Hindawi
Applied Computational Intelligence and Soft Computing
Volume 2017, Article ID 9635348, 8 pages
https://doi.org/10.1155/2017/9635348

Research Article

Deep Hashing Based Fusing Index Method for Large-Scale Image Retrieval

Lijuan Duan,[1,2] Chongyang Zhao,[1,3] Jun Miao,[4] Yuanhua Qiao,[5] and Xing Su[1]

[1] *Faculty of Information Technology, Beijing University of Technology, Beijing 100124, China*
[2] *Beijing Key Laboratory on Integration and Analysis of Large-Scale Stream Data, Beijing, China*
[3] *National Engineering Laboratory for Critical Technologies of Information Security Classified Protection, Beijing 100124, China*
[4] *School of Computer Science, Beijing Information Science and Technology University, Beijing 100101, China*
[5] *College of Applied Science, Beijing University of Technology, Beijing 100124, China*

Correspondence should be addressed to Xing Su; xingsu@bjut.edu.cn

Received 31 March 2017; Accepted 26 April 2017; Published 24 May 2017

Academic Editor: Ridha Ejbali

Hashing has been widely deployed to perform the Approximate Nearest Neighbor (ANN) search for the large-scale image retrieval to solve the problem of storage and retrieval efficiency. Recently, deep hashing methods have been proposed to perform the simultaneous feature learning and the hash code learning with deep neural networks. Even though deep hashing has shown the better performance than traditional hashing methods with handcrafted features, the learned compact hash code from one deep hashing network may not provide the full representation of an image. In this paper, we propose a novel hashing indexing method, called the Deep Hashing based Fusing Index (DHFI), to generate a more compact hash code which has stronger expression ability and distinction capability. In our method, we train two different architecture's deep hashing subnetworks and fuse the hash codes generated by the two subnetworks together to unify images. Experiments on two real datasets show that our method can outperform state-of-the-art image retrieval applications.

1. Introduction

With the rapidly growing of images on the Internet, it is extremely difficult to find relevant images according to different people's needs. For example, nowadays the volume of images is becoming larger and larger, and a database having millions of images is quite common. Thus, a great deal of time and memory would be used in a linear search through the whole database. Moreover, images are always represented by real-valued features, so that the curse of dimension often occurred in many content-based image search engines and applications.

To address the inefficiency and the problem of memory cost of real-valued features, the ANN search [1] has become a popular method and a hot research topic in recent years. Among existing ANN techniques, hashing approaches are proposed to map images to compact binary codes that approximately preserve the data structure in the original space [2–6]. Due to the high query speed and low memory cost, the hashing and image binarization techniques have become the most popular and effective techniques to enhance identification and retrieval of information using content-based image recognition [4, 7–16]. Instead of real-valued features, images are represented by binary codes so that the time and memory costs of search can be greatly reduced [17]. However, the retrieval performance of most existing hashing methods heavily depends on the features they used, which are basically extracted in an unsupervised manner, thus more suitable for dealing with the visual similarity search than the semantic similarity search.

As we all know, the Convolutional Neural Network (CNN) has demonstrated its impressive learning power on image classification [5, 18–20], object detection [21], face recognition [22], and many other vision tasks [23–25]. The CNN used in these tasks can be regarded as a feature extractor guided by the objective function, specifically designed for the individual task [5]. The successful applications of CNN in various tasks imply that the features learned by CNN can well capture the underlying semantic structure of images in spite of significant appearance variations. Moreover, hashing with the deep learning network has shown that both feature representation and hash coding can be learned more effectively.

Inspired by the robustness of CNN features and the high performance of deep hashing methods, we propose a binary code generating and fusing framework to index large-scale image datasets, named Deep Hashing based Fusing Index (DHFI).

In our method, firstly, we train two different deep pairwise hashing networks which take image pairs along with labels to indicate whether the two images are similar as training inputs and produce binary codes as outputs. Then, we merge the hash codes produced by the two subnetworks together and regard the merged hash code as a fingerprint or binary index of an image. Under these two stages, images can be easily encoded by forward propagating through the network and then merging the network outputs to binary hash code representation.

The rest of the paper is organized as follows: Section 2 discusses the related work to the method. Section 3 describes DHFI method in detail. Section 4 extensively evaluates the proposed method on two large-scale datasets. Section 5 gives concluding remarks.

2. Related Work

Existing learning methods can be divided into two categories: data-independent methods and data-dependent methods [8, 24, 26, 27].

The hash function in data-independent methods is typically randomly generated and is independent of any training data. The representative data-independent methods include locality-sensitive hashing (LSH) [1] and its variants. Data-dependent methods try to learn the hash function from some training data, which is also called learning to hash (L2H) methods [15, 26]. L2H methods can achieve comparable or better accuracy with shorter hash codes when compared to data-independent methods. In real applications, L2H methods have become more popular than data-independent methods.

Existing L2H methods can be further divided into two categories: unsupervised hashing and supervised hashing refer to a comprehensive survey [28].

Unsupervised hashing methods use the unlabeled training data only to learn hash functions and encode the input data points to binary codes. Typical unsupervised hashing methods include reconstruction error minimization [29, 30], graph based hashing [3, 31], isotropic hashing (IsoHash) [9], discrete graph hashing (DGH) [32], scalable graph hashing (SGH) [33], and iterative quantization (ITQ) [8].

Supervised hashing utilizes information, such as class labels, to learn compact hash codes. Representative supervised hashing methods include binary reconstruction embedding (BRE) [7], Minimal Loss Hashing (MLH) [34], Supervised Hashing with Kernels (KSH) [4], two-step hashing (TSH) [35], fast supervised hashing (FastH) [12], and latent factor hashing (LFH) [36]. In the pipelines of these methods, images are first represented by handcrafted visual descriptor feature vectors (e.g., GIST [37], HOG [38]), followed by separate projection and quantization steps to encode vectors into binary hash codes. However, such handcrafted feature represents the low level information of a picture whose construction process is independent of the hash function learning process, and the resulting features might not be optimally compatible with hash codes.

Recently, as the deep learning has shown its effective image representation power on high level semantic information in a picture, then, a lot of feature learning based deep hashing methods have recently been proposed and have shown their better performance than traditional hashing methods with handcrafted features, such as convolutional neural network hashing (CNNH) [39], network in network hashing (NINH) [40], deep hashing network (DHN) [41], and deep pairwise supervised hashing (DPSH) [15]. CNNH is proposed by Xia et al. The CNNH method first learns the hash codes from the pairwise labels and then tries to learn the hash function and feature representation from image pixels based on hash codes. Lai et al. improved the two-stage CNNH by proposing NINH. NINH uses a triplet ranking loss to preserve relative similarities and the hash codes of images are encoded by dividing and encoding modules. Moreover, this method is a simultaneous feature learning and hash coding deep network so that image representations and hash codes can improve each other in the joint learning process. DHN further improves NINH by controlling the quantization error in a principled way and devising a more principled pairwise cross entropy loss to link the pairwise Hamming distances with the pairwise similarity labels, while DPSH learns hash codes by learning features and hash codes simultaneously with pairwise labels. Due to the fact that different components in deep pairwise supervised hashing (DPSH) can give feedback to each other, DPSH outperforms other methods in image retrieval application as far as we know.

In this work, we further improve the retrieval accuracy by two steps: (1) training two different architecture's deep hashing subnetworks and (2) fusing the hash codes generated by the two subnetworks to unify images so that the merged codes can represent more semantic information and support each other. These two important stages constitute the DHFI approach.

3. The Proposed Approach

In this section, we describe our method in detail. We first train two different architecture's deep hashing subnetworks. Then, we perform each image through the subnetworks to generate binary hash codes and fuse the hash codes generated by the same image together. For the first step discussed in Section 3.1, we follow the simultaneous feature learning and hash code learning method of [15]. The major novelty of our method is training two deep hashing subnetworks and fusing the hash codes generated by the two subnetworks together to index images.

3.1. Subnetwork Training. We have n images (feature points) $\chi = \{x_1, x_2, \ldots, x_n\}$ and the training set of supervised hashing

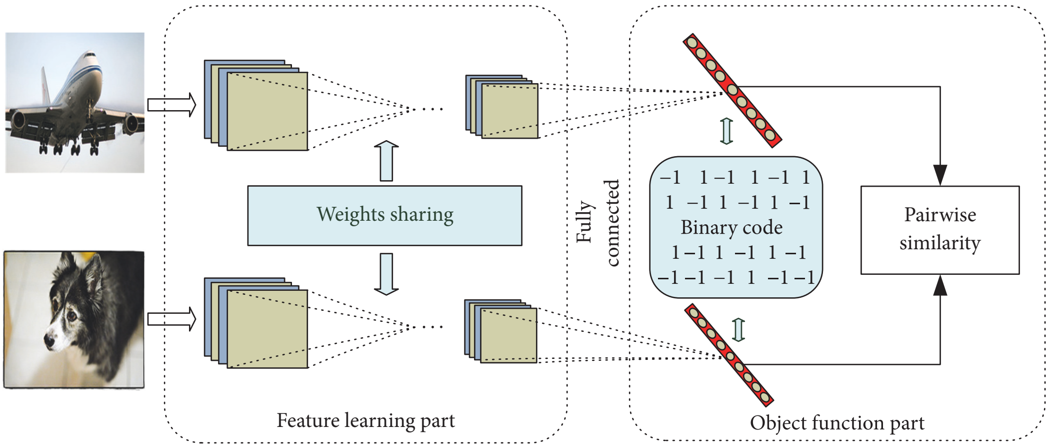

Figure 1: The end-to-end deep hash network learning architecture.

with pairwise labels also contains a set of pairwise labels $S = \{s_{ij}\}$ with $s_{ij} \in \{0, 1\}$, where $s_{ij} = 1$ means that x_i and x_j are similar and $s_{ij} = 0$ means that x_i and x_j are dissimilar. Here, the pairwise labels typically refer to semantic labels provided with manual efforts.

The goal of supervised hashing with pairwise labels is to learn a binary code $b_i \in \{-1, 1\}^c$ for each point x_i, where c is the code length. The binary code $B = \{b_i\}_{i=1}^n$ should preserve the similarity in S. More specifically, if $s_{ij} = 1$, then binary codes b_i and b_j should have a low Hamming distance; if $s_{ij} = 0$, the binary codes b_i and b_j should have a high Hamming distance. In general, we can write the binary code as $b_i = h(x_i) = [h_1(x_i), h_1(x_i), \ldots, h_c(x_i)]^T$, where $h(x_i)$ is the hash function to learn. For the subnetworks training step, we use the model and learning method called deep pairwise supervised hashing (DPSH) from Li et al. The model is an end-to-end deep learning method, which consists of two parts: the feature learning part and the objective function part.

The feature learning part has seven layers, which are the same as those of fast architecture's Convolutional Neural Network (CNN-F) in [42, 43].

As for the objective function part, given the binary codes $B = \{b_i\}_{i=1}^n$ for all the images, the likelihood of pairwise labels $S = \{s_{ij}\}$ can be defined as that of LFH [36]:

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

where $\Omega_{ij} = (1/2) b_i^T b_j$ and $\sigma(\Omega_{ij}) = 1/(1 + e^{-\Omega_{ij}})$. Please note that $b_i \in \{-1, 1\}^c$. When taking the negative log-likelihood of the observed pairwise labels in S, the problem becomes an optimization problem:

$$\begin{aligned} \min_{B} \quad J_1 &= -\log p(S \mid B) = -\sum_{S_{ij} \in S} \log p\left(s_{ij} \mid B\right) \\ &= -\sum_{S_{ij} \in S} \left(s_{ij}\Omega_{ij} - \log\left(1 - e^{\Omega_{ij}}\right)\right). \end{aligned} \tag{2}$$

The optimization problem above can make the Hamming distance between two similar images (points) as small as possible and make the Hamming distance between two dissimilar images (points) as large as possible simultaneously. While the problem is a discrete optimization problem, which is difficult to solve, we follow the strategy designed by Li et al., to reformulate the problem as follows:

$$\begin{aligned} \min_{B,u} \quad & J_2 = -\sum_{s_{ij} \in S} \left(s_{ij}\theta_{ij} - \log\left(1 + e^{\theta_{ij}}\right)\right) \\ \text{s.t.} \quad & u_i = b_i, \quad \forall i = 1, 2, \ldots, n, \\ & u_i \in R^{c \times 1}, \quad \forall i = 1, 2, \ldots, n, \\ & b_i \in \{-1, 1\}^c, \quad \forall i = 1, 2, \ldots, n, \end{aligned} \tag{3}$$

where $\theta_{ij} = (1/2) u_i^T u_j$ and $U = \{u_i\}_{i=1}^n$. And the problem can be continually optimized by moving the equality constraints in the equation to the regularization terms.

$$\begin{aligned} \min_{B,u} \quad & J_3 \\ & = -\sum_{s_{ij} \in S} \left(s_{ij}\theta_{ij} - \log\left(1 + e^{\theta_{ij}}\right)\right) + \eta \sum_{i=1}^{n} \left\| b_i - u_i \right\|_2^2, \end{aligned} \tag{4}$$

where η is the regularization term.

A fully connected hash layer is designed between the two parts to integrate them to a whole framework. The framework is shown in Figure 1. Please note that two images are input into the framework at each training time, and the loss function is based on pair labels of images.

For the hash layer, we set

$$u_i = W^T \phi\left(x_i; \theta\right) + \nu, \tag{5}$$

where θ denotes all the parameters of the first seven layers in the feature learning part, $\phi(x_i; \theta)$ denotes the output of the

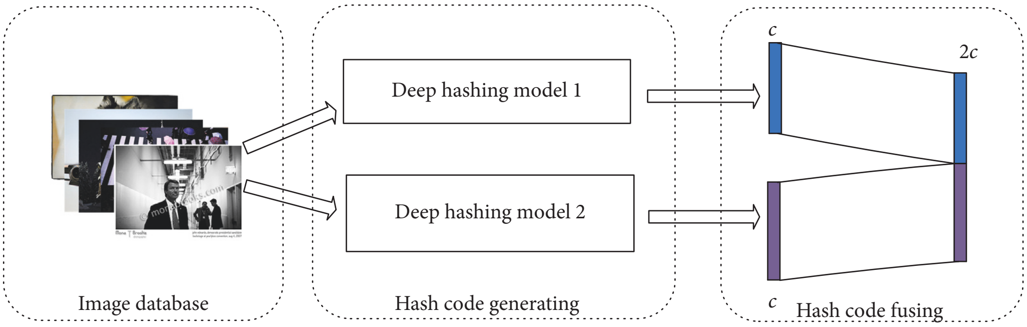

Figure 2: The deep hashing based fusing index learning architecture.

seventh layer associated with image (point) x_i, $W \in \mathbb{R}^{4096\times c}$ denotes a weight matrix, and $v \in \mathbb{R}^{c\times 1}$ is a bias vector.

After connecting the feature learning part and the objective function together, the problem of learning becomes

$$\begin{aligned}\min_{B,U} \quad & J_3 - \sum_{s_{ij}\in S}\left(s_{ij}\theta_{ij} - \log\left(1+e^{\theta_{ij}}\right)\right) \\ & + \eta\sum_{i=1}^{n}\left\|b_i - \left(W^T\phi\left(x_i;\theta\right)+v\right)\right\|_2^2 .\end{aligned} \tag{6}$$

In each subnetwork, following Li et al., we also adopt the minibatch based strategy and alternating method to learn the parameters containing W, v, θ, and B. We sample a minibatch of images (points) from the whole training set and each subnetwork learns based on these sampled images (points). Then, we optimize one parameter with other parameters fixed. b_i can be directly optimized as follows:

$$b_i = \operatorname{sgn}\left(u_i\right) = \operatorname{sgn}\left(W^T\phi\left(x_i;\theta\right)+v\right). \tag{7}$$

We use the back-propagation method to learn other parameters W, v, and θ. Specially, we can compute the derivatives of the loss function with respect of u_i as follows:

$$\begin{aligned}\frac{\partial J}{\partial u_i} = {} & \frac{1}{2}\sum_{j:s_{ij}\in S}\left(a_{ij}-s_{ij}\right)u_j + \frac{1}{2} \\ & \cdot \sum_{j:s_{ji}\in S}\left(a_{ji}-s_{ji}\right)u_j + 2\eta\left(u_i - u_j\right),\end{aligned} \tag{8}$$

where $a_{ij} = \sigma((1/2)u_i^T u_j)$. Then, we can update the parameters W, v, and θ by back-propagation:

$$\begin{aligned}\frac{\partial J}{\partial W} &= \phi\left(x_i;\theta\right)\left(\frac{\partial J}{\partial u_i}\right)^T, \\ \frac{\partial J}{\partial V} &= \frac{\partial J}{\partial u_i}, \\ \frac{\partial J}{\phi\left(x_i;\theta\right)} &= W\frac{\partial J}{\partial u_i}.\end{aligned} \tag{9}$$

In our method, we trained two deep hashing subnetworks by utilizing the learning algorithm in [15]. More specially, the CNN-F and the Caffe-alex [18] pretrained networks are separately used in the feature learning part of the different subnetworks.

3.2. Hash Codes Generating and Fusing

3.2. Hash Codes Generating and Fusing. After we have successfully completed the training of subnetworks, we can only get the hash codes for images in the training data. We still have to predict the hash codes for other images which did not appear in the training set. For any image $x_q \in X$, we let it through each subnetwork to predict its hash codes just by forward propagation:

$$b_q = h\left(x_q\right) = \operatorname{sgn}\left(W^T\phi\left(x_q;\theta\right)+v\right). \tag{10}$$

Thus we can get two hash codes related to x_q. We concatenate the two different hash codes learned from the two different subnetworks together in a vector way and use the concatenated code as the latest hash code of x_q. The hash code generating and fusing process is shown in Figure 2.

4. Experiments

4.1. Experimental Settings. All our experiments for DHFI are completed with MatConvNet [43] on a NVIDIA K40 GPU server.

In this section, we conduct extensive evaluations of the proposed method on two widely used benchmark datasets with different kinds of images: CIFAR-10 and NUS-WIDE. (1) The CIFAR-10 [44] dataset consists of 60K 32 × 32 color tiny images which are categorized into 10 classes (6K tiny images per class). It is a single-label dataset in which each image belongs to one of the 10 classes. (2) The NUS-WIDE dataset [45, 46] has nearly 270K images collected from the web. It is a multilabel dataset in which each image is annotated with one or multiple class labels in 81 semantic concepts. Following [15, 40], we only use the images from the 21 most frequent classes. For these classes, the number of images in each class is at least 5K.

The experimental protocols in [15] are also employed in our experiments. In CIFAR-10, 1000 images (100 images per class) are randomly selected as the query set. For the

Table 1: Accuracy in terms of MAP compared to two different deep DPSH models.

Method	CIFAR-10				NUS-WIDE			
	24 bits	32 bits	48 bits	64 bits	24 bits	32 bits	48 bits	64 bits
DPSH1	0.727	0.744	0.757	0.768	0.822	0.838	0.845	0.850
DPSH2	0.686	0.714	0.745	0.736	0.828	0.838	0.846	0.849
DHFI	**0.750**	**0.768**	**0.774**	**0.788**	**0.836**	**0.854**	**0.860**	**0.864**

unsupervised methods, we use the rest images as the training set. For the supervised methods, we randomly select 5000 images (500 images per class) from the rest of the images as the training set. The pairwise label set S is constructed based on the image class labels, where two images will be considered to be similar if they share the same class label.

In NUS-WIDE, 2100 query images from 21 most frequent labels (100 images per class) are randomly sampled as the query set by following the strategy used in [15, 39, 40]. For the supervised methods, we randomly select 500 images per class from the rest images as the training set. The pairwise label set S is constructed based on the image class labels. It means that two images will be considered to be similar if they share at least one common label.

Following [15], we compare our method to several state-of-the-art hashing methods, including SH [31], ITQ [8], SPLH [47], KSH [4], FastH [12], LFH [36], SDH [13], DPSH [15], CNNH [39], DHN [41], DSH [5], and NINH [40]. Note that SH and ITQ are unsupervised hashing methods and the other methods are supervised hashing methods. DPSH, CNNH, DHN, and DSH are four deep hashing methods with pairwise labels, while NINH is a triplet-based method. Beyond this, we also evaluate the nondeep hashing methods with deep features extracted by the CNN-F.

For hashing methods which use handcrafted features, we represent each image in CIFAR-10 by a 512-dimensional GIST vector. And we represent each image in NUS-WIDE by a 1134-dimensional low level feature vector, including 64-D color histogram, 144-D color correlogram, 73-D edge direction histogram, 128-D wavelet texture, 225-D block-wise color moments, and 500-D SIFT features.

For deep hashing methods, we first resize all images to 224 × 224 pixels and then directly use the raw image pixels as input and adopt the CNN-F network which has been pretrained on the ImageNet dataset to initialize the layers of feature learning part. Similar initialization strategy has also been adopted by other deep hashing methods [48].

For our method, we learn the hash codes separately from different architecture's pretrained networks; we use the fast architecture's Convolutional Neural Network (CNN-F) and Caffe-alex network to initialize the parameters.

4.2. Results and Discussion. The mean average precision (MAP) is often used to measure the accuracy in large-scale image retrieval applications. As most existing hashing methods, the MAP is used to measure the accuracy of the proposed method. For fair comparison, all of the methods use identical training and test sets. In this paper, the MAP value is calculated based on the top 5000 returned neighbors for NUS-WIDE dataset. The best MAP for each category in the tables are shown in boldface.

Firstly, to verify the effectiveness of deep binary hash code fusing, we compare our method to two different architecture's deep pairwise supervised hashing models; one uses the CNN-F pretrained model in the feature learning part and the other uses the Caffe-alex pretrained model in the feature learning part. The MAP results are listed in Table 1. Please note that DPSH1 uses CNN-F and DPSH2 uses Caffe-alex pretrained model. By comparing DHFI to DPSH1 and DPSH2, we find that DHFI can dramatically outperform both of them. It means that the integrated hash codes learned from different architecture's deep hashing subnetworks can get a better solution than hash codes generated from independent subnetwork.

Secondly, the MAP results of all methods are listed in Tables 2 and 3. Please note that, in Table 2, DPSH, DSH, DHN, NINH, and CNNH are deep hashing methods, and all the other methods are nondeep methods with handcrafted features. The results of NINH, CNNH, KSH, and ITQ are from [15, 39, 40], the results of DPSH are from [15], the results of DSH are from [5], and the results of DHN are from [41]. Please note that the above experimental settings and evaluation metrics are exactly the same as that in [15, 39, 40]. Hence, the comparison is reasonable. We can find that our method dramatically outperforms other baselines, including unsupervised methods, supervised methods with handcrafted features, and deep hashing methods with feature learning.

To further verify the effectiveness of the deep binary hash code fusing, we compare DHFI to other nondeep methods with deep features extracted by the fast architecture's Convolutional Neural Network (CNN-F). The results are shown in Table 3, where the notation of "+CNN" denotes that the methods use deep features as input. We can find that our method outperforms all the other nondeep baselines with deep features.

5. Conclusion

In this paper, we proposed a "two-stage" deep hashing based fusing index method for image retrieval. In the proposed method, we train two different architecture's deep hashing networks at first and then merge the hash codes generated from separate networks together to unify an image. Due to the fact that hash codes are learned from different networks and they may provide different information and supplement each other, the proposed method can learn better codes than other hashing methods. Experiments on real datasets show

Table 2: Accuracy in terms of MAP compared to hashing methods.

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
SPLH	0.171	0.173	0.178	0.184	0.568	0.589	0.597	0.601
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	0.476	0.472	0.489	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
DSH	0.616	0.651	—	0.661	0.548	0.551	—	0.562
DPSH	**0.713**	0.727	0.744	0.757	0.747	0.822	0.838	0.845
DHFI	0.613	**0.750**	**0.768**	**0.774**	**0.807**	**0.836**	**0.854**	**0.860**

Table 3: Accuracy in terms of MAP compared to nondeep methods with deep features.

Method	CIFAR-10				NUS-WIDE			
	12 bits	24 bits	32 bits	48 bits	12 bits	24 bits	32 bits	48 bits
SH + CNN	0.183	0.164	0.161	0.161	0.621	0.616	0.615	0.612
ITQ + CNN	0.237	0.246	0.255	0.261	0.719	0.739	0.747	0.756
SPLH + CNN	0.299	0.330	0.335	0.330	0.753	0.775	0.783	0.786
LFH + CNN	0.208	0.242	0.266	0.339	0.695	0.734	0.739	0.759
KSH + CNN	0.488	0.539	0.548	0.563	0.768	0.786	0.790	0.799
SDH + CNN	0.478	0.557	0.584	0.592	0.780	0.804	0.815	0.824
FastH + CNN	0.553	0.607	0.619	0.636	0.779	0.807	0.816	0.825
DHFI	**0.613**	**0.750**	**0.768**	**0.774**	**0.807**	**0.836**	**0.854**	**0.860**

that our method has superior performance over state-of-the-art image retrieval applications.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by Natural Science Foundation of China [Grant nos. 61370113, 61572004, 61650201, and 91546111], the Beijing Municipal Natural Science Foundation [Grant nos. 4152005 and 4162058], the Key Project of Beijing Municipal Education Commission [Grant no. KZ201610005009]; the Science and Technology Program of Tianjin [Grant no. 15YFXQGX0050], and the Science and Technology Planning Project of Qinghai Province [Grant no. 2016-ZJ-Y04].

References

[1] A. Andoni and P. Indyk, "Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions," *Foundations of Computer Science Annual Symposium on 51.1*, pp. 117–122, 2008.

[2] J. Wang, S. Kumar, and S. F. Chang, "Semi-supervised hashing for large-scale search," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, vol. 34, no. 12, pp. 2393–2406, 2012.

[3] H. Jégou, M. Douze, and C. Schmid, "Product quantization for nearest neighbor search," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 1, pp. 117–128, 2011.

[4] W. Liu, J. Wang, R. Ji, Y.-G. Jiang, and S.-F. Chang, "Supervised hashing with kernels," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '12)*, pp. 2074–2081, Providence, RI, USA, June 2012.

[5] H. Liu, R. Wang, S. Shan, and X. Chen, "Deep supervised hashing for fast image retrieval," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR, 1b)*, pp. 2064–2072, 2016.

[6] K. Zhan, J. Guan, Y. Yang, and Q. Wu, "Unsupervised discriminative hashing," *Journal of Visual Communication & Image Representation*, vol. 40, pp. 847–851, 2016.

[7] B. Kulis and T. Darrell, "Learning to hash with binary reconstructive embeddings," in *NIPS*, pp. 1042–1050.

[8] Y. Gong et al., "Iterative quantization: a Procrustean approach to learning binary codes for large-scale image retrieval," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR '11)*, pp. 817–824, 2011.

[9] W. Kong and W. J. Li, "Isotropic hashing," *Advances in Neural Information Processing Systems*, vol. 2, pp. 1646–1654, 2012.

[10] M. Rastegari, J. Choi, S. Fakhraei, D. Hal, and L. Davis, "Predictable dual-view hashing," in *Proceedings of 30th International Conference on Machine Learning*, pp. 1328–1336, 2013.

[11] K. He, F. Wen, and J. Sun, "*K*-means hashing: An affinity-preserving quantization method for learning binary compact codes," in *Proceedings of 26th IEEE Conference on Computer Vision and Pattern Recognition, CVPR'*, pp. 2938–2945, 2013.

[12] G. Lin, C. Shen, Q. Shi, A. Van Den Hengel, and D. Suter, "Fast supervised hashing with decision trees for high-dimensional data," in *Proceedings of 27th IEEE Conference on Computer Vision and Pattern Recognition, CVPR'*, pp. 1971–1978, usa, 2014.

[13] F. Shen, C. Shen, W. Liu, and H. T. Shen, "Supervised discrete hashing," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, CVPR'*, pp. 37–45, June 2015.

[14] K. Wang-Cheng, L. Wu-Jun, and Z. Zhi-Hua, "Column sampling based discrete supervised hashing," in *AAAI*, 2016.

[15] L. Wujun, W. Sheng, and K. Wangcheng, "Feature learning based deep supervised hashing with pairwise labels," in *IJCAI*, 2016.

[16] R. Das, S. Thepade, S. Bhattacharya, and S. Ghosh, "Retrieval Architecture with classified query for content based image recognition," *Applied Computational Intelligence and Soft Computing*, vol. 2016, 2 pages, 2016.

[17] Y. Xu, F. Shen, X. Xu, L. Gao, Y. Wang, and X. Tan, "Large-scale image retrieval with supervised sparse hashing," *Neurocomputing*, vol. 229, pp. 45–53, 2017.

[18] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks.," in *Proceedings of International Conference on Neural Information Processing Systems Curran Associates Inc*, pp. 1097–1105, 2012.

[19] C. Szegedy, W. Liu, Y. Jia et al., "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '15)*, pp. 1–9, Boston, Mass, USA, June 2015.

[20] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: surpassing human-level performance on imagenet classification," in *Proceedings of the 15th IEEE International Conference on Computer Vision (ICCV '15)*, pp. 1026–1034, IEEE, Santiago, Chile, December 2015.

[21] C. Szegedy, A. Toshev, and D. Erhan, "Deep Neural Networks for object detection," *Advances in Neural Information Processing Systems*, vol. 26, pp. 2553–2561, 2013.

[22] Y. Sun, X. Wang, and X. Tang, *Deep Learning Face Representation by Joint Identification-Verification*, vol. 27, 2015.

[23] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '15)*, pp. 3431–3440, IEEE, Boston, Mass, USA, June 2015.

[24] Y. Liu, Y. Pan, H. Lai, C. Liu, and J. Yin, "Margin-based two-stage supervised hashing for image retrieval," *Neurocomputing*, vol. 214, pp. 894–901, 2016.

[25] D. Xie, L. Zhang, and L. Bai, "Deep Learning in Visual Computing and Signal Processing," *Applied Computational Intelligence and Soft Computing*, vol. 2017, pp. 1–13, 2017.

[26] J. Deng, N. Ding, Y. Jia et al., "Large-scale object classification using label relation graphs," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 8689, pp. 48–64, 2014.

[27] Z. Songhao et al., "Integration of semantic and visual hashing for image retrieval," *Journal of Visual Communication & Image Representation*, 2016.

[28] W. Kong and W. J. Li, "Isotropic hashing," *Advances in Neural Information Processing Systems*, vol. 2, no. 2012, pp. 1646–1654, 2012.

[29] W. Jingdong et al., "Hashing for similarity search: a survey," *Computer Science*, 2014.

[30] R. Salakhutdinov and G. Hinton, "Learning a nonlinear embedding by preserving class neighbourhood structure," *Journal of Machine Learning Research*, vol. 2, pp. 412–419, 2007.

[31] Y. Weiss, A. Torralba, and R. Fergus, "Spectral hashing," in *Proceedings of Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December DBLP*, pp. 1753–1760, 2008.

[32] L. Wei, "Discrete graph hashing," in *Proceedings of International Conference on Neural Information Processing Systems MIT Press*, pp. 3419–3427, 2014.

[33] Q. Jiang Yuan and W. J. Li, "Scalable graph hashing with feature transformation," in *Inproceeding of International Conference on Artificial Intelligence AAAI Press*, pp. 2248–2254, 2015.

[34] N. Mohammad Emtiyaz and D. J. Fleet, "Minimal loss hashing for compact binary codes," in *Proceedings of nternational Conference on Machine Learning*, pp. 353–360, Bellevue, Washington, USA, 2011.

[35] G. Lin, C. Shen, D. Suter, and A. V. D. Hengel, "A general two-step approach to learning-based hashing," in *Proceedings of 2013 14th IEEE International Conference on Computer Vision, ICCV 2013*, pp. 2552–2559, aus, December 2013.

[36] P. Zhang, W. Zhang, W.-J. Li, and M. Guo, "Supervised hashing with latent factor models," in *Proceedings of 37th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2014*, pp. 173–182, aus, July 2014.

[37] A. Oliva and A. Torralba, "Modeling the shape of the scene: a holistic representation of the spatial envelope," *International Journal of Computer Vision*, vol. 42, no. 3, pp. 145–175, 2001.

[38] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '05)*, vol. 1, pp. 886–893, June 2005.

[39] R. Xia, Y. Pan, H. Lai, C. Liu, and S. Yan, "Supervised hashing for image retrieval via image representation learning," *AAAI Conference on Artificial Intelligence*, pp. 2156–2162, 2014.

[40] H. Lai, Y. Pan, Y. Liu, and S. Yan, "Simultaneous feature learning and hash coding with deep neural networks," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, CVPR'*, pp. 3270–3278, June 2015.

[41] Z. Han, "Deep hashing network for efficient similarity retrieval," in *Proceedings of Thirtieth AAAI Conference on Artificial Intelligence AAAI Press*, pp. 2415–2421, 2016.

[42] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman, "Return of the devil in the details: delving deep into convolutional nets," *Computer Science*, 2014.

[43] A. Vedaldi and K. Lenc, "MatConvNet: convolutional neural networks for matlab," in *Proceedings of the 23rd ACM International Conference on Multimedia*, pp. 689–692, Brisbane, Australia, October 2015.

[44] A. Krizhevsky, "Learning multiple layers of features from tiny images," 2012, Learning Multiple Layers of Features from Tiny Images.

[45] T.-S. Chua, J. Tang, R. Hong, H. Li, Z. Luo, and Y. Zheng, "NUS-WIDE: A real-world web image database from national university of singapore," in *Proceedings of ACM International Conference on Image and Video Retrieval, CIVR 2009*, pp. 368–375, grc, July 2009.

[46] X. Zhao, X. Li, and Z. Zhang, “Multimedia retrieval via deep learning to rank,” *IEEE Signal Processing Letters*, vol. 22, no. 9, pp. 1487–1491, 2015.

[47] J. Wang, S. Kumar, and S. F. Chang, “Sequential Projection Learning for Hashing with Compact Codes,” in *Inproceeding of International Conference on Machine Learning DBLP*, pp. 1127–1134, 2010.

[48] F. Zhao, Y. Huang, L. Wang, and T. Tan, “Deep semantic ranking based hashing for multi-label image retrieval,” in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015*, pp. 1556–1564, June 2015.

Advances in Multimedia
The Scientific World Journal
International Journal of Distributed Sensor Networks
Journal of Industrial Engineering
Applied Computational Intelligence and Soft Computing
Advances in Fuzzy Systems
Modelling & Simulation in Engineering
Journal of Computer Networks and Communications
Advances in Artificial Intelligence
Advances in Computer Engineering
International Journal of Computer Games Technology
International Journal of Biomedical Imaging
Advances in Artificial Neural Systems
Advances in Software Engineering
Journal of Robotics
Advances in Human-Computer Interaction
Computational Intelligence and Neuroscience
International Journal of Reconfigurable Computing
Journal of Electrical and Computer Engineering