Multimed Tools Appl (2018) 77:10471–10484
DOI 10.1007/s11042-017-4489-0

Multiple hierarchical deep hashing for large scale image retrieval

Liangfu Cao[1] · Lianli Gao[1] · Jingkuan Song[1] · Fumin Shen[1] · Yuan Wang[1]

Received: 6 December 2016 / Revised: 19 January 2017 / Accepted: 6 February 2017 /
Published online: 27 February 2017

Abstract Learning-based hashing methods are becoming the mainstream for large scale visual search. They consist of two main components: hash codes learning for training data and hash functions learning for encoding new data points. The performance of a content-based image retrieval system crucially depends on the feature representation, and currently Convolutional Neural Networks (CNNs) has been proved effective for extracting high-level visual features for large scale image retrieval. In this paper, we propose a Multiple Hierarchical Deep Hashing (MHDH) approach for large scale image retrieval. Moreover, MHDH seeks to integrate multiple hierarchical non-linear transformations with hidden neural network layer for hashing code generation. The learned binary codes represent potential concepts that connect to class labels. In addition, extensive experiments on two popular datasets demonstrate the superiority of our MHDH over both supervised and unsupervised hashing methods.

Keywords Multimedia · Deep hashing · Large scale image retrieval · Convolutional neural networks

1 Introduction

With the huge surge in multimedia content over the Internet, as well as high speed Internet connectivity, most of the websites now contain large volume of images [33, 34, 46]. This triggers the development of hashing systems, where the goal is to build a hashing system, which enable the accuracy and efficiency of object recognition [9, 38, 39], image retrieval

✉ Lianli Gao
Lianli.gao@uestc.edu.cn

[1] University of Electronic Science and Technology of China, No.2006, Xiyuan Ave, West Hi-Tech Zone, 611731, Chengdu China

[6, 21, 47], image matching [7, 37], image tagging [48] and so on. Recently, research [3, 8, 10, 15, 19, 27, 32, 35, 41, 43, 49] has been done in computer vision community aiming to encode high-dimension image features into low-dimension short codes. These codes obtain efficiency in both searching and storage [31]. In fact, the basic idea of hashing is to establish hash functions to project high-dimensional feature vectors into a set of compact binary codes, thus similar data samples can be mapped into similar hash codes.

In general, existing hashing-based methods can be generally divided into two categories: *data-independent* and *data-dependent* [42]. For the first category, applying stochastic projections to establish randomized hash functions is mainly employed in the early exploration of hashing. One of the most popular method is Locality Sensitive Hashing (LSH) [5], and its derivations [20] is proposed by integrating with kernels or additional constraint. Theoretically, LSH-based methods are focusing on confirming long codes asymptotically preserving original metrics in Hamming spaces, thus to achieve the desired performance, longer codes are usually required. However, long code is not only able to reach low recall, but improves the Hamming codes storage as well. For the data-dependent approaches, they learn hash functions, which are able to generate short hash codes to conduct efficient large-scale data search. With short hash codes, fast approximate nearest neighbor (ANN) search can be achieved in a constant time complexity as long as neighbors of a data sample are well-preserved in the Hamming space. Moreover, in a huge database, short hash codes are extremely effective in saving storage. Many classic hash functions including PCA iterative quantization (PCA-ITQ)) [10], K-means hashing (KMH) [12], spectral hashing(SH) [43], binary reconstructive embedding (BRE) [19], minimal loss hashing (MLH) [29], and sequential projection learning hashing (SPLH) [41] etc., belong to this category. Nevertheless, nonlinear relationship among data samples cannot be well addressed in most of these hashing methods, thus [3, 11] propose approaches to integrate kernels to deal this issue. However, the scalability problem is still inevitable.

In this work, to take the full advantage of data relationships to learn effective short hash codes, we develop a novel deep learning framework: Multiple Hierarchical Deep Hashing (MHDH). The fundamental idea is shown in Fig. 1. Unlike most existing hashing methods, which aim to look for a simple linear projection and then project each data point into a binary vector, we present a novel deep neural network to explore multiple hierarchical nonlinear projections to make a full use of data relationships. This neural network is integrated

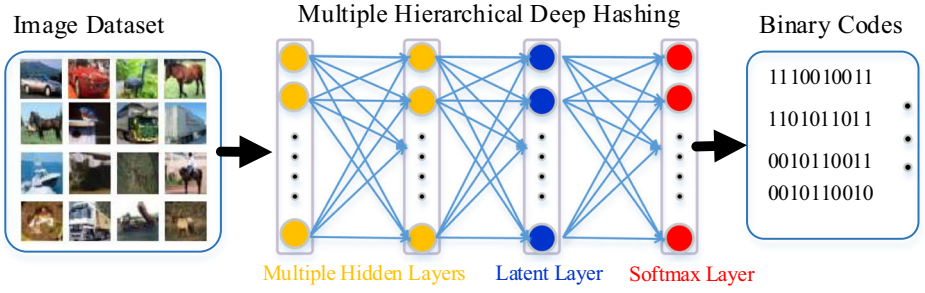


Fig. 1 The overview of MHDH, which aims to learn efficient short hash code to support large-scale image retrieval. This network consists of three types of layers: 1) conv layers (marked with *yellow*) that seek to project image to a rich description with multiple hierarchical nonlinear projections; 2) A latent layer or hash layer (*blue*) is used to further encode the previous generated description, and 3) a softmax layer (the final *red layer*)

with quantization layers to obtain effective binary codes to enhance efficient large-scale image search. Given labels of data samples, hash codes can be obtained by encoding the output of the last latent layer of the neural network. The output represents potential concepts and those concepts are connected to class labels. Comparing with conventional hash approaches, our method learns binary hash codes in a manner of point-wised way and is scalable for large-scale datasets. Abundant experimental results conducted on two popular datasets demonstrate the superiority and extensibility of our MHDH over both supervised and unsupervised hashing methods.

2 Related work

2.1 Deep learning

There is an increasing interest in deep learning for solving large-scale vision problems. Deep learning derived from machine learning, including a set of algorithms that model high-level abstractions from data by using multiple layers with complex structures or multiple non-linear transformations [1, 16, 22, 24]. In recent years, various deep learning architectures such as deep neural networks [14], convolutional deep neural networks [16, 28] and recurrent neural networks [44] have been widely applied to computer vision fields [2] and natural language processing [4], where they have been fully demonstrated the superiority on various visual application.

2.2 Learning-based hashing

There are numerous learning-based hashing methods being proposed, and they can be divided into three categories: unsupervised, semi-supervised and supervised. For unsupervised approach, they do not need any label or class information about the training data set. To date, projection and quantization are widely used for generate hash function. The classic unsupervised hash method PCA Hashing (PCAH) [40] and its variants use quantization and prove that quantization performs better than random projection. Later researchers find out that orthogonal projections may benefit to quantify. This is because data variance decreases rapidly. Therefore, Weiss et al. [43] propose a spectral hashing (SH), which learns short binary codes through quantization with nonlinear functions along the data PCA direction. In addition, ITQ [10] is proposed by simultaneously maximizing the variance of each hash code and minimizing the quantization loss to approximate the Euclidean distance in the Hamming distance. KMH [12] is focusing on minimizing the hamming distance loss between the quantized centers and the cluster centers. Compared with unsupervised hashing method, supervised and semi-supervised methods make use of full or partial label information to facilitate the generation of better hash codes. For example, Semi-Supervised Hashing (SSH) [41] is proposed by maximally reducing the empirical loss on the labeled training data and maximizing variances of the labeled and unlabeled training sets. Kulis and Darrell [19] introduces a binary re-constructive embedding (BRE) method, which tries to minimize the reconstruction error between the original Euclidean distance and the trained hamming distance. Norouzi and Fleet [29] illustrates a minimal loss hashing (MLH) method by minimizing the loss between the trained Hamming distance and quantization error among similar data samples. Strecha et al. [37] develops a LDA hashing through maximizing the within-class variations and minimizing the between-class variations of binary codes.

2.3 Deep hashing

Some deep hashing methods [25, 26, 36, 45, 50] have already been proposed, and these deep hashing methods have achieved a great improvement in similar data retrieval. The basic idea of conventional deep hashing [45] is to exploit CNN networks to extract feature vectors, which contains more representative information than traditional hand-crafted visual descriptors (e.g., GIST and HOG). On the basis of existing convolutional deep neural networks, most of deep hashing methods transform CNN to their own neural networks. For example, Lin et al. [25] inserts a latent layer as a hash layer before the last fully-connected layer in the pretrained Alexnet [18]. However, this network is time-consuming when training phase since it contains multiple CNN. Liong et al. [26] proposed a new deep hash method which including supervised and unsupervised hash. In this framework, it presents three layers neural network, and last layer as hash layer, in addtion, for supervised manner, it generate pair-wised label matrix as supervised information. Different from above deep hash, we consider if class label information is available, we present a new novel deep framework for hashing, and our deep hash is not only easy to train, but also achieves a better perfromance.

3 Proposed method

In this section, we first present the preliminary concepts for image retrieval, and then propose our MHDH method.

3.1 Preliminary concepts

Let the training data represents as $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N] \in \mathbb{R}^{d\times N}$, which contains N data samples, where $\mathbf{x}_n \in \mathbb{R}^d (1 \leq n \leq N)$ is the n-th sample in $\mathbf{X}$. The purpose of hashing is to learn a series of hash functions $f : \mathbb{R}^d \rightarrow \{-1, 1\}$[1]. Suppose we have to learn K hashing functions, the learned functions map each $\mathbf{x}_n$ into a compact binary vector and represented as $\mathbf{b}_n = [\mathbf{b}_{n1}, \mathbf{b}_{n2}, \cdots, \mathbf{b}_{nK}] \in \{-1, 1\}^{1\times K}$. Specifically, the k-th hash bit $\mathbf{b}_{nk}$ of $\mathbf{x}_n$ is calculated as follows:

$$b_{nk} = f_k(x_n) = sgn(w_k^T \mathbf{x}_n) \tag{1}$$

where f_k is the k-th hashing function, $w_k \in \mathbb{R}^d$ is the projection vector in f_k, and $sgn(v)$ is a sign function, which returns 1 if $v > 0$ otherwise returns -1.

Let $\mathbf{W} = [w_1, w_2, \cdots, w_K] \in \mathbb{R}^{d\times K}$ be a projection matrix. The projection of $\mathbf{x}_n$ can be calculated as: $\mathbf{W}^T\mathbf{x}_n$, which is used to compute hash codes with a quantization:

$$b_n = sgn(\mathbf{W}^T\mathbf{x}_n) \tag{2}$$

3.2 Our MHDH

Figure 1 shows our proposed MHDH framework. In this work, we present a novel deep neural network by seeking multiple hierarchical nonlinear transformations to learn binary codes to facilitate large-scale image search. In our model, the outputs of the classification layer depend on a set of hidden attributes, where each attribute is *on* or *off*. The learned hash codes should guarantee that similar binary codes should have the same labels. Based

[1]During training process, we consider '0' bit as '-1' bit, then in the implementation of encoding and testing, we use '0' again.

on this assumption, we design a latent layer that represented as a fully connected layer. In our method, activation functions in the latent layer are hyperbolic tangent functions, thus the activations can approximate to $\{-1, 1\}$.

Next, given a data sample $\mathbf{x}_n$, we learn a binary code $\mathbf{b}_n$ by inputing the data into the network and going through multiple nonlinear transformations. First, we assume that there are $L+1$ layers in our deep neural network, and there are p^l units for the l-th layer. Given a data sample $\mathbf{x}_n \in \mathbb{R}^d$, the output of the first layer is: $\mathbf{h}_n^1 = \varphi(\mathbf{W}^1\mathbf{x}_n + \mathbf{b}^1) \in \mathbb{R}^{p^1}$, where $\mathbf{W}^1 \in \mathbb{R}^{p^1 \times d}$ is a projection matrix , $\mathbf{b}^1 \in \mathbb{R}^{p^1}$ is a bias term, and $\varphi(\cdot)$ is a nonlinear activation function. Then, the output of the first layer is regarded as the input of the second layer, and $\mathbf{h}_n^2 = \varphi(\mathbf{W}^2\mathbf{h}_n^1 + \mathbf{b}^2) \in \mathbb{R}^{p^2}$, where $\mathbf{W}^2 \in \mathbb{R}^{p^2 \times p^1}$ is a projection matrix and $\mathbf{b}^2 \in \mathbb{R}^{p^2}$ is a bias vector of the second layer. Similarly, the output of the l-th layer is: $\mathbf{h}_n^l = \varphi(\mathbf{W}^l\mathbf{h}_n^{l-1} + \mathbf{b}^l) \in \mathbb{R}^{p^l}$. Then, the final layer of the network is:

$$\mathbf{h}_n^L = softmax(\mathbf{W}^L\mathbf{h}_n^{L-1} + \mathbf{b}^L) \in \mathbb{R}^C \tag{3}$$

where the $softmax(\cdot)$ is a classification function, and C is the number of images categories. Next, we can get the binary codes though performing quantization at the latent layer which is the former layer before the last layer:

$$\mathbf{b}_n = sgn(\mathbf{h}_n^{L-1}) \tag{4}$$

Let $\mathbf{B} = [\mathbf{b}_1, \mathbf{b}_2, \cdots, \mathbf{b}_n] \in \{-1, 1\}^{K \times N}$ be the generated binary codes. Given an input vector $\mathbf{x}_i$ that is a member of class $\mathbf{c}$, a value of stochastic variable Y, can be written as:

$$P(Y = \mathbf{c}|\mathbf{x}_i, \mathbf{W}, b) = softmax_c(\mathbf{W}\mathbf{x}_i + b) \tag{5}$$

where $\mathbf{x}_i$ stands for the data items that belongs to the class $\mathbf{c}$. Next, we calculate the maximum probability y_{pred}, which ensures that when $\mathbf{x}_i$ belongs to c class, we get the maximum probability. Here, we define y_{pred} as below:

$$y_{pred} = argmax_c P(Y = \mathbf{c}|\mathbf{x}_i, \mathbf{W}, b) \tag{6}$$

Because the final layer is a softmax layer, it can be seen as a case of multi-class logistic regression. So, it is naturally for us to use the negative log-likelihood as our loss function, thus the aforementioned optimization problem to learn the parameters is defined as follows:

$$\begin{array}{ll} \ell(\theta = \{\mathbf{W}, b\}, \mathbb{D}) = & -\Sigma_i log(P(Y = \mathbf{c}|\mathbf{x}_i, \mathbf{W}, b)) \\ + & \frac{\lambda}{2}\Sigma_l(\|\mathbf{W}^l\|_F^2 + \|b^l\|_F^2) \end{array} \tag{7}$$

Where $\mathbb{D}$ is a training set, θ is a set of all parameters. The second term is a regularization term, which is designed for avoiding over-fitting. The λ is a parameter to balance the impact of the second term. For training process, we adopt the stochastic gradient descent (SGD) technique to learn parameters $\{\mathbf{W}^l, \mathbf{b}^l\}_{l=1}^L$ and solve this optimization problem.

4 Experiments

In this section, we evaluate the performance of MHDH and compare it with the state-of-the-art hash methods. We used two frequently-used datasets in our experiments.

4.1 Experimental settings

First, we introduce our experimental settings as follow:

CIFAR-10 Krizhevsky [17] is composed of 10 object classes and 6,000 images are included in each class. For each image, the size is 32 × 32. Following the same setting in [41], the dataset is divided into a test set and a training set. For test dataset, it contains 1,000 data samples and 100 sample for each class. In addition, these 1,000 samples are randomly selected. The rest 59,000 images are used as training data. Moreover, we extract 512-D GIST feature vectors [30] to represent images in the CIFAR-10 dataset. Some sample images from CIFAR-10 dataset are shown in Fig. 2.

MNISTS Lecun and Cortes [23] consists of 10 catagories of handwirtten digits. There are 70,000 images in this dataset and the size of each image is 28 × 28. Specifically, we randomly sample 1,000 data samples as test images, and the remaining 69,000 images are used as training data. Each image is represented as a 784-D gray-scale feature vector.

Evaluation metrics Three evaluation metrics are used to evaluate the performance of different methods as follow: 1) mean average precision (mAP), which is the mean of the average precision scores for each query image; 2) precision at N samples, it is the ratio of true similar images among top N retrieved images; 3) Hamming look-up result, it measures the precision over all the points whose Hamming distance compared with query hash codes is less than or equal to r, here $r = 2$ [26].

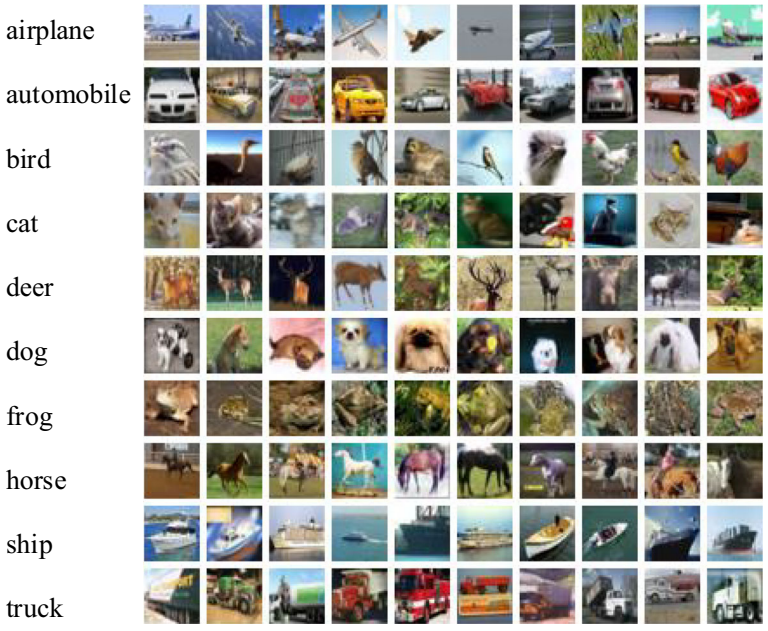


Fig. 2 Some sample images from the CIFAR-10 dataset. There are 10 object classes and each class shows 10 sample images

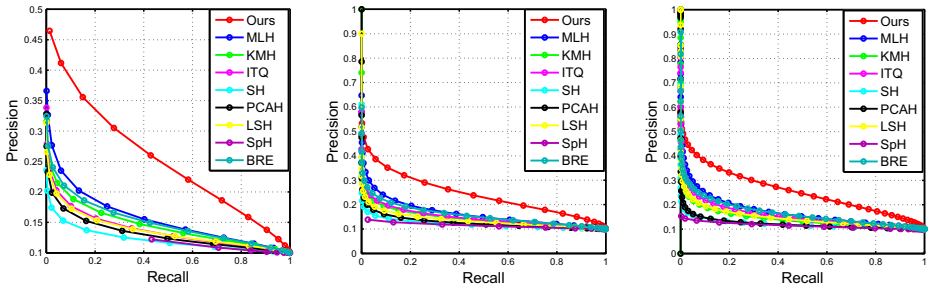
Table 1 Results on the CIFAR-10 dataset

Method	mAP (%)			P (%) @ $N = 1000$			P (%) @ $r = 2$	
	16	32	64	16	32	64	16	32
PCA-ITQ	15.67	16.20	16.64	22.46	25.30	27.09	22.60	14.99
KMH	13.59	13.93	14.46	20.28	21.97	22.80	22.08	5.72
SpH	13.98	14.58	15.38	20.13	22.33	25.19	20.96	12.50
SH	12.55	12.42	12.56	18.83	19.72	20.16	18.52	20.60
PCAH	12.91	12.60	12.10	18.89	19.35	18.73	21.29	2.68
LSH	12.55	13.76	15.07	16.21	19.10	22.25	16.73	7.07
DH	16.17	16.62	16.96	23.79	26.00	27.70	23.33	15.77
MLH	18.37	20.49	21.89	24.43	29.60	33.01	23.52	28.72
BRE	14.42	15.14	15.88	20.68	22.86	25.14	25.14	20.29
SDH	18.80	20.83	22.51	26.32	30.42	33.60	23.26	31.48
MHDH	38.43	40.47	42.41	37.91	39.71	41.63	34.74	40.68

There are unsupervised hashing methods' results showing on the top rows (from 3rd to 9th row), and supervised hashing methods' results are showing on the bottom table from 10th to 13th row. All the methods are evaluated by mAP and precision. In addition, the last two columns show the Hamming look-up results, here r is set as 2

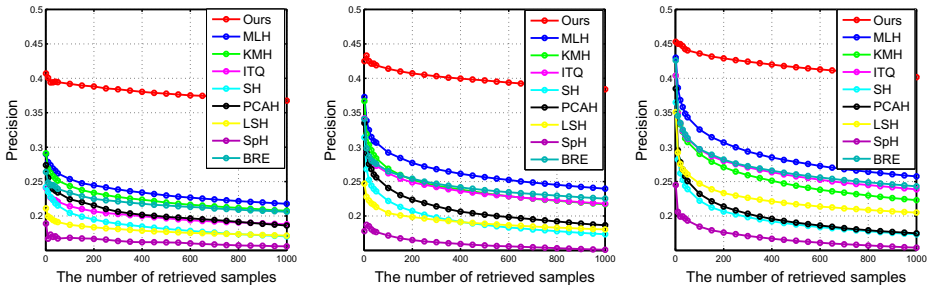
Baselines In our experiments, we compare our approach both unsupervised hashing approaches: PCA-ITQ [10], KMH [12], SpH [13], SH [43], PCAH [40], LSH [5], and DH [26]; and supervised approaches: MLH [29], BRE [19], SDH [26].

Network settings In the experiments, we designed our deep framework with 4 layers. Empirically, the neuron number of final output layer neuron is set as 10. Also, we set dimensions for these middle layers as $[60 \rightarrow 30 \rightarrow 16]$, $[80 \rightarrow 50 \rightarrow 32]$, and $[100 \rightarrow 80 \rightarrow 64]$ for the 16, 32 and 64 bits, respectively. The parameter λ was empirically set as 0.01. The weights of all network layers are randomly initialized, and we set the learning rate as 0.01 for our SGD optimization method . Furthermore, the hyperbolic tangent functions are used as activation functions in our framework.



(a) 16 bits (b) 32 bits (c) 64 bits

Fig. 3 The recall vs. precision curve on the CIFAR-10 dataset. The hash code length varies from 16, 32 to 64 bits



(a) 16 bits (b) 32 bits (c) 64 bits

Fig. 4 The retrieval precision results with variable number of retrieved samples on the CIFAR-10 dataset. The hash code length varies from 16, 32 to 64 bits

4.2 Experimental results

4.2.1 Results on CIFAR-10

Table 1 shows our results compared with other hashing methods on the CIFAR-10 dataset. From Table 1, we can have the following observations:

– Compared with unsupervised approaches, our MHDH performs best on three evaluation metrics. In terms of mAP, the best counterpart DH achieves 16.17 %, 16.62 % and 16.96 % with 16-bit, 32-bit and 64-bit, respectively, while our approach has 22.26 %, 23.85 % and 25.45 % increases on 16-bit, 32-bit and 64-bit, respectively. This is mainly because we used label information.

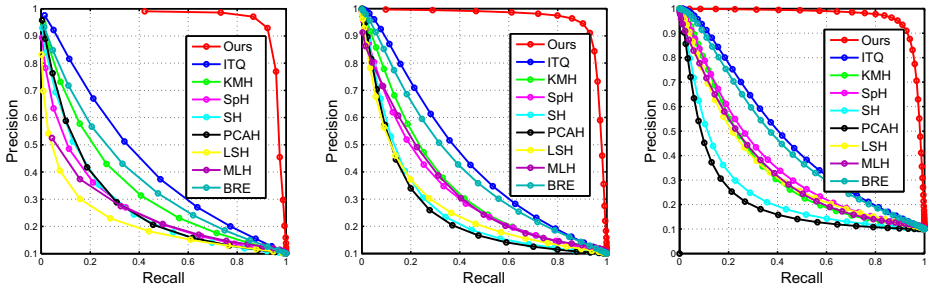
Fig. 5 On the CIFAR-10 dataset, top 6 retrieved images of 4 query images were returned by different hashing methods retrieval. The image at the first column are queries and we use 64 bit hash codes for retrieval

Table 2 Results of comparison on the MNIST dataset

Method	mAP (%)			P (%) @ $N = 1000$			P (%) @ $r = 2$	
	16	32	64	16	32	64	16	32
PCA-ITQ	41.18	43.82	45.37	66.39	74.04	77.42	65.73	73.14
KMH	32.12	33.29	35.78	60.43	67.19	72.65	61.88	68.85
SpH	25.81	30.77	34.75	49.48	61.27	69.85	51.71	64.26
SH	26.64	25.72	24.10	56.29	61.29	61.98	57.52	65.31
PCAH	27.33	24.85	21.47	56.56	59.99	57.97	36.36	65.54
LSH	20.88	25.83	31.71	37.77	50.16	61.73	25.10	55.61
DH	43.14	44.97	46.74	67.89	74.72	78.63	66.10	73.29
MLH	49.48	66.05	71.23	45.63	60.07	63.84	35.53	69.58
BRE	33.34	35.09	36.80	60.72	68.86	73.08	34.09	64.21
SDH	46.75	51.01	52.50	65.19	70.18	72.33	63.92	77.07
MHDH	94.44	95.10	95.13	94.40	95.04	95.06	93.56	94.36

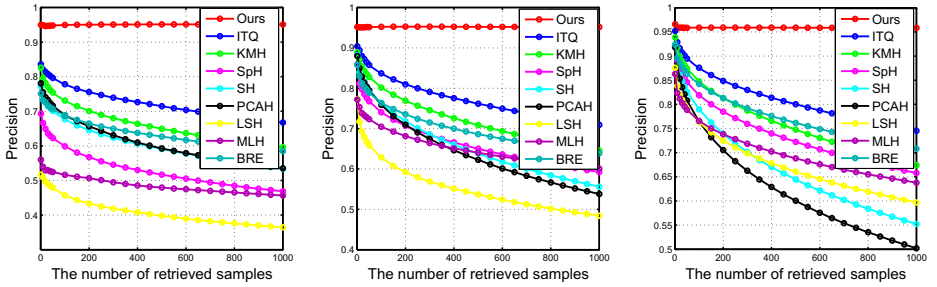
– Compared with the state-of-the-art supervised deep hash methods, MHDH also performs best on three evaluation metrics. Specifically, in terms of mAP, MHDH nearly doubles the best counterpart SDH with 19.36 % increase on 16-bit, 19.64 % on 32-bit, and 19.9 % on 64-bit. In terms of P@N=1,000, MHDH performs best with 37.91 % on 16-bit, 41.63 % on 64-bit. When P@ r=2, MHDH also gains the best performance.

The precision and recall obtained by the compared methods on the CIFAR-10 dataset with varying code length from 16 bits, 32 bits to 64 bits are show in Fig. 3. We can see from Fig.3 that with short code lengths, our method achieves the best results. In addition, as the increase of code length, the corresponding performance increases. When code length is 64 bit, our method achieves the best results. Figure 4 shows the comparison results in terms of precision on the CIFAR-10 dataset with varying code length with respect to the number of retrieved samples. From Fig. 4, we can see that our algorithm performs best. As the number of retrieved samples increases, our MHDH are relatively more stable than other methods. Figure 5 exhibits some query examples on the CIFAR-10 dataset. For each query, each method returns the top-6 query results by using 64-bit hash codes.



(a) 16 bits (b) 32 bits (c) 64 bits

Fig. 6 The recall vs. precision curve on the MNIST dataset. The hash code length varies from 16, 32 to 64 bits



(a) 16 bits (b) 32 bits (c) 64 bits

Fig. 7 The retrieval precision results with variable number of retrieved samples on the MNIST dataset. The hash code length varies from 16, 32 to 64 bits

4.2.2 Results on MNIST

In this subsection, we compare our method with the state-of-the-art methods on the MNIST dataset. The results are reported in Table 2. From Table 2, we can see that the proposed MHDH achieves the best mAP with 16 bit, 32b it and 64 bit. With 1,000 retrieved samples, MHDH still gains the hight precision scores in terms of 16 bit, 32 bit and 64 bit. When r=2, MHDH performs best 93.56 % with 16 bit and 94.36 with 32 bit, while the best compared method SDH only achieves 63.92 % with 16 bit and 77.07 % with 32 bit. This indicates the efficiency of our proposed MHDH. Figure 6 shows recall vs. precision curves generated by our method and the state-of-the-art methods. By observing the curves, we can see that MHDH significantly performs better than the compared methods. In addition, Fig. 7 demonstrates the precision with varying of retrieved samples. The number of retrieved number ranges from 0 to 1,000. For 16 bit, 32 bit and 64 bit, MHDH performs best with all kinds of settings.

5 Conclusion

In this paper, we have proposed a novel deep hashing method named MHDH for supporting large scale image retrieval. A latent layer is firstly integrated into the deep neural network. Next, a quantization layer is applied on to the latent layer to generate hash codes. The MHDH is focusing on obtaining good multiple hierarchical non-linear projections to facilitate hash code generation. Experimental results conducted on two popular datasets demonstrate the superiority of our MHDH over the state-of-the-art (i.e., both supervised and unsupervised) hashing methods.

Acknowledgments This work is supported by the Fundamental Research Funds for the Central Universities (Grant no. ZYGX2014J063), the National Natural Science Foundation of China (Grant no. 61502080) and the Priority Academic Program Development of Jiangsu Higher Education Institutions, and Jiangsu Collaborative Innovation Center on Atmospheric Environment and Equipment Technology.

References

1. Bengio Y (2009) Learning deep architectures for ai. Found Trends Mach Learn 2(1):1–55

2. Bradski GR (1998) Computer vision face tracking for use in a perceptual user interface. Intel Technol J Q2(Q2):214–219
3. Chang SF (2012) Supervised hashing with kernels. In: IEEE Conference on computer vision and pattern recognition, pp 2074–2081
4. Chowdhury GG (2003) Natural language processing. Ann Rev Inf Sci Technol 37(1):51–89
5. Datar M, Immorlica N, Indyk P, Mirrokni VS (2004) Locality-sensitive hashing scheme based on p-stable distributions. In: Twentieth Symposium on computational geometry, pp 253–262
6. Gao L, Song J, Liu X, Shao J, Liu J, Shao J (2015) Learning in high-dimensional multimedia data: the state of the art. Multimed Syst1–11
7. Gao L, Song J, Nie F, Yan Y (2015) Optimal graph learning with partial tags and multiple features for image and video annotation. In: CVPR, pp 4371–4379
8. Gao L, Song J, Zou F, Zhang D, Shao J (2015) Scalable multimedia retrieval by deep learning hashing with relative similarity learning. In: ACM multimedia, pp 903–906
9. Gao LL, Song J, Shao J, Zhu X, Shen HT (2015) Zero-shot image categorization by image correlation exploration. In: ICMR, pp 487–490
10. Gong Y, Lazebnik S (2011) Iterative quantization: a procrustean approach to learning binary codes. In: IEEE Conference on computer vision and pattern recognition, pp 817–824
11. He J, Liu W, Chang SF (2010) Scalable similarity search with optimized kernel hashing. In: ACM SIGKDD International conference on knowledge discovery and data mining. Washington, Dc, pp 1129–1138
12. He K, Wen F, Sun J (2013) K-means hashing: an affinity-preserving quantization method for learning binary compact codes. In: IEEE Conference on computer vision and pattern recognition, pp 2938–2945
13. Heo JP, Lee Y, He J, Chang SF, Yoon SE (2015) Spherical hashing: binary code embedding with hyperspheres. IEEE Trans Pattern Anal Mach Intelli 37(11):1–1
14. Hinton G, Deng L, Yu D, Dahl GE, Mohamed A, Jaitly N, Senior A, Vanhoucke V, Nguyen P, Sainath TN (2012) Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. IEEE Signal Process Mag 29(6):82–97
15. Hu G, Shao J, Gao L, Yang Y (2015) Exploring viewable angle information in georeferenced video search. In: ACM Multimedia, pp 839–842
16. Ji S, Yang M, Yu K (2013) 3d convolutional neural networks for human action recognition. IEEE Trans Pattern Anal Mach Intell 35(1):221–31
17. Krizhevsky A (2012) Learning multiple layers of features from tiny images
18. Krizhevsky A, Sutskever I, Hinton GE (2012) Imagenet classification with deep convolutional neural networks. Adv Neural Inf Process Syst 25(2):2012
19. Kulis B, Darrell T (2009) Learning to hash with binary reconstructive embeddings. In: Advances in neural information processing systems 22: conference on neural information processing systems 2009. Proceedings of a meeting held 7-10 December 2009. Vancouver, pp 1042–1050
20. Kulis B, Grauman K (2009) Kernelized locality-sensitive hashing for scalable image search 30(2):2130–2137
21. Kulis B, Jain P, Grauman K (2009) Fast similarity search for learned metrics. IEEE Trans Pattern Anal Mach Intell 31(12):2143–57
22. Le QV, Zou WY, Yeung SY, Ng AY (2011) Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis. In: Computer vision and pattern recognition, pp 3361–3368
23. Lecun Y, Cortes C (2010) The mnist database of handwritten digits
24. Lee H, Grosse R, Ranganath R, Ng AY (2009) Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In: International conference on machine learning, ICML 2009. Montreal, pp 609–616
25. Lin K, Yang HF, Hsiao JH, Chen CS (2015) Deep learning of binary hash codes for fast image retrieval. In: IEEE Conference on computer vision and pattern recognition workshops, pp 27–35
26. Liong VE, Lu J, Wang G, Moulin P, Zhou J (2015) Deep hashing for compact binary codes learning. In: IEEE Conference on computer vision and pattern recognition, pp 2475–2483
27. Liu X, He J, Lang B, Chang SF (2013) Hash bit selection: a unified solution for selection problems in hashing. In: Computer vision and pattern recognition, pp 1570–1577
28. Lu H, Li B, Zhu J, Li Y, Li Y, Xu X, He L, Li X, Li J, Serikawa S (2016) Wound intensity correction and segmentation with convolutional neural networks. Concurr Comput Pract Exper
29. Norouzi ME, Fleet DJ (2011) Minimal loss hashing for compact binary codes. In: International conference on machine learning, ICML 2011. Bellevue, pp 353–360
30. Oliva A, Torralba A (2001) Modeling the shape of the scene: a holistic representation of the spatial envelope. Int J Comput Vis 42(3):145–175

31. Pan Z, Zhang Y, Kwong S (2015) Efficient motion and disparity estimation optimization for low complexity multiview video coding. IEEE Trans Broad 61(2):1–1
32. Song J (2013) Effective hashing for large-scale multimedia search
33. Song J, Yang Y, Yang Y, Huang Z, Shen HT (2013) Inter-media hashing for large-scale retrieval from heterogeneous data sources, pp 785–796
34. Song J, Yang Y, Li X, Huang Z, Yang Y (2014) Robust hashing with local models for approximate similarity search. IEEE Trans Cybern 44(7):1225–1236
35. Song J, Gao L, Yan Y, Zhang D, Sebe N (2015) Supervised hashing with pseudo labels for scalable multimedia retrieval. In: ACM Multimedia, pp 827–830
36. Song J, Gao L, Zou F, Yan Y, Sebe N (2016) Deep and fast: deep learning hashing with semi-supervised graph construction *. Image Vis Comput 55:101–108
37. Strecha C, Bronstein A, Bronstein M, Fua P (2012) Ldahash: improved matching with smaller descriptors. IEEE Trans Pattern Anal Mach Intell 34(1):66–78
38. Torralba A, Fergus R, Freeman WT (2008) 80 million tiny images: a large data set for nonparametric object and scene recognition. IEEE Trans Pattern Anal Mach Intell 30(11):1958
39. Torralba A, Fergus R, Weiss Y (2008) Small codes and large image databases for recognition. In: IEEE Computer society conference on computer vision and pattern recognition, pp 1–8
40. Wang J, Kumar S, Chang SF (2010) Semi-supervised hashing for scalable image retrieval, pp 3424–3431
41. Wang J, Kumar S, Chang SF (2012) Semi-supervised hashing for large-scale search. IEEE Trans Pattern Anal Mach Intell 34(12):2393–406
42. Wang J, Zhang T, Song J, Sebe N, Shen HT (2016) A survey on learning to hash
43. Weiss Y, Torralba A, Fergus R (2008) Spectral hashing. In: Conference on neural information processing systems, Vancouver, pp 1753–1760
44. Williams R, Zipser D (1989) A learning algorithm for continually running fully recurrent neural networks. Neural Comput 1(2):270–280
45. Xia R, Pan Y, Lai H, Liu C, Yan S (2014) Supervised hashing for image retrieval via image representation learning
46. Xie S, Wang Y (2014) Construction of tree network with limited delivery latency in homogeneous wireless sensor networks. Wireless Person Commun 78(1):231–246
47. Xu H, Wang J, Li Z, Zeng G, Li S, Yu N (2011) Complementary hashing for approximate nearest neighbor search. In: IEEE International conference on computer vision, ICCV 2011. Barcelona, pp 1631–1638
48. Xu X, He L, Lu H, Shimada A, Taniguchi RI (2016) Non-linear matrix completion for social image tagging PP(99), pp 1–1
49. Xu X, He L, Shimada A, Taniguchi RI, Lu H (2016) Learning unified binary codes for cross-modal retrieval via latent semantic hashing. Neurocomputing 213:191–203
50. Zhong G, Yang P, Wang S, Dong J (2015) A deep hashing learning network. Comput Sci

Liangfu Cao is a undergraduate student in the school of Computer Science and Engineering, University of Electronic Science and Technology. His research interest is image and video retrieval.

Lianli Gao's background lines in Semantic Web, Machine Learning and Computer Vision. She received my PhD degree in Information Technology from University of Queensland, Brisbane, Australia. Currently, am an Associate Professor working at the University of Electronic Science and Technology of China.

Jingkuan Song received his PhD degree in Information Technology from The University of Queensland, Australia. He received his BS degree in Software Engineering from University of Electronic Science and Technology of China. Currently, he is a postdoctoral researcher in the Dept. of Information Engineering and Computer Science, University of Trento, Italy. His research interest includes large-scale multimedia search and machine learning.

Fumin Shen received his B.S. and Ph.D. degree from Shandong University and Nanjing University of Science and Technology, China, in 2007 and 2014, respectively. Currently he is a lecturer in school of Computer Science and Engineering, University of Electronic of Science and Technology of China, China. His major research interests include computer vision and machine learning, including face recognition, image analysis, hashing methods, and robust statistics with its applications in computer vision.

Dr. Yuan Wang works in the department of industrial systems and Engineering, National University of Singapore.