf{u}_i-\mathbf{u}_j\|^2)^{-1}}{\sum_{k\neq l}(1+\|\mathbf{u}_k-\mathbf{u}_l\|^2)^{-1}}$.

In our method, an alternating strategy is used to learn these parameters. Specifically, we optimize one parameter with the other parameters fixed (or clamped). Firstly, $\mathbf{b}_i$ can be directly optimized by

$$\mathbf{b}_i = sgn(\mathbf{u}_i) = sgn(\mathbf{M}^T\phi(\mathbf{x}_i; \theta) + \mathbf{v}) \tag{12}$$

Then the remaining parameters can be optimized using the back-propagation (BP) algorithm.

4. Experiments

4.1. Datasets and Setting

To make a fair comparison, we conducted experiments on two widely used benchmark datasets: CIFAR-10 (Krizhevsky and Hinton, 2009) and NUS-WIDE (Chua et al., 2009).

- **CIFAR-10** dataset contains 60,000 color images of size 32*32 which are categorized into 10 classes with 6,000 images for each class. Each image is only associated with one class.

- **NUS-WIDE** contains nearly 270,000 color images collected from the web. It is a multi-label dataset in which each image is annotated with one or multiple class labels corresponding to 81 semantic concepts. Following (Xia et al., 2014; Zhang et al., 2015; Li et al., 2015, 2017), we used a subset of 195,834 images that belong to the 21 most frequent classes, and each class consists of at least 5000 images.

Like most previous work, the standard mean average precision (MAP) was used as the main metric to evaluate the performance of the proposed method together with the comparison baselines. For the two benchmark datasets, the similar pairs were constructed according to the image labels, i.e., two images were considered similar only if they shared at least one common semantic label.

We compare our method with several state-of-the-art hashing approaches. They can be roughly divided into traditional hashing and deep hashing. The traditional approaches can be further categorized into unsupervised and supervised methods. The unsupervised methods include SH (Weiss et al., 2009) and ITQ (Gong et al., 2013). Supervised methods include KSH (Liu et al., 2012), FastH (Lin et al., 2014), LFH (Zhang et al., 2014), and SDH (Shen et al., 2015). Both hand-crafted features and features extracted using a CNN-F network were used as the input for these traditional hashing methods. In a manner similar to previous work, we used a 512-dimensional GIST descriptor to represent each image of the CIFAR-10 dataset, and an 1134-dimensional feature vector for the NUS-WIDE dataset, which is the concatenation of a 64-D color histogram, a 144-D color correlogram, a 73-D edge direction histogram, a 128-D wavelet texture, a 225-D block-wise color moments and a 500-D BoW representation based on SIFT descriptors.

The compared deep hashing methods are CNNH (Xia et al., 2014), NINH (Lai et al., 2015), DSRH (Zhao et al., 2015), DSCH (Zhang et al., 2015), DRSCH (Zhang et al., 2015), DQN (Cao et al., 2016), DHN (Zhu et al., 2016), DPSH (Li et al., 2015), DTSH (Wang et al., 2016), and DSDH (Li et al., 2017). Although DPSH, DTSH, and DSDH are based on the CNN-F network and DQN, DHN, and DSRH are based on AlexNet, it is still comparable since both CNN-F and AlexNet consist of five convolutional layers and two fully connected layers. Here, most of the results were directly retrieved from the original papers.

We compare our method to baselines under the following two experimental settings. For the first setting, we randomly selected 100 images per class (1,000 images in total) as the test query set and 500 images per class (5,000 images in total) as the training set in CIFAR-10. For the NUS-WIDE, we randomly sampled 100 images per class (2,100 images in total) as the test query set and 500 images per class (10,500 images in total) as the training set. As for the second experimental setting, in CIFAR-10, 1,000 images per class were selected as the test query set, and the remaining 50,000 images were used as the training set. In NUS-WIDE, 100 images per class were randomly sampled as the test query images, and the remaining 193,734 images were used as the training set. Since NUS-WIDE contains a large number of images, we only considered the top 5,000 returned neighbors under the first setting and the top 50,000 under the second experimental setting when computing MAP for NUS-WIDE.

Table 1. Mean Average Precision(MAP) under the first experimental setting. The best performance is shown in boldface.

Method	CIFAR-10				NUS-WIDE			
	12 bits	24 bits	32 bits	48 bits	12 bits	24 bits	32 bits	48 bits
SH	0.127	0.128	0.126	0.129	0.454	0.406	0.405	0.400
ITQ	0.162	0.169	0.172	0.175	0.452	0.468	0.472	0.477
LFH	0.176	0.231	0.211	0.253	0.571	0.568	0.568	0.585
KSH	0.303	0.337	0.346	0.356	0.556	0.572	0.581	0.588
SDH	0.285	0.329	0.341	0.356	0.568	0.600	0.608	0.637
FastH	0.305	0.349	0.369	0.384	0.621	0.650	0.665	0.687
CNNH	0.439	0.511	0.509	0.522	0.611	0.618	0.625	0.608
NINH	0.552	0.566	0.558	0.581	0.674	0.697	0.713	0.715
DHN	0.555	0.594	0.603	0.621	0.708	0.735	0.748	0.758
DQN	0.554	0.558	0.564	0.580	0.768	0.776	0.783	0.792
DPSH	0.713	0.727	0.744	0.757	0.752	0.790	0.794	0.812
DTSH	0.710	0.750	0.765	0.774	0.773	0.808	0.812	0.824
DSDH	0.740	0.786	0.801	0.820	0.776	0.808	0.820	0.829
DISH	0.738	0.792	0.822	0.841	0.781	0.823	0.837	0.840
Ours	**0.771**	**0.817**	**0.839**	**0.858**	**0.801**	**0.833**	**0.849**	**0.861**

Table 2. Mean Average Precision(MAP) under the second experimental setting. The best performance is shown in boldface.

Method	CIFAR-10				NUS-WIDE			
	12 bits	24 bits	32 bits	48 bits	12 bits	24 bits	32 bits	48 bits
DSRH	0.608	0.611	0.617	0.618	0.609	0.618	0.621	0.631
DSCH	0.609	0.613	0.617	0.620	0.592	0.597	0.611	0.609
DRSCH	0.615	0.622	0.629	0.631	0.618	0.622	0.623	0.628
DPSH	0.763	0.781	0.795	0.807	0.715	0.722	0.736	0.741
DTSH	0.915	0.923	0.925	0.926	0.756	0.776	0.785	0.799
DSDH	0.935	0.940	0.939	0.939	0.815	0.814	0.820	0.821
DIDH	0.941	0.945	0.948	0.952	0.843	0.849	0.857	0.862
Ours	**0.953**	**0.959**	**0.961**	**0.970**	**0.851**	**0.859**	**0.864**	**0.871**

4.2. Results and Analysis

The MAP results of all methods on CIFAR-10 and NUS-WIDE for the first experimental setting are listed in Table 1. The table shows that the proposed method substantially outperforms all the methods compared. Specifically, we can see that there is a large margin between deep hashing and traditional hashing methods on CIFAR-10. Compared to the representative traditional methods, the MAP results delivered by our method are more than twice those from SDH, FastH, and ITQ. For the deep hashing methods, our proposed method considers both supervised information and distribution similarity, so it improves the performance of DSDH by on average 3.5% for different bit lengths. These results verify that our similarity and distribution preserving method is both effective and efficient in obtaining good binary codes. As for NUS-WIDE, it is also shown that our method outperforms the other methods compared by about 3%. Compared to our previous work DISH, the proposed method gives both better results and a more stable improvement on the two benchmarks. On the CIFAR-10 dataset, DISH with KL divergence performs poorly with short code length. For example, for 12 bits it does not outperform DSDH, while with the JS divergence the new method achieves stable and robust retrieval performance.

An important practical concern is the quality of retrieval results for the top N returned images. Hence improving the precision with respect to the top returned samples is an imperative. Fig. 2 shows the precision curve for the different number of top returned images with 64-bit hash codes. The experiment is conducted on CIFAR-10. These methods have overall stable performance when the number of returned images varies, and the precision gently converges when more than 800 images have been returned. In general, our method outperforms the alternative methods compared.

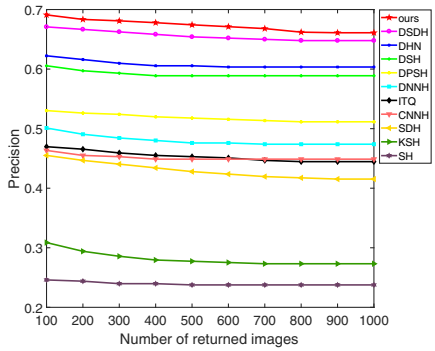


Fig. 2. Precision curve w.r.t top N @64 bits.

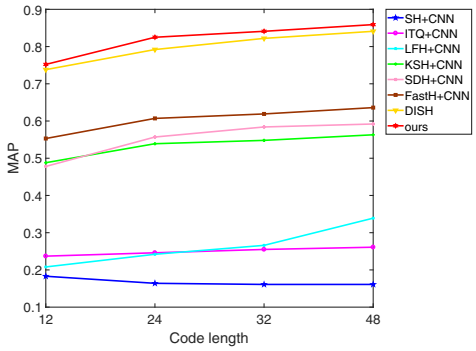


Fig. 3. Mean Average Precision(MAP) under the second experimental setting.

We also compare the different hashing methods under the second experimental setting. This involves a larger sample of training images. While the deep architecture requires large amounts of data for training, our method is based on preserving distributions, and distribution modeling in turn depends to a large extent on the amount of data available. As a result, more training images are conducive to the performance of these algorithms. Table 2 lists the MAP results for the different approaches compared. From the table almost all the deep hashing methods perform much better than when trained in the first experimental setting. This means that they are more suitable for large-scale datasets. With sufficient training data and adequate guidance by the selected loss function, our method is not only superior in performance to each of the alternative methods but also achieves a greater improvement in performance over that achieved with the first experimental setting.

To further verify the effectiveness of our proposed method, we compare it with some traditional hashing methods that use deep features extracted by CNN-F pre-trained on ImageNet. The results are reported in Fig. 3. We can see that all of the traditional hashing methods have a significant performance improvement when CNN features are used. In particular, the performance of FastH with CNN features on CIFAR-10 is nearly twice than that obtained with hand-crafted features. However, there is still a large gap between our method and the traditional approaches in terms of retrieval performance.

Conclusion

In this paper, we have proposed a novel deep hashing method. In addition to the commonly used pairwise label information and classification information, we have introduced the JS divergence to constrain the information loss during the hashing embedding process. This means we can preserve both the local and global structure of the data. Extensive experiments show that our method can achieve comparable performance to the state-of-the-art methods in image retrieval applications. In our future work, we will extend the probabilistic framework using additional elements from information theory. For example, using divergences to measure pairwise similarity would be an interesting direction.

Acknowledgments

This work was supported by the National Natural Science Foundation of China project no. 61772057, in part by Beijing Natural Science Foundation project no. 4162037, and the support funding from State Key Lab. of Software Development Environment, and Jiangxi and Qingdao Research Institute of Beihang University.

References

Bai, X., Yan, C., Yang, H., Bai, L., Zhou, J., Hancock, E.R., 2018. Adaptive hash retrieval with kernel based similarity. Pattern Recognition 75, 136–148.

Bai, X., Yang, H., Zhou, J., Ren, P., Cheng, J., 2014. Data-dependent hashing based on p-stable distribution. IEEE Transactions on Image Processing 23, 5033–5046.

Cao, Y., Long, M., Wang, J., Zhu, H., Wen, Q., 2016. Deep quantization network for efficient image retrieval., in: AAAI, pp. 3457–3463.

Chua, T.S., Tang, J., Hong, R., Li, H., Luo, Z., Zheng, Y., 2009. Nus-wide: a real-world web image database from national university of singapore, in: Proceedings of the ACM international conference on image and video retrieval, ACM. p. 48.

Erin Liong, V., Lu, J., Wang, G., Moulin, P., Zhou, J., 2015. Deep hashing for compact binary codes learning, in: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 2475–2483.

Gionis, A., Indyk, P., Motwani, R., et al., 1999. Similarity search in high dimensions via hashing, in: Vldb, pp. 518–529.

Gong, Y., Lazebnik, S., Gordo, A., Perronnin, F., 2013. Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval. IEEE Transactions on Pattern Analysis and Machine Intelligence 35, 2916–2929.

Hu, M., Yang, Y., Shen, F., Xie, N., Hong, R., Shen, H.T., 2019. Collective reconstructive embeddings for cross-modal hashing. IEEE Transactions on Image Processing 28, 2770–2784.

Hu, M., Yang, Y., Shen, F., Xie, N., Shen, H.T., 2018. Hashing with angular reconstructive embeddings. IEEE Transactions on Image Processing 27, 545–555.

Hu, Q., Wu, J., Cheng, J., Wu, L., Lu, H., 2017. Pseudo label based unsupervised deep discriminative hashing for image retrieval, in: Proceedings of the 2017 ACM on Multimedia Conference, ACM. pp. 1584–1590.

Huang, S., Xiong, Y., Zhang, Y., Wang, J., 2017. Unsupervised triplet hashing for fast image retrieval, in: Proceedings of the on Thematic Workshops of ACM Multimedia 2017, ACM. pp. 84–92.

Jegou, H., Douze, M., Schmid, C., 2011. Product quantization for nearest neighbor search. IEEE transactions on pattern analysis and machine intelligence 33, 117–128.

Krizhevsky, A., Hinton, G., 2009. Learning multiple layers of features from tiny images .

Kulis, B., Darrell, T., 2009. Learning to hash with binary reconstructive embeddings, in: Advances in neural information processing systems, pp. 1042–1050.

Lai, H., Pan, Y., Liu, Y., Yan, S., 2015. Simultaneous feature learning and hash coding with deep neural networks. arXiv preprint arXiv:1504.03410 .

Li, Q., Sun, Z., He, R., Tan, T., 2017. Deep supervised discrete hashing, in: Advances in Neural Information Processing Systems, pp. 2479–2488.

Li, W.J., Wang, S., Kang, W.C., 2015. Feature learning based deep supervised hashing with pairwise labels. arXiv preprint arXiv:1511.03855 .

Li, Z., Tang, J., 2015. Unsupervised feature selection via nonnegative spectral analysis and redundancy control. IEEE Transactions on Image Processing 24, 5343–5355.

Li, Z., Tang, J., Mei, T., 2018. Deep collaborative embedding for social image understanding. IEEE transactions on pattern analysis and machine intelligence .

Lin, G., Shen, C., Shi, Q., Van den Hengel, A., Suter, D., 2014. Fast supervised hashing with decision trees for high-dimensional data, in: Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on, IEEE. pp. 1971–1978.

Lin, G., Shen, C., Suter, D., Van Den Hengel, A., 2013. A general two-step approach to learning-based hashing, in: Computer Vision (ICCV), 2013 IEEE International Conference on, IEEE. pp. 2552–2559.

Lin, K., Lu, J., Chen, C.S., Zhou, J., 2016. Learning compact binary descriptors with unsupervised deep neural networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1183–1192.

Lin, K., Yang, H.F., Hsiao, J.H., Chen, C.S., 2015. Deep learning of binary hash codes for fast image retrieval, in: Computer Vision and Pattern Recognition Workshops (CVPRW), 2015 IEEE Conference on, IEEE. pp. 27–35.

Liu, H., Wang, R., Shan, S., Chen, X., 2016. Deep supervised hashing for fast image retrieval, in: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 2064–2072.

Liu, W., Mu, C., Kumar, S., Chang, S.F., 2014. Discrete graph hashing, in: Advances in neural information processing systems, pp. 3419–3427.

Liu, W., Wang, J., Ji, R., Jiang, Y.G., Chang, S.F., 2012. Supervised hashing with kernels, in: Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on, IEEE. pp. 2074–2081.

Liu, W., Wang, J., Kumar, S., Chang, S.F., 2011. Hashing with graphs, in: Proceedings of the 28th international conference on machine learning (ICML-11), Citeseer. pp. 1–8.

Luo, Y., Yang, Y., Shen, F., Huang, Z., Zhou, P., Shen, H.T., 2018. Robust discrete code modeling for supervised hashing. Pattern Recognition 75, 128–135.

Maaten, L.v.d., Hinton, G., 2008. Visualizing data using t-sne. Journal of machine learning research 9, 2579–2605.

Shen, F., Gao, X., Liu, L., Yang, Y., Shen, H.T., 2017a. Deep asymmetric pairwise hashing, in: Proceedings of the 25th ACM international conference on Multimedia, ACM. pp. 1522–1530.

Shen, F., Shen, C., Liu, W., Shen, H.T., 2015. Supervised discrete hashing., in: CVPR, p. 5.

Shen, F., Xu, Y., Liu, L., Yang, Y., Huang, Z., Shen, H.T., 2018. Unsupervised deep hashing with similarity-adaptive and discrete optimization. IEEE transactions on pattern analysis and machine intelligence 40, 3034–3044.

Shen, F., Yang, Y., Liu, L., Liu, W., Tao, D., Shen, H.T., 2017b. Asymmetric binary coding for image search. IEEE Transactions on Multimedia 19, 2022–2032.

Song, J., He, T., Gao, L., Xu, X., Shen, H.T., 2017. Deep region hashing for efficient large-scale instance search from images. arXiv preprint arXiv:1701.07901 .

Song, J., Yang, Y., Huang, Z., Shen, H.T., Luo, J., 2013. Effective multiple feature hashing for large-scale near-duplicate video retrieval. IEEE Transactions on Multimedia 15, 1997–2008.

Su, K., Yang, G., Yang, L., Li, D., Su, P., Yin, Y., 2019. Learning binary hash codes for finger vein image retrieval. Pattern Recognition Letters 117, 74–82.

Tang, J., Li, Z., 2018. Weakly supervised multimodal hashing for scalable social image retrieval. IEEE Transactions on Circuits and Systems for Video Technology 28, 2730–2741.

Tang, J., Li, Z., Wang, M., Zhao, R., 2015. Neighborhood discriminant hashing for large-scale image retrieval. IEEE Transactions on Image Processing 24, 2827–2840.

Tang, J., Li, Z., Zhu, X., 2018. Supervised deep hashing for scalable face image retrieval. Pattern Recognition 75, 25–32.

Wang, C., Bai, X., Wang, S., Zhou, J., Ren, P., 2019. Multiscale visual attention networks for object detection in vhr remote sensing images. IEEE Geoscience and Remote Sensing Letters 16, 310–314.

Wang, J., Zhang, T., Sebe, N., Shen, H.T., et al., 2018. A survey on learning to hash. IEEE transactions on pattern analysis and machine intelligence 40, 769–790.

Wang, X., Shi, Y., Kitani, K.M., 2016. Deep supervised hashing with triplet labels, in: Asian Conference on Computer Vision, Springer. pp. 70–84.

Weiss, Y., Torralba, A., Fergus, R., 2009. Spectral hashing, in: Advances in neural information processing systems, pp. 1753–1760.

Xia, R., Pan, Y., Lai, H., Liu, C., Yan, S., 2014. Supervised hashing for image retrieval via image representation learning., in: AAAI, p. 2.

Xiao, B., Hancock, E.R., Wilson, R.C., 2009. Graph characteristics from the heat kernel trace. Pattern Recognition 42, 2589–2606.

Yan, C., Bai, X., Wang, S., Zhou, J., Hancock, E.R., 2019. Cross-modal hashing with semantic deep embedding. Neurocomputing 337, 58–66.

Yang, H., Bai, X., Liu, Y., Wang, Y., Bai, L., Zhou, J., Tang, W., 2016. Maximum margin hashing with supervised information. Multimedia Tools and Applications 75, 3955–3971.

Zhang, P., Zhang, W., Li, W.J., Guo, M., 2014. Supervised hashing with latent factor models, in: Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval, ACM. pp. 173–182.

Zhang, R., Lin, L., Zhang, R., Zuo, W., Zhang, L., 2015. Bit-scalable deep hashing with regularized similarity learning for image retrieval and person re-identification. IEEE Transactions on Image Processing 24, 4766–4779.

Zhang, X., Zhou, L., Bai, X., Hancock, E., 2018. Deep supervised hashing with information loss, in: Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR), Springer. pp. 395–405.

Zhao, F., Huang, Y., Wang, L., Tan, T., 2015. Deep semantic ranking based hashing for multi-label image retrieval, in: Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on, IEEE. pp. 1556–1564.

Zhu, H., Long, M., Wang, J., Cao, Y., 2016. Deep hashing network for efficient similarity retrieval., in: AAAI, pp. 2415–2421.

Conflict of interest

The authors declared that they have no conflicts of interest to this work.
We declare that we do not have any commercial or associative interest that represents a conflict of interest in connection with the work submitted.