Deep Discriminative Supervised Hashing via Siamese Network

Yang Li†, Zhuang Miao†, Jiabao Wang†, Yafei Zhang†, Nonmembers, and Hang Li†, Member

SUMMARY  The latest deep hashing methods perform hash codes learning and image feature learning simultaneously by using pairwise or triplet labels. However, generating all possible pairwise or triplet labels from the training dataset can quickly become intractable, where the majority of those samples may produce small costs, resulting in slow convergence. In this letter, we propose a novel deep discriminative supervised hashing method, called DDSH, which directly learns hash codes based on a new combined loss function. Compared to previous methods, our method can take full advantages of the annotated data in terms of pairwise similarity and image identities. Extensive experiments on standard benchmarks demonstrate that our method preserves the instance-level similarity and outperforms state-of-the-art deep hashing methods in the image retrieval application. Remarkably, our 16-bits binary representation can surpass the performance of existing 48-bits binary representation, which demonstrates that our method can effectively improve the speed and precision of large scale image retrieval systems.

key words: hashing, convolutional neural network, siamese network, image retrieval

1. Introduction

With the dramatic increase of image data on the Internet, large scale image retrieval has attracted intensive attention in the computer vision society [1]. Among existing image retrieval techniques, hashing has become one of the most powerful and effective techniques due to its computation efficiency and low memory cost.

Over the past few decades, a variety of hashing methods have been proposed to boost the performance of approximate nearest neighbor (ANN) search [2]. These methods can be divided into two main categories: data-independent methods and data-dependent methods. Data-independent methods use random projections to construct hash functions which are independent of any training data, such as locality sensitive hashing (LSH) [3] and shift-invariant kernels hashing (SIKH) [4]. Compared with data-dependent methods, data-independent methods usually require longer codes to achieve satisfactory performance. Hence, data-dependent methods are rapidly becoming the dominant approach in real applications. Representative data-dependent methods include spectral hashing (SH) [5], iterative quantization (ITQ) [6], supervised discrete hashing (SDH) [7], sequential projection learning for hashing (SPLH) [8], latent factor hashing (LFH) [9], supervised hashing with kernels (KSH) [10], fast supervised hashing (FastH) [11], etc.

However, most existing data-dependent hashing methods are based on a hand-crafted feature construction procedure, which is independent of the hash function learning procedure. Hence, hand-crafted features (e.g. SIFT [12] or GIST [13]) cannot be optimally compatible with hash function learning and result in sub-optimal hash codes. Convolutional neural network hashing (CNNH) [14] is the first deep hashing method to perform feature learning for supervised hashing with triplet labels. However, CNNH is a two-stage method. The learned image representations in the second stage cannot give feedback for learning better approximate hash codes in the first stage. Network in network hashing (NINH) [15] proposed a one-stage deep hashing method with a triplet ranking loss. Deep semantic ranking-based hashing (DSRH) [16] proposed to preserve the relative similarity between images and semantic structures on images simultaneously by a triplet ranking loss with orthogonality constraint. Deep regularized similarity comparison hashing (DRSCH) [17] extended the triplet-based relative comparison by incorporating a regularization term for hashing learning. Deep pairwise supervised hashing (DPSH) [18] proposed a simultaneous feature learning and hash codes learning method by maximizing the likelihood of pairwise similarities. Deep triplet supervised hashing (DTSH) [19] presented a novel formulation of the likelihood of the given triplet labels to evaluate the quality of learned hash codes and achieved the highest performance compared to other deep hashing methods. Nevertheless, generating all possible triplet labels from the training dataset can quickly become intractable, where the majority of those samples may produce small costs, resulting in slow convergence [20].

The above-mentioned previous works demonstrate that the loss functions have a significant influence for hashing learning. And we believe that the loss functions and network structures being explored for hashing learning can be further improved. Therefore, in this letter, we propose a novel deep discriminative supervised hashing (DDSH) method to simultaneously predict hash codes and image labels in an end-to-end manner. The main contributions of DDSH can be summarized as follows:

- We propose a deep discriminative supervised hashing method via siamese network to simultaneously learn discriminative image features and hash codes in an end-
to-end manner. This approach significantly improves the representation ability of hash codes.

- We present a new combined loss function that has two components: pairwise hashing loss and classification loss. Compared to previous networks, we take full advantages of the annotated data in terms of pairwise similarity and image identities. To be best of our knowledge, this is the first attempt to use such a combined loss function to improve the discriminative ability of hash codes.
- Extensive experiments on standard benchmarks demonstrate that our DDSH method preserves the instance-level similarity and outperforms state-of-the-art deep hashing methods in the image retrieval application.

2. Methodology

As shown in Fig. 1, the flowchart of the proposed end-to-end deep learning architecture for our DDSH method contains two components. The first component is a deep siamese network to learn image representations from pixels and map the learned image representations to hash codes. And the second component is a combined loss function, which has two parts: pairwise hashing loss and classification loss. All the two components are seamlessly integrated into our DDSH framework in an end-to-end way. Hence, different components can give feedback to each other in our learning procedure, which can produce better hash codes than other methods without end-to-end architecture.

2.1 Siamese Network Architecture

The overall siamese network architecture of the proposed DDSH method is illustrated in Fig. 1. It is essentially a two-branch convolutional siamese network that takes a pair of images as input and aims to learn deep hashing representations. However, these two convolutional neural networks (CNNs) have exactly the same structures and parameters.

Our DDSH model contains a VGG-F model [21] as a component, from which we have removed the final fully-connected layer. Hence, only the first 7 layers of VGG-F are used in this component. Various CNN architectures, such as GoogleNet [22] and ResNet [23], can also be used in our DDSH model. But it is not the focus of this letter to study different networks. Let \( \theta \) denote all the parameters of the first seven layers of the VGG-F model. Given an input pair of images \( x_i \) and \( x_j \), the proposed siamese network produces two \( D \)-dimensional vector of output \( y_i = \phi(x_i, \theta) \) and \( y_j = \phi(x_j, \theta) \). Then, \( y_i \) and \( y_j \) will enter a combined loss unit where the operation will simultaneously predict the label of the two images and the hash codes similarity score.

2.2 The Combined Loss Function

As shown in Fig. 1, the combined loss function has two parts: pairwise hashing loss and classification loss. The pairwise hashing loss takes a pair of hash features and learns to distinguish whether they come from the same image label or not. In the meantime, the classification loss learns to classify each deep feature output of the base network into a class corresponding to the input image label.

**Pairwise hashing loss.** The pairwise hashing loss unit first takes two feature vectors \( y_i \) and \( y_j \) as input, then \( y_i \) and \( y_j \) are fed into a fully-connected layer with \( c \) nodes, and its output are binary codes which are defined as follows:

\[
\begin{align*}
\mathbf{b}_i &= h(x_i) = sgn(W^T y_i + v) \\
&= sgn(W^T \phi(x_i, \theta) + v) = sgn(u_i) 
\end{align*}
\]

where \( h(x_i) \) denotes the hash function need to learn. \( W \) denotes the weight matrix connecting the output of the VGG-F model and the fully-connected layer with \( c \) nodes, \( v \) is the bias vector. And \( u_i = W^T \phi(x_i, \theta) + v \). Based on the learned binary codes \( \mathbf{b}_i \) and \( \mathbf{b}_j \) of the input pair of images \( x_i \) and \( x_j \), we can define the pairwise loss function similar to that of LFH [9]:

\[
L_p = -\log p(S|\mathcal{B}) = -\sum_{s_{ij} \in \mathcal{S}} p(s_{ij}|\mathcal{B}) = -\sum_{s_{ij} \in \mathcal{S}} (s_{ij}\Theta_{ij} - \log(1 + e^{\Theta_{ij}}))
\]

where \( \mathcal{S} = \{s_{ij}\} \) denotes a set of pairwise labels with \( s_{ij} \in \{0, 1\} \). \( s_{ij} = 0 \) means that \( x_i \) and \( x_j \) are dissimilar, \( s_{ij} = 1 \) means that \( x_i \) and \( x_j \) are similar. \( \mathcal{B} = \{\mathbf{b}_i\}^n_{i=1} \) denotes the binary codes set which should preserve the similarity in \( \mathcal{S} \). \( \Theta_{ij} = \frac{1}{2} \mathbf{b}_i^T \mathbf{b}_j \) denotes half of the inner product between two hash codes \( \mathbf{b}_i \) and \( \mathbf{b}_j \).

The pairwise loss function in (2) reflects the negative log-likelihood of the observed pairwise labels in \( \mathcal{S} \). Minimizing this pairwise loss function \( L_p \) will make the Hamming distance between two similar images smaller and simultaneously make the dissimilar images larger. This exactly matches the goal of supervised hashing with pairwise labels. However, the major problem in the pairwise loss is that they only use weak image labels, and do not take all the annotated information into consideration.

**Classification loss.** The classification loss unit directly learns the non-linear functions from an input image to the
image label and cross-entropy loss is used to distinguishes different image from each other. After the features are extracted from the base VGG-F network, a softmax layer with \( n \) nodes are then connected, where \( n \) is the class number in the training set. As conventional multi-class recognition approaches, we can define the classification loss function:

\[
L_c = - \sum_{i=1}^{m} \log \frac{e^{Q_i^T y + b_i}}{\sum_{j=1}^{n} e^{Q_j^T y + b_j}}
\]

where \( y \) denotes the output deep feature of the VGG-F model and the fully-connected layer with \( n \) nodes, \( v \) is the bias vector. The size of mini-batch is \( m \). The major drawback of classification loss is that they do not directly learning hash function. And classification loss cannot account for the similarity measurement between image pairs, which can be problematic during the image retrieval process.

The above-mentioned observations show that the pairwise hashing loss and classification loss have complementary advantages and limitations. Therefore, we propose combining the strengths of the two loss functions and leverage their complementary aspects to improve the discriminative ability of the learned hash codes. The joint supervision of combined loss is:

\[
L = L_p + L_c = - \sum_{a,b \in S} (s_{ab} \Theta_{ab} - \log(1 + e^{\Theta_{ab}})) - \sum_{i=1}^{m} \log \frac{e^{Q_i^T y + b_i}}{\sum_{j=1}^{n} e^{Q_j^T y + b_j}}
\]

Clearly, the siamese network supervised by the combined loss is trainable and can be optimized by the standard stochastic gradient descent (SGD) method. The pairwise hashing loss can minimise the variance of the two distributions and the mean distances of images belonging to the same class. And the classification loss can maximise the mean distances of images between non-matching pairs. During training, we first compute all the gradients produced by pairwise hashing loss and classification loss respectively and add the weighted gradients together to update the network. After training procedure, for any image \( x_i \), we can predict its hash code just by forward propagation, i.e., \( h_i = h(x_i) = \text{sgn}(W^T \phi(x_i, \theta) + v) \).

3. Experiments

3.1 Dataset and Experimental Setup

We evaluate the performance of our method and other hash based algorithms on CIFAR-10 dataset. The CIFAR-10 dataset contains 60,000 color images of size \( 32 \times 32 \), which can be divided into 10 categories (6,000 images for each category). We resize all the images to \( 224 \times 224 \). In the first experiment, we randomly sampled 1,000 samples, 100 per class, as the query data, and used the remaining 59,000 images as the database images. During the training procedure, we randomly select 5,000 images (500 images per class) from the database images as the training set. In the second experiment, we randomly sampled 10,000 samples, 1,000 per class, as the query data, and used the remaining 50,000 images as the database images. During the training procedure, all the 50,000 database images are used as training images. Note that for fair comparison, the above experimental setting and evaluation metric are exactly the same as that in [18], [19]. We compare our method with several state-of-the-art hashing methods. These methods can be categorized into three classes:

- Traditional data-dependent methods with hand-crafted features (512-dimensional GIST descriptor), including SH [5], ITQ [6], SPLH [8], KSH [10], FastH [11], LFH [9], and SDH [7].
- Deep hashing methods with pairwise labels, including CNNH [14] and DPSH [18].
- Deep hashing methods with triplet labels, including NIH [15], DSRH [16], DSH [17], DRSH [17] and DTSH [19].

All experiments were implemented in MATLAB using MatConvNet toolbox [24] with Intel (R) Core (TM) i5-4690 CPU (3.50GHz), 16 GB DDR3. We initialize the first seven layers of our DDSH model with the VGG-F network pre-trained on ImageNet, and fine-tuning our model during training. The max training epoch is set to 150 epochs. The mini-batch size is set to 128. Our model can be trained at 32 images per second. We employ mean average precision (MAP) to evaluate the performance of our method and other baselines similar to most previous work [15], [17]–[19]. We extract the hash code for each image and the MAP score is computed by the Hamming distance of hash codes.

3.2 Experimental Results

In the first experimental setting, the MAP of Hamming ranking results of our method and other previously published results are summarized together in Table 1. The first and second best results are highlighted by **bold** and underline.

### Table 1 Mean average precision of Hamming ranking for different number of bits under the first experimental setting

<table>
<thead>
<tr>
<th>Methods</th>
<th>12-bits</th>
<th>24-bits</th>
<th>32-bits</th>
<th>48-bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ours (DDSH)</td>
<td>0.762</td>
<td>0.795</td>
<td>0.809</td>
<td>0.818</td>
</tr>
<tr>
<td>DTSH [19]</td>
<td>0.710</td>
<td>0.750</td>
<td>0.765</td>
<td>0.774</td>
</tr>
<tr>
<td>DPSH [18]</td>
<td>0.713</td>
<td>0.727</td>
<td>0.744</td>
<td>0.757</td>
</tr>
<tr>
<td>NIH [15]</td>
<td>0.552</td>
<td>0.566</td>
<td>0.558</td>
<td>0.581</td>
</tr>
<tr>
<td>CNNH [14]</td>
<td>0.439</td>
<td>0.511</td>
<td>0.509</td>
<td>0.522</td>
</tr>
<tr>
<td>FastH [11]</td>
<td>0.305</td>
<td>0.349</td>
<td>0.369</td>
<td>0.384</td>
</tr>
<tr>
<td>SDH [7]</td>
<td>0.285</td>
<td>0.329</td>
<td>0.341</td>
<td>0.356</td>
</tr>
<tr>
<td>KSH [10]</td>
<td>0.303</td>
<td>0.337</td>
<td>0.346</td>
<td>0.356</td>
</tr>
<tr>
<td>LFH [9]</td>
<td>0.176</td>
<td>0.231</td>
<td>0.211</td>
<td>0.253</td>
</tr>
<tr>
<td>SPLH [8]</td>
<td>0.171</td>
<td>0.173</td>
<td>0.178</td>
<td>0.184</td>
</tr>
<tr>
<td>ITQ [6]</td>
<td>0.162</td>
<td>0.169</td>
<td>0.172</td>
<td>0.175</td>
</tr>
<tr>
<td>SH [5]</td>
<td>0.127</td>
<td>0.128</td>
<td>0.126</td>
<td>0.129</td>
</tr>
</tbody>
</table>
methods with hand-crafted features and deep hashing methods with pairwise or triplet labels. This shows that the combined loss function can notably enhance the discriminative power of deeply learned hash codes, demonstrating the effectiveness of our DDSH method. Among the state-of-the-art methods in the literature, DTSH [19] achieves the best results. However, DTSH method used triplet labels, which suffers from dramatic data expansion when constituting the sample triplets from the training set. Consequently, our DDSH method is more efficient and easy to implement, which outperforms DTSH by more than 5%.

To further verify the hash learning ability of our method, we use more training data in the second experiment. And we compare our DDSH method to DSRH, DSCH, DRSCH, DPSH and DTSH under the second experimental setting. From Table 2, we can find that our DDSH method can achieve more higher accuracy with more training data. Please note that DSRH, DSCH, DRSCH, DPSH and DTSH can also perform simultaneously hash codes learning and feature learning in an end-to-end framework. But our DDSH method also gives significant improvements over the state-of-the-art deep hashing methods across different number of bits. Remarkably, the retrieval accuracy of our method with 16-bits can surpass the DTSH method [19] with 48-bits (ours 0.944 MAP vs DTSH 0.926 MAP). Once the feature bits are 3 times lower, the memory consumption will decrease 3 times and the retrieval efficiency will be significantly improved. Therefore, our method can effectively improve the speed and precision of image retrieval systems.

### 4. Conclusions

This letter presents a new method for end-to-end hash learning. Our method proposed using a siamese network trained with a joint supervision of pairwise hashing loss and classification loss. This is the first time such a combined loss function has been applied in hash learning. And this new method overcomes a variety of issues that have previously arisen when using only pairwise or triplet labels. We argue that the superior results provided by our method are due to the highly complementary between pairwise hashing loss and classification loss. Comprehensive experiments convincingly demonstrated that our method is more suitable for producing better descriptive and compact hash representations for large scale image retrieval.

### References


