Discrete Spatial Importance-Based Deep Weighted Hashing

Yang Shi[1], Xiushan Nie[2*], Quan Zhou[1], Xiaoming Xi[2], and Yilong Yin[1**]

[1] Shandong University, 1500 Shunhua Road, Jinan, Shandong Province, China
[2] Shandong Jianzhu University, 1000 Fengming Road, Jinan, Shandong Province, China

Abstract. Hashing is a widely used technique for large-scale approximate nearest neighbor searching in multimedia retrieval. Recent works have proved that using deep neural networks is a promising solution for learning both feature representation and hash codes. However, most existing deep hashing methods directly learn hash codes from a convolutional neural network, ignoring the spatial importance distribution of images. The loss of spatial importance negatively affects the performance of hash learning and thus reduces its accuracy. To address this issue, we propose a new deep hashing method with weighted spatial information, which generates hash codes by using discrete spatial importance distribution. In particular, to extract the discrete spatial importance information of images effectively, we propose a method to learn the spatial attention map and hash code simultaneously, which makes the spatial attention map more conductive to hash-based retrieval. The experimental results of three widely used datasets show that the proposed deep weighted hashing method is superior to the state-of-the-art hashing method.

Keywords: Hashing, neural networks, spatial importance

1 Introduction

With the rapid developments in science and technology, people can use the sensors around them more conveniently, which greatly increases the amount of data uploaded through the sensors, including a large number of images and videos. Therefore, the ability to deal with this data has become an urgent problem, and the approximate nearest search is a method to solve it. As one of the approximate nearest search methods, hashing maps the high-dimensional data into a compact binary code in the Hamming space. Hashing greatly reduces the required storage space and the calculation of the hamming distance of XOR binary codes is much

* Corresponding author: Xiushan Nie and Yilong Yin.
** This work was supported in part by the National Natural Science Foundation of China (61671274, 61876098), National Key R&D Program of China (2018YFC0830100, 2018YFC0830102) and special funds for distinguished professors of Shandong Jianzhu University.

faster than that of the Euclidean distances of the original data. Because of its excellent storage capacity and efficient computing power, hashing has attracted wide attention from researchers.

At the same time, with the rapid development of deep learning research, deep model has achieved remarkable improvements in computer vision performance including image classification [1], image retrieval [2], video classification [3] and other directions [4] [5] [6] [7]. Therefore, many deep hashing methods have been developed, which can learn both deep features and hash codes. At present, the key concept of deep hashing methods is to obtain the hash codes of images by putting the images directly into convolutional neural networks (CNNs). Unfortunately, this simple operation does not distinguish the importance of the region of the image, every pixel in the image has the same impact on the hash code learning, which may lead to suboptimal performance [8]. Recently, the research on fully convolutional networks (FCN) [9]in image positioning shows that the feature map of a convolution layer can retain a great deal of spatial information. Therefore, the ability to optimize hash codes with this spatial information has become an important new research topic.

To fully use the spatial importance information, in this study, we propose a deep hashing method with weighted discrete spatial importance (SIWH). To explore spatial importance information, we designed an effective hash-based spatial attention model to adaptively learn the spatial importance closely related to the target. Meanwhile, we also used a CNN to extract semantic information, which is much richer in detail than the handcrafted features of images [10].

In summary, we emphasize the following three contributions:

(1) We propose a unified framework to learn weighted hash codes using discrete spatial importance distributions, which can assign different hash code lengths to images according to their spatial importance. After reviewing available research, we believe that this is the first attempt to learn weighted hashing according to the importance of spatial position.

(2) We develop an effective hash-guide spatial attention model. In this model, the spatial attention network and hash learning network are trained simultaneously, which makes the spatial information conducive to hash code generation.

(3) We make an extensive evaluation of three widely used image retrieval benchmarks. The experimental results show that our method significantly outperforms the state-of-the-art methods' results, which demonstrates the superiority and effectiveness of our method.

The rest of this paper is organized as follows. In Section 2, we briefly review related works. The proposed method is introduced in Section 3. The extensive experiments and discussions of the experimental results are provided in Section 4, and we present conclusions in Section 5.

2 Related Works

In this section, we will briefly review the related hashing methods, including shallow-based and deep-based models.

In general, hashing methods can be divided into two categories data-independent and data-dependent [11]. In the early years of these methods' development, researchers focused on data-independent hashing methods, such as LSH [12] and its variants (SKLSH) [13]. The LSH method generates hash codes through random projection. However, to achieve satisfactory performance, these data-independent methods often need a long hash code.

To obtain more compact binary codes, data-dependent hash methods have been proposed in more recent years. These methods attempt to learn hash functions from a training set, and they can be divided into unsupervised methods and supervised methods[14]. Unsupervised methods only use unmarked training data to learn hash functions. For example, spectral hashing (SH) [15] minimizes the weighted hamming distance of image pairs, where the weight is defined as the similarity measure of the image pairs. Iterative quantization (ITQ) [16] attempts to minimize quantization errors on projected image descriptors to reduce the information loss caused by the difference between the real value feature space and the binary hamming space.

Compared to unsupervised methods, supervised methods provide better accuracy due to the usage of label information. Predictable discriminative binaries (DBC) [17] look for hyperplanes as hash functions that separate categories with large margins. Minimal loss hashing (MLH) [18] optimizes the upper bound of hinged losses to learn the hash function.

Meanwhile, semi-supervised hashing (SSH) [19], uses rich unmarked data to standardize hash functions. Although the above methods use linear projections as hash functions, they have difficulty in handling linearly indivisible data. To overcome this limitation, supervised hashing with kernels (KSH) [20] and binary reconstructive embedding (BRE) [21] are proposed to study the similarity–preserving hash function in kernel space.

With the wide application of deep learning in recent years, deep hashing frameworks have attracted more and more attention in hashing methods. Deep hashing (DH) [22] uses nonlinear deep networks to generate improved hash codes. Fang et al.[23] and Lai et al.[24] were the first to attempt to combine learning feature representations and hash functions. Lu et al.[25] used a deep network with a stacked fully connected layer to construct multiple hierarchical nonlinear transforms to learn binary hash codes. Deep supervised hashing (DSH) [2] tries to maintain similarity and minimize binary loss. A supervised semantics preserving hash (SSDH) [26] constructs the underlying hashing layer to generate hash code by directly minimizing classification errors on the hashing layer output. Li et al.[27] proposed deep pairwise labels supervised hashing (DPSH), which uses pairwise labels to perform both feature learning and hash code learning. Deep supervised discrete hashing (DSDH) [28] develops DPSH based on the ideal assumption that hash codes should be classified. Cao et al.[29] proposed deep Cauchy hashing (DCH), which designed loss functions based on the Cauchy distribution to generate highly concentrated hash codes. Jiang et al.[30] proposed the Deep discrete supervised hashing (DDSH), which uses pairwise supervised information to directly guide the discrete coding program and the

deep characteristic learning program. Jiang et al.[31] proposed asymmetric deep supervised hashing (ADSH), which processes query points and database points in an asymmetric path. Deep ordinal hashing (DOH [32]) learns a group of ranking-based hash functions by jointly exploiting the local spatial information and the global semantic information. In general, deep supervised hashing can significantly improve performance.

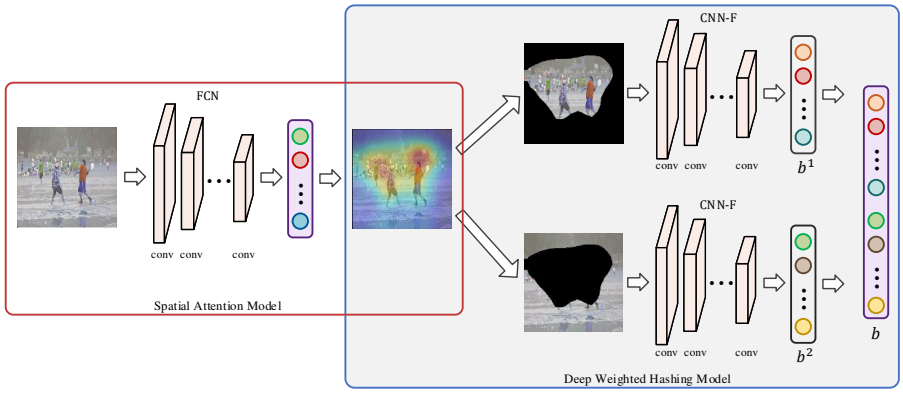


Fig. 1. Framework of the proposed SIWH method, which consists of two components: Spatial Attention Model and Deep Weighted Hashing Model

However, the aforementioned deep-learning-based hashing methods use the deep model to generate semantics of whole images, which ignore the spatial importance distribution of the images. To fully use this information, in this work, we propose a spatial attention module to learn a weighted hash. In the proposed method, we first use the spatial attention model to obtain the importance of each position, and then learn the weighted hash function with the allocation of the hash code length appropriately according to the different importance information.

At the same time, attention models have been extensively studied in image/video subtitling [33] and object detection [8]. [34] proposed that attention model can selectively learn significant areas of images in specific visual tasks. And [4] proposed attention model to learn the attention position of image subtitle. Although attention model has been successfully used in various visual tasks, it is still not fully utilized in image retrieval.

3 Discrete Spatial Importance-Based Deep Weighted Hashing

In this section, we describe the proposed deep hashing framework in detail. The entire framework is shown in Fig. 1. Firstly, the spatial attention model is used to

generate the spatial attention map for the input image by FCN network. Then, the attention mask is generated according to the spatial attention map, and the attention and non-attention parts of the input image are divided by the mask. After that, the attention and the non-attention parts are put into CNN network learning hash function. Finally, the learned hash codes are combined to generate the final hash code.

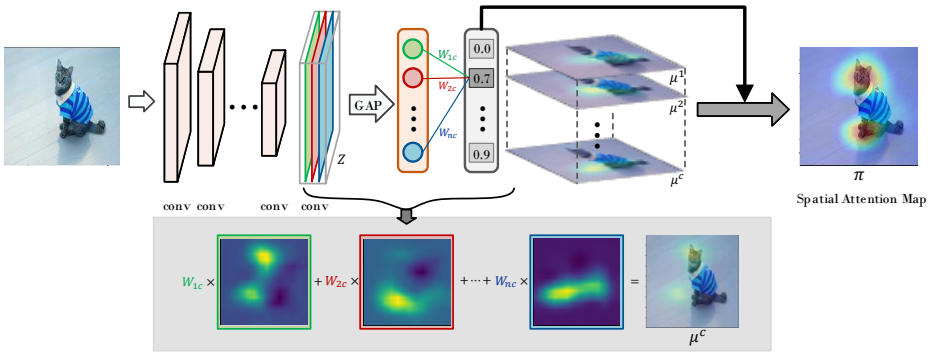


Fig. 2. An imaginary illustration of the Spatial Attention Model.

3.1 Notation and Definition

Suppose $\mathbf{X} = \{x_n\}_{n=1}^{N}$ be a set of N images, $\mathbf{Y} = \{\mathbf{y}_n\}_{n=1}^{N}$ is the set of their corresponding binary label vectors, where $\{\mathbf{y}_n\} \in \{0,1\}^C$, and C defines the total number of the categories. The non-zero entry in $\mathbf{y}_n$ indicates that the n^{th} image belongs to the corresponding class. Let $\mathbf{S} = \{s_{ij}\} \in \{0,1\}^{N \times N}$ be the similarity matrix, where $s_{ij} = 1$ if the image pair (x_i, x_j) shares at least one common class, otherwise $s_{ij} = 0$. Hash coding learns a collection of K-bit binary codes $\mathbf{B} \in \{0,1\}^{K \times N}$, where the n^{th} column $\mathbf{b}_n \in \{0,1\}^K$denotes the binary codes for the n^{th} sample x_n. Our goal is to learn a set of mappings $H : x \rightarrow \mathbf{b}$, which maps the image x into a hash code $\mathbf{b}$.

3.2 Network Architecture

The proposed method uses two networks, in which the FCN model aims to capture spatial principal distribution information using the spatial attention model, and the CNN model explores semantic information. For CNN networks, we utilize CNN-F [35] as the backbone, where the first five convolutional layers and the first two fully connected layers of CNN-F are denoted as $conv1$, $conv2$, $conv3$, $conv4$, $conv5$, $fc6$ and $fc7$, respectively. Based on these layers, we design a task-specific fully connected layer fck. For FCN networks, we use a FCN network with VGG [36] as its backbone to extract the attention position. The FCN network

adopts the first 5 convolutional layers of the VGG network, containing $conv1$, $conv2$, $conv3$, $conv4$, and $conv5$. In the proposed network, we use a convolutional layer $conv6$ to replace the first fully connected layer $fc6$ of the VGG network. Behind the above layers, there is a fully connected fcc layer for classification prediction. In particular, we perform global average pooling on the $conv5$ layer and provide the output to the fully connected classification layer (i.e. fcc). The average of these feature maps will be used to generate the probability output of the fcc layer. Furthermore, we introduce an intuitive method for extracting image spatial importance by utilizing the weight matrix of the fcc layer and the feature map of the $conv5$ layer. Like the CNN network, the $conv6$ layer is designed to learn visual descriptors by encoding spatial information.

3.3 Spatial Attention Model

Intuitively, when searching for relevant images in the database, we always pay more attention on the regions that are highly related to the main objects or peoples in the image. Consequently, we introduce a hash-guide spatial attention model, which aims to generate object-specific spatial attention map for hash learning. Fig. 2 illustrates the generation of the spatial attention map.

As is shown in Fig. 2, we use the Full Convolution Network with the global average pooling to generate the attention locations of the image. Specifically, we perform the global average pooling on $conv5$ layer and provide the output to the fully-connected classification layer (fcc). The outputs of the global average pooling can be regarded as the spatial average of feature maps of the $conv5$ layer. Those spatial average values are used to generate the probabilistic outputs of the fcc layer.

In this section, we introduce an intuitive way to produce the spatial attention map by projecting the weight matrix of the fcc layer on the feature maps of the $conv5$ layer.

Let $\mathbf{Z}_{uv}$ define the channel-wise representation at the spatial location (u, v) of the $conv5$ layer of the FCN network, which can be computed as

$$\mathbf{Z}_{uv} = \Psi_F(x; \mathbf{\Omega}_F), \tag{1}$$

where $\mathbf{Z}_{uv} \in \mathbb{R}^{M_F}$ with M_F being the number of feature maps. The notations $u \in \{1, \cdots, U\}$ and $v \in \{1, \cdots, V\}$, where U and V are the width and height of feature maps. The notation x represents the input image, and Ψ_F defines the non-linear projection function for the FCN network. The notation $\mathbf{\Omega}_F = \{\mathbf{W}_F^d, \mathbf{c}_F^d\}_{d=1}^{D_F}$ defines a set of non-linear projection parameters with D_F being the depth of the FCN network.

Considering the weight matrix, $\mathbf{W} \in \mathbb{R}^{M_F \times C}$ performs a mapping of the spatial average values to the semantic class labels. We define the object-specific local response at the spatial location (u, v) as μ_{uv}^c, which can be obtained by

$$\mu_{uv}^c = max(\mathbf{w}_c^T \mathbf{z}_{uv}, 0), \ for\, c = 1, \cdots, C, \tag{2}$$

where $\mathbf{w}_c$ is the c^{th} column of $\mathbf{W}$. μ_{uv}^c indicates the importance at the spatial location (u, v) in the input image x which is classified to the c^{th} class. As each spatial location (u, v) corresponds to a different local patch in the original image, the local response μ_{uv}^c indicates the relative similarity of the local image patch to the c^{th} class.

Essentially, we can obtain local discriminative information at different spatial locations for a particular class. The larger value of μ_{uv}^c indicates that the image patch at the spatial location (u, v) is more related to the c^{th} class. Inversely, the smaller value of μ_{uv}^c indicates it is less related to the c^{th} class. Therefore, μ^c is the attention map for class c. By integrating μ^c together, we can define the spatial attention map π to identify all the object-specific image patches. Specifically, the local response at (u, v), denoted as π_{uv}, is defined as follows

$$\pi_{uv} = \sum_{c=1}^{C} p_c \mu_{uv}^c / \sum_{c=1}^{C} p_c \tag{3}$$

where p_c is the c^{th} probabilistic output of the classification (i.e., fcc) layer of FCN network.

3.4 Deep Weighted Hashing Model

Given a query image, users typically pay more attention to the attention area of the query image. Therefore, differently weighted hash codes should be assigned between attention and non-attention areas during hash learning. To achieve this purpose, in the proposed SIWH, we first generate an attention mask using the spatial attention map, and then learn weighted hash codes for different attention areas.

Using the attention map obtained before, we assign 1 to the location whose response value is equal or greater than the threshold value t, and assign a 0 to the rest, generating the attention mask $\mathbf{A}$. We can also swap the 0 and the 1 in $\mathbf{A}$ to generate $\bar{\mathbf{A}}$. Then, we expand the attention mask $\mathbf{A}$ to the size of the image x, which is $\mathbf{A}'$. We then multiply $\mathbf{A}'$ with the image x to get the attention image of x, which is denoted as x^1. Similarly, we use $\bar{\mathbf{A}}$ and the image x to generate x^2.

Next, we put the attention image x^1 and non-attention image x^2 into two CNN networks to learn hash codes. First, we define the feature representation extracted from the $fc8$ layer of the CNN network as $\mathbf{g}^1$ and $\mathbf{g}^2$, computed by

$$\mathbf{g}^1 = \Psi_{C^1}(x^1; \mathbf{\Omega}_{C^1}),\ \mathbf{g}^2 = \Psi_{C^2}(x^2; \mathbf{\Omega}_{C^2}), \tag{4}$$

where $\mathbf{g}^1 \in \mathbb{R}^{M_{C^1}}$ and $\mathbf{g}^2 \in \mathbb{R}^{M_{C^2}}$. The notation Ψ_{C^1} and Ψ_{C^2} define the non-linear projection function for the two CNN networks, respectively. $\mathbf{\Omega}_{C^1} = \{\mathbf{W}_{C^1}^d, \mathbf{c}_{C^1}^d\}_{d=1}^{D_{C^1}}$ and $\mathbf{\Omega}_{C^2} = \{\mathbf{W}_{C^2}^d, \mathbf{c}_{C^2}^d\}_{d=1}^{D_{C^2}}$ define two sets of non-linear projection parameters with D_{C^1} and D_{C^2} being the depths of the CNN networks, respectively.

We can use a symbolic function to get the hash code $\mathbf{b}^1$ and $\mathbf{b}^2$, which are computed by

$$\mathbf{b}^1 = sgn(\mathbf{g}^1),\ \mathbf{b}^2 = sgn(\mathbf{g}^2). \tag{5}$$

Then, we can define $\mathbf{b}$ as the whole hash code, which can be calculated as $\mathbf{b} = \mathbf{b}^1 \bigoplus \mathbf{b}^2$, where $\bigoplus$ denotes the connect operation. We define K as the code length of $\mathbf{b}$, K^1 and K^2 are the code lengths of $\mathbf{b}^1$ and $\mathbf{b}^2$ respectively, and $K = K^1 + K^2$.

3.5 Objective Function

As mentioned earlier, given the binary codes $\mathbf{B} = \{\mathbf{b}_i\}_{i=1}^N$ and the pairwise labels $\mathbf{S} = \{s_{ij}\}$ for all the images, we can define the likelihood of the pairwise labels [37]:

$$p(s_{ij} \mid \mathbf{B}) = \begin{cases} \sigma(\theta_{ij}), & s_{ij} = 1 \\ 1 - \sigma(\theta_{ij}), & s_{ij} = 0 \end{cases} \tag{6}$$

where $\theta_{ij} = \frac{1}{2}\mathbf{b}_i^T\mathbf{b}_j$, and $\sigma(\theta_{ij}) = \frac{1}{1+e^{-\theta_{ij}}}$. Please note that $\mathbf{b}_i \in \{-1, 1\}^K$.

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

$$\begin{aligned} \min_{\mathbf{B}} L_s &= -\log p(\mathbf{S} \mid \mathbf{B}) = -\sum_{s_{ij} \in \mathbf{S}} \log p(s_{ij} \mid \mathbf{B}) \\ &= -\sum_{s_{ij} \in \mathbf{S}} (s_{ij}\theta_{ij} - \log(1 + e^{\theta_{ij}})). \end{aligned} \tag{7}$$

It is easy to find that the above optimization problem can make the hamming distance between two similar points as small as possible and the hamming distance between two different similar points as large as possible. This fits perfectly with the goal of supervised hash of paired tags.

The problem in (7) is a discrete optimization problem, which is difficult to solve. However, we can solve this problem by directly transferring $\mathbf{b}_i$ from discrete value to a continuous relaxation $\mathbf{d}_i$. Therefore, a quantization loss term is included as follows:

$$\min_{\mathbf{B}} L_q = \sum_{i=1}^{N} \parallel \mathbf{b}_i - \mathbf{d}_i \parallel_2^2, \tag{8}$$

where $\mathbf{d}_i$ is the real value output of the network.

In the FCN, the classification loss term is defined as the cross entropy loss, which is expressed as follows:

$$\min_{\mathbf{B}} L_c = -\sum_{i=1}^{N} (y_i \log \bar{y}_i + (1 - y_i) \log(1 - \bar{y}_i)), \tag{9}$$

where $\bar{y}_i$ is the prediction value output of the network.

Thus, the final objective function is as follows:

$$\begin{aligned} \min_{\mathbf{B}} L &= L_s + \eta L_q + \beta L_c \\ &= -\sum_{s_{ij}\in\mathbf{S}} (s_{ij}\theta_{ij} - \log(1+e^{\theta_{ij}})) \\ &+ \eta \sum_{i=1}^{N} \parallel \mathbf{b}_i - \mathbf{d}_i \parallel_2^2 \\ &- \beta \sum_{i=1}^{N} (y_i \log \bar{y}_i + (1-y_i)\log(1-\bar{y}_i)), \end{aligned} \tag{10}$$

where η and β are the hyper-parameters.

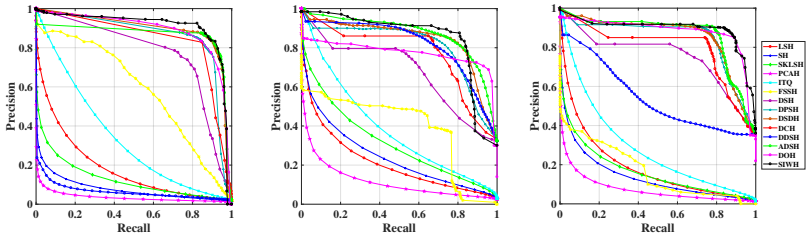
Among these terms, we need to optimize parameters for $\mathbf{\Omega}_F = \{\mathbf{W}_F^d, \mathbf{c}_F^d\}_{d=1}^{D_F}$ and $\mathbf{\Omega}_{C^*} = \{\mathbf{W}_{C^*}^d, \mathbf{c}_{C^*}^d\}_{d=1}^{D_{C^*}}$. The parameters $\mathbf{W}_*^d$ and $\mathbf{c}_*^d$ can be automatically updated by applying SGD with BP algorithm in pytorch.

4 Experiment

In this section, we report the results of our extensive experiments to verify the efficiency of the proposed method (SIWH) on three widely used image retrieval datasets, including CIFAR-10 [38], NUS-WIDE[39], and MS-COCO [40]. The CIFAR-10 dataset is a single-label dataset, while the NUS-WIDE and MS-COCO dataset are multi-label datasets.

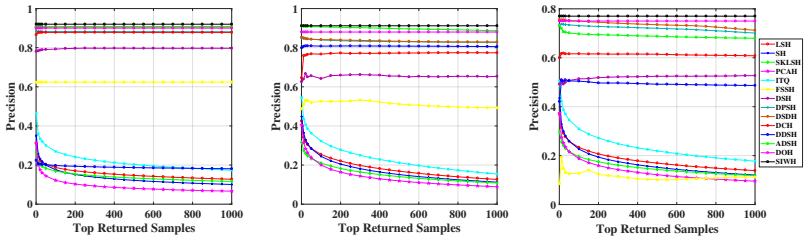
Table 1. The results of mAP with respect to different code lengths on the three datasets.

Method	CIFAR10				NUS-WIDE				MS-COCO			
	24 bits	48 bits	64 bits	128 bits	24 bits	48 bits	64 bits	128 bits	24 bits	48 bits	64 bits	128 bits
LSH	0.2722	0.3586	0.4490	0.4887	0.0654	0.1882	0.2993	0.3900	0.0868	0.1462	0.1774	0.3007
SH	0.2346	0.2959	0.3187	0.5168	0.1238	0.1729	0.2358	0.3448	0.0837	0.1048	0.1289	0.2373
SKLSH	0.2378	0.2983	0.3872	0.5517	0.0922	0.1387	0.2596	0.4354	0.0551	0.1369	0.1893	0.3966
PCAH	0.1430	0.1720	0.1863	0.2018	0.0924	0.0809	0.0890	0.1131	0.0662	0.0633	0.0702	0.0918
ITQ	0.3648	0.4245	0.4283	0.4502	0.3109	0.3884	0.4139	0.4571	0.2289	0.2862	0.3085	0.3515
FSSH	0.6853	0.7124	0.6919	0.7204	0.3959	0.3716	0.4462	0.5411	0.3105	0.3415	0.4063	0.4316
DSH	0.7864	0.7830	0.7834	0.7835	0.6598	0.6653	0.6587	0.6598	0.5153	0.5069	0.5147	0.5072
DPSH	0.8821	0.8853	0.8858	0.8876	0.8390	0.8429	0.8423	0.8468	0.6623	0.6871	0.6965	0.7073
DSDH	0.8985	0.9004	0.9002	0.8970	0.8225	0.8328	0.8347	0.8415	0.6988	0.7191	0.7220	0.7227
DCH	0.8753	0.8752	0.8749	0.8273	0.7552	0.7632	0.7647	0.7602	0.5858	0.5954	0.5948	0.5953
DDSH	0.8452	0.8861	0.8916	0.8993	0.7352	0.8102	0.8083	0.7957	0.5821	0.6032	0.6142	0.6162
ADSH	0.8957	0.9040	0.9060	0.9059	0.8582	0.8893	0.8890	0.8873	0.6351	0.6376	0.6508	0.6617
DOH	0.8651	0.8616	0.8697	0.8739	0.7830	0.7846	0.7808	0.7935	0.7336	0.7487	0.7537	0.7521
SIWH	**0.9231**	**0.9304**	**0.9294**	**0.9316**	**0.8673**	**0.8948**	**0.9023**	**0.9109**	**0.7522**	**0.7986**	**0.8007**	**0.8153**



(a) CIFAR10 (b) NUS-WIDE (c) MS-COCO

Fig. 3. The results of Precision-recall curves with respect to 64-bit hash code on three datasets.



(a) CIFAR10 (b) NUS-WIDE (c) MS-COCO

Fig. 4. The results of P@N curves with respect to 64-bit hash code on three datasets.

4.1 Baselines and Evaluation Metrics

We compare the proposed SIWH method with thirteen state-of-the-art image hashing methods, including six non-deep hashing methods (LSH [12], SH [15], SKLSH [13], PCAH [41], ITQ [16], and FSSH [42]) and seven deep hashing methods (ADSH [31], DPSH [27], DSH [2], DDSH [30], DSDH [28], DCH [29], and DOH [32]). We have briefly reviewed these hashing methods in Section 2 above. Among them, LSH, SH, and SHLSH are unsupervised hash methods, and the rest are supervised hash methods.

For the non-deep methods, we use the CNN-F network to extract the 4096-dimensional features of the $fc7$ layer. For all deep methods, we use the same network (CNN-F) for equivalent comparison. The parameters of all comparison methods were selected as their default values.

In addition, we use three widely used indicators to evaluate search performance: mean average accuracy (mAP), top N precision (P@N), and Precision-Recall curves (PR).

4.2 Experimental Settings

We implemented the proposed SIWH method using the open source pytorch framework on the NVIDIA TITAN XP GPU server. In addition to the layers

from VGG, $conv1$ to $conv5$, and $fc6$ to $fc7$, we use "kaiming" initialization to initialize the network. Our network is trained by using a small batch random gradient descent with learning speed set to 10^{-5}. In all experiments, we fixed the minimum batch size to be 16. The hyperparameters η and β of the two loss functions are set to 1e-2 and 3e-2 respectively.

SIWH involves two hyperparameters, namely the threshold value of the attention mask t and the proportion of the attention hash code K^1/K. For t, we use a linear search to select t in 0.5, 0.625, 0.75, 0, 875. Specifically, for the CIFAR-10 data set, we set t to 0.875 and for the NUS-WIDE and MS-COCO data sets, we set t to 0.75. For K^1/K, we conducted a comparative experiment of the parameters.

4.3 Experimental Results and Analysis

(1) **Accuracy.** The mAP results for SIWH and all baselines are shown in Table 1. From Table 1, we observe that SIWH is significantly better than the comparative baselines from different datasets with different lengths of code. Compared with the best deep hash method ADSH, the mAP values of SIWH implemented on the CIFAR10 and NUS-WIDE datasets achieve an average performance improvement of 2.57% and 1.29%, respectively. When compared to the existing best deep hash method DOH on the MS-COCO dateset, SIWH achieves an average performance improvement of 4.47%. The substantial improvements demonstrate the effectiveness of the proposed method.

In Fig. 3, we plot the PR curves for a 64-bit code length on the three datasets. Specifically, the PR curve represents overall performance, and the PR curve near the upper right indicates superior performance. As shown in Figure 3, SIWH achieves superior performance compared to the benchmarks of the other methods on all datasets. Additionally, SIWH can achieve higher accuracy at lower recall points, which is preferred for an actual image retrieval system.

In Fig. 4, we report the performance of P@N in terms of a 64-bit code length, and dependent on the number of top samples returned. We observe that SIWH performs better than any of the comparison methods on different datasets.

Overall, it was observed from the experimental results that in terms of mAP, P@N, and PR curves, SIWH was significantly better than the baseline for all comparisons on the different datasets. Such a major improvement shows the superiority of the proposed hashing method.

Table 2. The mAP results of Our method and its variants.

Method	CIFAR10				NUS-WIDE				MS-COCO			
	24 bits	48 bits	64 bits	128 bits	24 bits	48 bits	64 bits	128 bits	24 bits	48 bits	64 bits	128 bits
SIWH-wa	0.7853	0.7809	0.7951	0.8095	0.5947	0.6698	0.7084	0.7260	0.5570	0.5727	0.5922	0.6043
SIWH-oa	0.8343	0.8444	0.8400	0.8566	0.6337	0.6421	0.6445	0.6526	0.5370	0.5509	0.5821	0.5891
SIWH-ona	0.4601	0.4794	0.4466	0.4558	0.2163	0.2352	0.2553	0.2131	0.2327	0.2326	0.2461	0.2434
SIWH	**0.9231**	**0.9304**	**0.9294**	**0.9316**	**0.8673**	**0.8948**	**0.9023**	**0.9109**	**0.7522**	**0.7986**	**0.8007**	**0.8153**

Table 3. The mAP results of Our method with different K^1/K.

K^1/K (%)	CIFAR10				NUS-WIDE				MS-COCO			
	24 bits	48 bits	64 bits	128 bits	24 bits	48 bits	64 bits	128 bits	24 bits	48 bits	64 bits	128 bits
87.5	0.8941	0.8944	0.8996	0.8936	0.8479	0.8295	0.8363	0.8452	0.7281	0.7653	0.7979	0.7959
75	**0.9231**	**0.9304**	**0.9294**	**0.9316**	**0.8673**	**0.8948**	**0.9023**	**0.9109**	**0.7522**	**0.7986**	**0.8007**	**0.8153**
62.5	0.8927	0.8843	0.8816	0.8922	0.8451	0.8370	0.8453	0.8476	0.7058	0.7556	0.7936	0.7904
50	0.8713	0.8790	0.8821	0.8857	0.8238	0.8272	8247	0.8215	0.7066	0.7646	0.7807	0.7907

Table 4. The mAP results of Our method With Different Attention.

Method	CIFAR10				NUS-WIDE				MS-COCO			
	24 bits	48 bits	64 bits	128 bits	24 bits	48 bits	64 bits	128 bits	24 bits	48 bits	64 bits	128 bits
classification	0.8921	0.8990	0.9055	0.8964	0.8097	0.8018	0.8192	0.8016	0.6566	0.6655	0.6606	0.6737
hashing	**0.9231**	**0.9304**	**0.9294**	**0.9316**	**0.8673**	**0.8948**	**0.9023**	**0.9109**	**0.7522**	**0.7986**	**0.8007**	**0.8153**

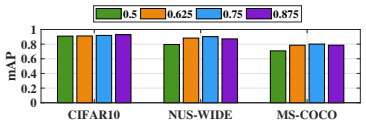


Fig. 5. The mAP with respect to different t for 64-bits hash code on three datasets.

(2) **Comparison With Variants.** In the proposed deep network, we construct a spatial attention model to learn spatial importance distribution information and combine it with a CNN network to learn hash codes uniformly. To prove the influence of spatial attention models, we studied three variations of SIWH: 1) SIWH-wa: the proposed SIWH learns hash codes without an attention model; 2) SIWH-oa: the proposed SIWH learns hash codes only using the attention region; and 3) SIWH-ona: the proposed SIWH learns hash codes only using the non-attention region.

Table 2 presents the mAP results for SIWH and the three variations described above. As we can see, the proposed method is superior to SIWH using no attention method. Therefore, the spatial attention model utilized by the proposed hashing method can be used to generate an identifiable hash code and will produce higher retrieval performance. Additionally, our method is significantly better than using either the attention or non-attention area alone. This result demonstrates that our integration is effective.

(3) **Effect Of Parameter.** One of the parameters involved in the proposed SIWH is the threshold value of attention mask t. To verify sensitivity, we performed experiments to analyze the effects on different datasets by using a linear

search in 0.5, 0.625, 0.75, 0.875. The retrieval performance expressed by mAP is shown in Figure 5. For the CIFAR10 data set, the performance impact under different settings is very small. However, on NUS-WIDE and MS-COCO, when t is set to 0.5, the performance is slightly reduced. As shown in Figure 5, for CIFAR-10, we can set t to 0.875. For NUS-WIDE and MS-COCO, we can set t to 0.75.

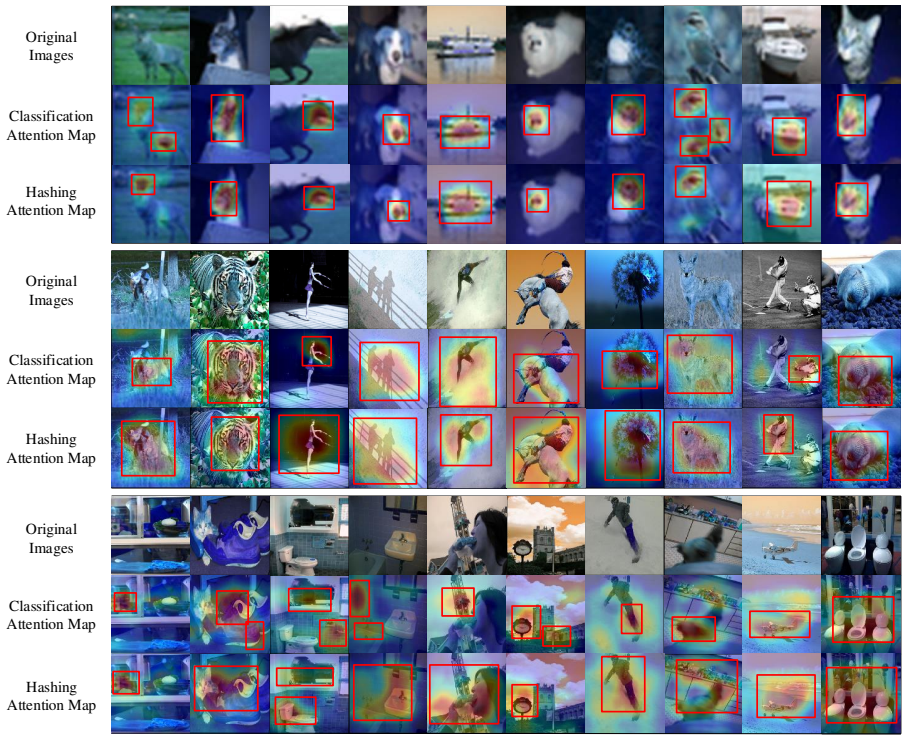


Fig. 6. Some visual examples of spatial attention maps for CIFAR-10, NUS-WIDE and MS-COCO datasets, respectively. The first line shows the original images. The second line represents the classification attention maps. The third line displays the DWSH-D hashing attention maps. The bottom line indicates the DWSH-C hashing attention maps.

The proposed SIWH also involves another parameter, the proportion of the attention hash code K^1/K. We set different values for it, and the retrieval performance represented by mAP is shown in Table 3. We found that on all three datasets, when K1/K was set to 0.75, the performance was superior to the rest.

(4) **Comparison With Classification Attention.** Generally, most attention models are used for classification. However, in the proposed method SIWH,

the attention model is guided by the hash learning process. In fact, there exists many differences between classification-guided and hash-guided attention map generation. Some visual examples of spatial attention maps are shown in Fig. 6, including classification-guided and hash-guided attention maps. It can be seen that both attention models can learn to distinguish the attention regions, which are indicated by the red border areas, and we can also see that there are some differences in the attention regions selected.

In the proposed method, the generation of the attention map is guided by hash learning. Therefore, we explored the performance difference between the hash-guided attention model and classification-guided attention model, and the mAP performance is shown in Table 4, where the mAP of hash-guide model is superior to that of the classification-guided model. It is indicated that the proposed attention model is beneficial to hash-based retrieval tasks.

5 Conclusions

In this study, we propose a novel deep weighted hashing method that learns hash codes by discrete spatial importance. Specifically, two network branches are designed to generate hash codes while simultaneously learning the spatial importance distribution. The proposed method can generate distinguishable hash codes with high quality, thus achieving excellent performance in image retrieval. Many experimental results on three datasets verify the superiority of the proposed method in learning hash codes for image retrieval. In future work, we will investigate how to learn the weights automatically for the weighted hash codes.

References

1. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Neural Information Processing Systems. (2012) 1097–1105
2. Liu, H., Wang, R., Shan, S., Chen, X.: Deep supervised hashing for fast image retrieval. In: computer vision and pattern recognition. (2016) 2064–2072
3. Liong, V.E., Lu, J., Tan, Y.P., Zhou, J.: Deep video hashing. IEEE Transactions on Multimedia **19** (2017) 1209–1219
4. Long, C., Zhang, H., Xiao, J., Nie, L., Chua, T.S.: Sca-cnn: Spatial and channel-wise attention in convolutional networks for image captioning. In: Computer Vision and Pattern Recognition. (2017) 6298–6306
5. Nie, X., Li, X., Chai, Y., Cui, C., Xi, X., Yin, Y.: Robust image fingerprinting based on feature point relationship mining. IEEE Transactions on Information Forensics and Security **13** (2018) 1509–1523
6. Nie, X., Jing, W., Cui, C., Zhang, J., Zhu, L., Yin, Y.: Joint multi-view hashing for large-scale near-duplicate video retrieval. IEEE Transactions on Knowledge and Data Engineering (2019) 1–1
7. Nie, X., Yin, Y., Sun, J., Liu, J., Cui, C.: Comprehensive feature-based robust video fingerprinting using tensor model. IEEE Transactions on Multimedia **19** (2016) 785–796

8. Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., Torralba, A.: Learning deep features for discriminative localization. In: Computer Vision and Pattern Recognition. (2015) 2921–2929
9. Shelhamer, E., Long, J., Darrell, T.: Fully convolutional networks for semantic segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence **39** (2017) 640–651
10. Oquab, M., Bottou, L., Laptev, I., Sivic, J.: Learning and transferring mid-level image representations using convolutional neural networks. In: Computer Vision and Pattern Recognition. (2014) 1717–1724
11. Wang, J., Zhang, T., Song, J., Sebe, N., Shen, H.T.: A survey on learning to hash. IEEE Transactions on Pattern Analysis and Machine Intelligence **40** (2018) 769–790
12. Andoni, A., Indyk, P.: Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. Communications of the ACM **51** (2008) 117–122
13. Liuz, Y., Cuiz, J., Huang, Z., Liz, H., Shenx, H.T.: Sk-lsh : An efficient index structure for approximate nearest neighbor search. Proceedings of the Vldb Endowment **7** (2014) 745–756
14. Wang, J., Liu, W., Kumar, S., Chang, S.F.: Learning to hash for indexing big data - a survey. Proceedings of the IEEE **104** (2016) 34–57
15. Weiss, Y., Torralba, A., Fergus, R.: Spectral hashing. In: International Conference on Neural Information Processing Systems. (2008) 1753–1760
16. Yunchao, G., Svetlana, L., Albert, G., Florent, P.: Iterative quantization: a procrustean approach to learning binary codes for large-scale image retrieval. IEEE Transactions on Pattern Analysis and Machine Intelligence **35** (2013) 2916–2929
17. Rastegari, M., Farhadi, A., Forsyth, D.: Attribute discovery via predictable discriminative binary codes. In: European Conference on Computer Vision. (2012) 876–889
18. Norouzi, M., Blei, D.M.: Minimal loss hashing for compact binary codes. In: Proceedings of the 28th international conference on machine learning (ICML-11), Citeseer (2011) 353–360
19. Wang, J., Kumar, S., Chang, S.: Semi-supervised hashing for large-scale search. IEEE Transactions on Pattern Analysis and Machine Intelligence **34** (2012) 2393–2406
20. Liu, W., Wang, J., Ji, R., Jiang, Y.G., Chang, S.F.: Supervised hashing with kernels. In: Computer Vision and Pattern Recognition, IEEE (2012) 2074–2081
21. Kulis, B., Darrell, T.: Learning to hash with binary reconstructive embeddings. In: Advances in neural information processing systems. (2009) 1042–1050
22. Liong, V.E., Lu, J., Gang, W., Moulin, P., Jie, Z.: Deep hashing for compact binary codes learning. In: Computer Vision and Pattern Recognition. (2015) 2475–2483
23. Fang, Z., Huang, Y., Liang, W., Tan, T.: Deep semantic ranking based hashing for multi-label image retrieval. In: Computer Vision and Pattern Recognition. (2015) 1556–1564
24. Lai, H., Pan, Y., Liu, Y., Yan, S.: Simultaneous feature learning and hash coding with deep neural networks. In: computer vision and pattern recognition. (2015) 3270–3278
25. Lu, J., Liong, V.E., Zhou, J.: Deep hashing for scalable image search. IEEE Transactions on Image Processing **26** (2017) 2352–2367
26. Yang, H., Lin, K., Chen, C.: Supervised learning of semantics-preserving hash via deep convolutional neural networks. IEEE Transactions on Pattern Analysis and Machine Intelligence **40** (2018) 437–451

27. Li, W.J., Wang, S., Kang, W.C.: Feature learning based deep supervised hashing with pairwise labels. In: International Joint Conference on Artificial Intelligence. (2016) 1711–1717
28. Li, Q., Sun, Z., He, R., Tan, T.: Deep supervised discrete hashing. In: Advances in Neural Information Processing Systems. (2017) 2479–2488
29. Cao, Y., Long, M., Liu, B., Wang, J.: Deep cauchy hashing for hamming space retrieval. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2018) 1229–1237
30. Jiang, Q.Y., Cui, X., Li, W.J.: Deep discrete supervised hashing. IEEE Transactions on Image Processing **27** (2018) 5996–6009
31. Jiang, Q.Y., Li, W.J.: Asymmetric deep supervised hashing. In: Thirty-Second AAAI Conference on Artificial Intelligence. (2018) 3342–3349
32. Jin, L., Shu, X., Li, K., Li, Z., Qi, G., Tang, J.: Deep ordinal hashing with spatial attention. IEEE Transactions on Image Processing **28** (2019) 2173–2186
33. You, Q., Jin, H., Wang, Z., Chen, F., Luo, J.: Image captioning with semantic attention. In: Computer Vision and Pattern Recognition. (2016) 4651–4659
34. Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhutdinov, R., Zemel, R., Bengio, Y.: Show, attend and tell: Neural image caption generation with visual attention. Computer Science (2015) 2048–2057
35. Chatfield, K., Simonyan, K., Vedaldi, A., Zisserman, A.: Return of the devil in the details: Delving deep into convolutional nets. arXiv preprint arXiv:1405.3531 (2014)
36. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. Computer Science (2014)
37. Zhang, P., Zhang, W., Li, W.J., Guo, M.: Supervised hashing with latent factor models. In: International ACM SIGIR Conference on Research and Development in Information Retrieval. (2014) 173–182
38. Krizhevsky, A., Hinton, G.: Learning multiple layers of features from tiny images. Technical report, Citeseer (2009)
39. Chua, T.S., Tang, J., Hong, R., Li, H., Luo, Z., Zheng, Y.: Nus-wide: a real-world web image database from national university of singapore. In: Proceedings of the ACM international conference on image and video retrieval, ACM (2009) 48
40. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: European conference on computer vision, Springer (2014) 740–755
41. Wang, X.J., Zhang, L., Jing, F., Ma, W.Y.: Annosearch: Image auto-annotation by search. In: Computer Vision and Pattern Recognition. Volume 2. (2006) 1483–1490
42. Luo, X., Nie, L., He, X., Wu, Y., Chen, Z.D., Xu, X.S.: Fast scalable supervised hashing. In: International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM (2018) 735–744