Robust and Index-Compatible Deep Hashing for Accurate and Fast Image Retrieval

Jing Liu[1,2], Dayan Wu[1,2], Wanqian Zhang[1,2], Bo Li[2(✉)], and Weiping Wang[2]

[1] Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China
{liujing,wudayan,zhangwanqian}@iie.ac.cn
[2] School of Cyber Security, University of Chinese Academy of Sciences, Beijing 100049, China
{libo,weipingwang}@iie.ac.cn

Abstract. Hashing methods have been widely used in large-scale image retrieval. However, the constraints on the hash codes of similar images learned by the previous hashing methods are too strong, which may lead to overfitting and difficult convergence. Besides, the binary codes output by the previous hashing methods are not optimally compatible with the multi-index approach, which is the most effective method for Hamming distance query acceleration. In this paper, we propose a novel Robust and Index-Compatible Deep Hashing (RICH) method to learn compact similarity-preserving binary codes, which focuses on improving the retrieval accuracy and time efficiency simultaneously. With the learned binary codes, we can achieve better results compared with the state-of-the-arts in retrieval accuracy. Meanwhile, remarkable promotions of the retrieval time efficiency have been made in the Hamming distance query process.

Keywords: Deep hashing · Image retrieval
Hamming distance query · The multi-index approach

1 Introduction

With the explosive growth of images on the web, much attention has been devoted to the nearest neighbor search via hashing methods. Mapping image data onto compact and similarity-preserving binary codes is important for large-scale image retrieval. In the literature, existing hashing methods can be grouped into two categories: data-independent methods and data-dependent methods.

In data-independent methods, the hash function is randomly generated which is independent of any training data. LSH [2] and KLSH [8] are representative data-independent methods. LSH uses random linear projections to produce binary codes, and KLSH is a kenelized method for dealing with high-dimensional and non-linear data. There are also some other variants of LSH [6,13] that have

R. Hong et al. (Eds.): PCM 2018, LNCS 11165, pp. 67–77, 2018.
https://doi.org/10.1007/978-3-030-00767-6_7

been proposed. Due to the limitations of making no use of training data, these methods are usually difficult to achieve satisfactory performance.

Data-dependent methods try to learn the hash function from training data. They can be further divided into unsupervised and supervised methods. Unsupervised methods only use the feature information of data points without using any label information during the learning procedure. ITQ [4] is one of the representative unsupervised methods, which iteratively optimizes the projection matrix of images to minimize the quantization error. In order to deal with the label information, supervised methods are proposed. SDH [15], KSH [1], MLH [14] are well known supervised methods. All of them use hand-crafted features of the training images, which have limitations in capturing the deep semantic information of images thus limit the retrieval accuracy of the learned codes.

Recently, many deep learning based hashing methods have been proposed. CNNH [18] and NINH [9] are early approaches adopting neural networks to learn hash codes. [9,11,12,17,19,20] are also well known methods of this kind. Most of these deep hashing methods are supervised and the supervised information is usually based on pair-wise or triplet labels. Due to the outstanding learning ability of the deep neural networks, they have shown much better performance than the traditional hashing methods.

However, most of the existing deep hashing methods try to make the hash codes of similar images exactly the same, which is unreasonable in some cases. As shown in Fig. 1, two similar images in the semantic space may be very different in visual, which is a common case in the existing real-world image data sets. Under this circumstance, forcing the hash codes of similar images to be exactly the same may lead to overfitting and difficult convergence.

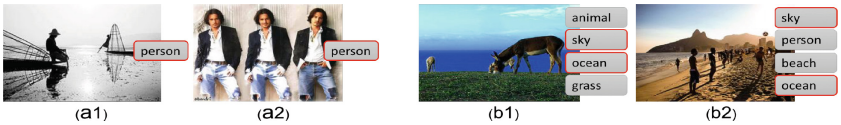

Fig. 1. Two exemplary image pairs from NUS-WIDE. The images in the group (a) are annotated with a single label. The images in the group (b) are annotated with multiple labels. Following previous works, both of them are considered as similar pair, even though they are very different in visual.

Besides, most current deep hashing methods are not compatible with the multi-index approach [3,5], which is an inverted index based method. The method splits each code into disjoint but consecutive blocks and creates a separate index for each block. In the searching phase, the images whose codes have no matching blocks with the query code will be filtered out. Consequently, the candidate images will be checked and ranked. However, the codes generated by previous deep hashing methods are not distributed uniformly, which will slow down the Hamming distance query process with multi-index.

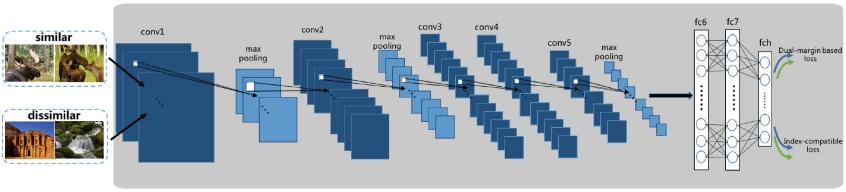

Fig. 2. Overview of the proposed framework. The network consists of 5 convolution layers and 3 fully connected layers. The last layer is hashing layer (fch), which has k output nodes.

In this paper, we introduce a novel framework based on deep learning model, named Robust and Index-Compatible Deep Hashing (RICH). An overview of the proposed framework is illustrated in Fig. 2. Through the proposed architecture, images are first encoded into real-valued feature vectors. Then each vector is converted to a hash code by a hash layer (fch). After that, these hash codes are used in a dual-margin based hashing loss that aims to preserve similarity between images and an index-compatible loss that aims to minimize the number of the matching blocks between the binary codes of dissimilar images. The contributions of this study can be summarized as follows: (1) We present a novel CNN based framework for learning hash functions to improve the retrieval accuracy and time efficiency simultaneously. (2) A loss function is elaborately designed to guide the learning of the neural network. With the proposed loss function, the learned hash functions are more scalable and robust to different data sets. Besides, the hash codes output by RICH are more compatible with the multi-index approach. (3) Extensive experiments demonstrate that the proposed method gains better retrieval accuracy and more robust performance than previous methods. Meanwhile, the retrieval time is also significantly reduced when adopting the multi-index approach.

2 The Proposed Approach

Suppose that we have a training set of N images $\{x_i\}_{i=1}^{N}$. Our goal is to learn the nonlinear hashing function $H : x \rightarrow h \in \{-1, 1\}^k$ mapping each image to k-bit binary code. Accordingly, our deep hash function is defined as:

$$h(x) = sign(f(x)) \tag{1}$$

where $f(x)$ indicates the output of layer fch. Therefore, k-bit binary codes can be obtained through k such hash functions $h(x) = [h_1(x), h_2(x), ..., h_k(x)]$. To encourage the fch layer representation to be optimal for hash coding and can be better applied to the multi-index approach, the loss function is elaborately designed.

2.1 Dual-Margin Based Loss Term

Our first goal is to make the codes of similar images close in the Hamming space, while the codes of dissimilar images far away from each other. Meanwhile, different from previous methods, we no longer make the codes of similar images exactly same. To achieve the goal, we propose a dual-margin based loss term as:

$$\begin{aligned} L_{hashing} = & \frac{1}{2} \sum_{s_{ij} \in S} \{ s_{ij} \max \left(D_h(h(x_i) \cdot h(x_j)) - m_0, 0 \right) \\ & + (1 - s_{ij}) \max \left(m_1 - D_h(h(x_i) \cdot h(x_j)), 0 \right) \} \end{aligned} \tag{2}$$

where $s_{ij} = 1$ indicates x_i and x_j are similar and $s_{ij} = 0$ implies x_i and x_j are dissimilar. $D_h(h(x_i) \cdot h(x_j))$ denotes the Hamming distance between two codes, and $m_0 > 0, m_1 > 0$ are margin threshold parameters. In the case of $s_{ij} = 1$, which means x_i, x_j are similar images, the first term will punish them only when their Hamming distance exceeds the threshold m_0. In the case of $s_{ij} = 0$, only the second term takes effects, and it will punish dissimilar images mapped to close binary codes when their Hamming distance falls below the margin threshold m_1.

However, it is intractable to directly optimize the Eq. (2), due to the binary constraints on $h(x)$ and discrete Hamming distance (i.e. $D_h(\cdot)$) computing. As implied in previous methods [11], we replace the Hamming distance by Euclidean distance. Meanwhile, in order to reduce the error of quantization, we also impose a regularizer on the real-valued network outputs to approach the desired discrete values $(-1/+1)$. Thus, the complete dual-margin based loss term is rewritten as:

$$\begin{aligned} \mathcal{L}_{hashing} = & \frac{1}{2} \sum_{s_{ij} \in S} \left\{ s_{ij} \max \left(||f(x_i) - f(x_j)||_2^2 - m_0, 0 \right) \right. \\ & \left. + (1 - s_{ij}) \max \left(m_1 - ||f(x_i) - f(x_j)||_2^2, 0 \right) \right\} \\ & + \lambda \sum_{i=1}^{N} \left(|||f(x_i)| - \mathbf{1}||_1 \right) \end{aligned} \tag{3}$$

where $f(x)$ is the continuous output vector of the hash layer. $|| \cdot ||_1$ is the L1-norm of vector, and $|| \cdot ||_2^2$ is the Euclidean-norm. $|\cdot|$ is the element-wise absolute value operation, and $\mathbf{1}$ is a vector of all ones. λ is a trade-off parameter that controls the strength of the regularizer. Note that the higher-order norms of the regularizer are also applicable. We choose the L1-norm on account of its less computational cost, which is beneficial for accelerating the training process.

2.2 Index-Compatible Loss Term

As we mentioned before, making the generated codes better applied to the multi-index approach is our second goal. However, there is a drawback of the previous deep hashing methods that the codes they generate is not uniformly distributed. Therefore, the majority of the codes have exactly the same value in some blocks. As shown in Fig. 3, all the codes have the same value in *block2*, which will make

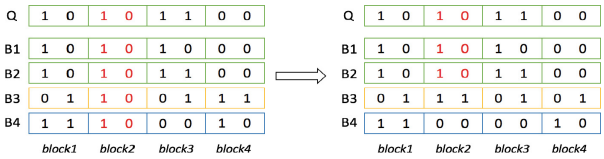

Fig. 3. Q is the hash code of the query image. B1 and B2 are the codes of images similar to the query image, while B3 and B4 are the ones dissimilar to the query image.

all the dissimilar images contained in the candidate sets when adopting the multi-index approach.

To address this problem, for corresponding blocks of the codes between dissimilar image pairs, we push them a certain distance away from each other. The index-compatible loss term is designed as follows:

$$\mathcal{L}_{index} = \frac{1}{2} \sum_{s_{ij} \in S} (1 - s_{ij}) \sum_{t=1}^{b} \max \left(m_2 - ||f^t(x_i) - f^t(x_j)||_2^2, 0 \right) \tag{4}$$

where $f^t(x)$ denotes the t-th block of the hash layer (fch) outputs of image x, and m_2 is a margin threshold parameter.

Integrating the index-compatible loss with the dual-margin based loss, we can get the objective loss function of RICH:

$$\mathcal{L}_{RICH} = \mathcal{L}_{hashing} + \alpha \mathcal{L}_{index} \tag{5}$$

where $\alpha > 0$ is a trade-off parameter between $\mathcal{L}_{hashing}$ and $\mathcal{L}_{index}$.

2.3 Optimization

By substituting Eqs. (3) and (4) into Eq. (5), we rewrite the overall loss function of RICH as follows:

$$\begin{aligned} \mathcal{L}_{RICH} = & \frac{1}{2} \sum_{s_{ij} \in S} \left\{ s_{ij} \max \left(||f(x_i) - f(x_j)||_2^2 - m_0, 0 \right) \right. \\ & \left. + (1 - s_{ij}) \max \left(m_1 - ||f(x_i) - f(x_j)||_2^2, 0 \right) \right\} \\ & + \frac{\alpha}{2} \sum_{s_{ij} \in S} (1 - s_{ij}) \sum_{t=1}^{b} \max \left(m_2 - ||f^t(x_i) - f^t(x_j)||_2^2, 0 \right) \\ & + \lambda \sum_{i=1}^{N} \left(|||f(x_i)| - \mathbf{1}||_1 \right) \end{aligned} \tag{6}$$

Next, we can employ back-propagation algorithm with mini-batch gradient descent method to train the network. So the gradients of Eq. (6) w.r.t $f(x_i), f(x_j) \, \forall i, j$ need to be computed. Since the *max* operation and the absolute

value operation is non-differentiable at some certain points, we use subgradients instead, and define subgradients to be 1 at these points. In order to express clearly, we have a separate calculation on the gradients of the regularizer.

$$\begin{aligned}
\frac{\partial \mathcal{L}_{hashing}}{\partial f(x_i)} &= (f(x_i) - f(x_j))(s_{ij} I_{dis>m_0} - (1 - s_{ij}) I_{dis<m_1}) + \delta(f(x_i)) \\
\frac{\partial \mathcal{L}_{hashing}}{\partial f(x_j)} &= (f(x_j) - f(x_i))(s_{ij} I_{dis>m_0} - (1 - s_{ij}) I_{dis<m_1}) + \delta(f(x_j)) \\
\frac{\partial \mathcal{L}_{index}}{\partial f^t(x_i)} &= (1 - s_{ij})(f^t(x_i) - f^t(x_j)) I_{||f^t(x_i) - f^t(x_j)||_2^2 < m_2} \\
\frac{\partial \mathcal{L}_{index}}{\partial f^t(x_j)} &= (1 - s_{ij})(f^t(x_j) - f^t(x_i)) I_{||f^t(x_i) - f^t(x_j)||_2^2 < m_2}
\end{aligned} \tag{7}$$

where

$$\delta(x) = \begin{cases} 1, & -1 \leq x \leq 0 \ or \ x \geq 1 \\ -1, & otherwise \end{cases} \tag{8}$$

where dis denotes the Euclidean distance $||f(x_i) - f(x_j)||_2^2$. And we use the function $I_{condition} = 1$ to indicate $condition$ is true, and $I_{contidion} = 0$ when $condition$ is false. With the computed subgradients over mini-batches, the rest of the back-propagation can be done in standard manner.

3 Experiments

3.1 Evaluation Setup

We conduct extensive experiments on two widely-used benchmark datasets, **CIFAR-10** and **NUS-WIDE**.

CIFAR-10 is a public image dataset, which consists of 60,000 32×32 images belonging to 10 categories. We randomly select 100 images per class (1,000 images in total) as the test query set, 5000 images per class (50,000 images in total) as the training set.

NUS-WIDE contains 269,648 multi-label images collected from Flickr. The association between images and 81 concepts are manually annotated. Following [11,20], we use the images associated with the 21 most frequent concepts, where each of these concepts associates with at least 5,000 images, resulting in a total of 195,834 images. Generally, if two images share at least one same label, they are considered similar, and dissimilar otherwise. We randomly select 2,100 images (100 images per class) for testing queries and the rest is used for training.

We use both deep learning based hashing methods and traditional hashing methods for thorough comparison. Five deep hashing methods: **DSH** [11], **DHN** [20], **DPSH** [10], **NINH** [9] and **CNNH** [18]. For fair comparison, we implement the **DSH** and **DHN** with the same AlexNet network structures. Most of other results are directly reported from previous works. Traditional hashing methods consists of **SH** [16] and **ITQ** [4], both of them use the CNN feature.

We implement RICH based on the open source Caffe[1] framework, and employ the AlexNet [7] neural network architecture, finetune convolutional layers and fully-connected layers that were copied from the pre-trained model. We use the mini-batch stochastic gradient descent with 0.9 momentum. The quantization penalty parameter λ and trade-off parameter α are chosen by cross-validation from 10^{-5} to 10^{2} with a multiplicative step-size 10. For the dual margin parameter m_0, we empirically set $m_0 = 4$ for CIFAR-10 and $m_0 = 8$ for NUS-WIDE, which means we allow that there exists 1/2 different bits between the codes of similar image pairs in CIFAR-10/NUS-WIDE. We set another margin parameter $m_1 = 2k$, which is the same as [11]. That means we expect at least half of the binary codes between dissimilar images to be different. For the parameters in the index-compatible loss term, we split codes to $\frac{k}{2}$ blocks, then we set $m_2 = 4$, which means we try to make each block of dissimilar codes share at least one distinct bit. The multi-index approach is implemented with Lucene-6.4.2[2], which is a high-performance and full-featured text search engine library.

We mainly evaluate the retrieval accuracy and retrieval time efficiency of RICH and other methods. Following [10,11,20], we use the mean Average Precision (mAP) to measure the retrieval accuracy. For NUS-WIDE, following the previous work [10], we calculate the mAP values within the top 5000 returned neighbors. To evaluate the time efficiency, we calculate the overall time of the Hamming distance query process.

3.2 Comparison of Retrieval Accuracy

The mAP results are reported in Table 1, which shows that the proposed RICH method substantially outperforms all the comparison methods.

As shown in Table 1, most of the deep hashing methods perform better than the traditional hashing methods, validating the advantage of learning image representations over using hand-crafted features. Furthermore, DSH achieves a better performance on CIFAR-10 than DPSH, but on NUS-WIDE, DSH is inferior to DPSH. It indicates that they are not robust enough on different data sets. However, RICH performance better on both CIFAR-10 and NUS-WIDE, validating its robustness. We attribute this approvement to the proposed dual-margin based loss term which can effectively prevent overfitting.

To further verify the effectiveness of the proposed dual-margin based loss term, we compare RICH with RICH*. As listed in Table 1, the mAP results of RICH* is obviously lower than RICH, which proves that the dual-margin based loss term can improve the retrieval accuracy. Furthermore, the promotion on NUS-WIDE is more significant, which means the dual-margin based loss is more effective on multi-label images data.

[1] http://caffe.berkeleyvision.org/.
[2] https://lucene.apache.org/.

Table 1. Comparison of mAP w.r.t. different number of bits on NUS-WIDE and CIFAR-10. RICH* is a variant of RICH when $m_0 = 0$.

Methods	NUS-WIDE				CIFAR-10			
	16-bits	32-bits	48-bits	64-bits	16-bits	32-bits	48-bits	64-bits
RICH	0.7867	0.7950	0.8177	0.8291	0.8801	0.8892	0.9062	0.9080
RICH*	0.7598	0.7680	0.7694	0.7740	0.8796	0.8839	0.9047	0.9082
DSH	0.7499	0.7602	0.7604	0.7631	0.8538	0.8831	0.9042	0.9065
DHN	0.8153	0.8178	0.8237	0.8215	0.8652	0.8792	0.8921	0.8890
DPSH	0.7752	0.7940	0.8120	0.8253	0.7206	0.7440	0.7570	0.7621
NINH	0.6866	0.7130	0.7155	0.7241	0.5662	0.5580	0.5810	0.5986
CNNH	0.6154	0.6255	0.6080	0.6098	0.4839	0.5090	0.5220	0.5534
ITQ+CNN	0.4235	0.4334	0.4607	0.4303	0.2436	0.2550	0.2610	0.2630
SH+CNN	0.3662	0.3560	0.3834	0.3405	0.1612	0.1610	0.1610	0.1620

3.3 Comparison of Time Efficiency

In general, the multi-index approach consists of two phases: *searching* phase and *checking* phase. In the *searching* phase, the binary code of the query image is first splitted into blocks, then we search the multi-index to identify all the binary codes that contain at least one matching block. In the *checking* phase, we rank the candidate images according to the Hamming distance between the binary codes of the candidate images and query image. We calculate the overall time of these two phases. Note that all the binary codes are stored in memory.

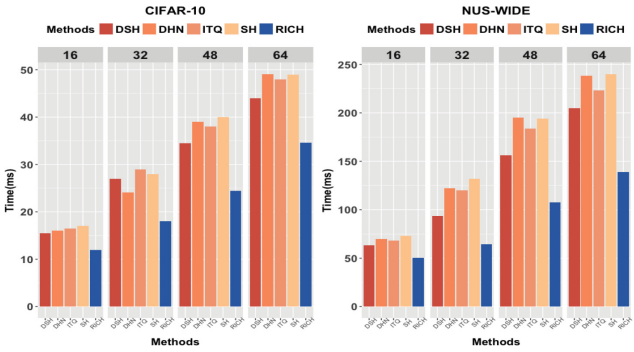

Fig. 4. Comparison of the query time w.r.t. different number of bits on NUS-WIDE and CIFAR-10.

The comparison result is presented in Fig. 4. With the binary codes learned by RICH, the overall query time is significantly reduced comparing to other methods. To be specific, there is a **29.5%–37.7%** reduction of the total query

time on CIFAR-10. For NUS-WIDE, there is a **22.3%–47.5%** reduction. We attribute this improvement to the reduction of the candidate images. We calculate the size of the candidate images of different method, and the results are shown in Table 2. We can see that the size of the candidate images are significantly reduced with the codes generated by RICH. Specifically, there is a **43.35%** reduction on NUS-WIDE and **21.80%** reduction on CIFAR-10.

Table 2. Comparison of Recall and the size of candidate images on NUS-WIDE and CIFAR-10 with 32 bits.

Methods	NUS-WIDE					CIFAR-10	
	≥ 1	≥ 2	≥ 3	≥ 4	#candidates	–	#candidates
DSH	0.9985	0.9994	0.9998	0.9999	188741	0.9999	49762
DHN	0.9978	0.9993	0.9997	0.9999	188447	0.9999	49951
RICH	0.9679	0.9888	0.9909	0.9999	106841	0.9999	38942

In Table 2, we further record the recall rate. On CIFAR-10, both the comparison methods and RICH achieve a recall rate of almost 100%, which means that almost no similar images are filtered out in the searching phase. On NUS-WIDE, "$\geq n$" indicates the recall rate of similar images sharing no less than n same labels. Note that with the increasing number of the sharing labels between the database images and the query image, the recall rate increases, which means that the images with higher similarity to the query images are less likely to be filtered out.

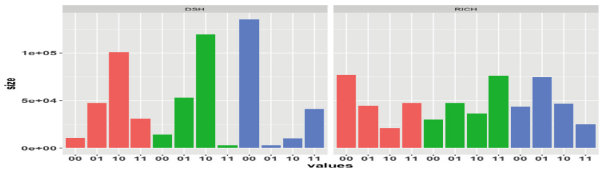

Fig. 5. Distribution of the codes generated by DSH (left) and RICH (right), with 32 bits on NUS-WIDE. The three blocks are randomly selected. The x-axis shows all the possible values of each block, and Y-axis shows the number of images. For the third block, most of the codes generated by DSH are valued "00" which is a extremely non-uniform distribution.

We further visualize the distribution of the codes generated by DSH an RICH. As shown in Fig. 5, the codes output by RICH are more uniform than the ones learned by DSH, which verifies the effectiveness of the index-compatible loss term.

4 Conclusion

In this paper, we propose a robust and index-compatible deep hashing method for accurate and fast image retrieval, named RICH. A loss function is elaborately designed, which consists of a dual-margin based loss term and an index-compatible loss term. With the learned codes, both the retrieval accuracy and the time efficiency are significantly improved. We attribute the improvement of retrieval accuracy to the relaxation of constraints on similar image pairs, and that of the time efficiency to the more uniform distribution of the codes. Extensive experiments validate the effectiveness of the proposed method.

References

1. Chang, S.F.: Supervised hashing with kernels. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 2074–2081 (2012)
2. Gionis, A., Indyk, P., Motwani, R., et al.: Similarity search in high dimensions via hashing. VLDB **99**, 518–529 (1999)
3. Gog, S., Venturini, R.: Fast and compact hamming distance index. In: SIGIR, pp. 285–294. ACM (2016)
4. Gong, Y., Lazebnik, S., Gordo, A., Perronnin, F.: Iterative quantization: a procrustean approach to learning binary codes for large-scale image retrieval. TPAMI **35**(12), 2916–2929 (2013)
5. Greene, D., Parnas, M., Yao, F.: Multi-index hashing for information retrieval. In: Foundations of Computer Science, pp. 722–731. IEEE (1994)
6. Ji, J., Li, J., Yan, S., Zhang, B., Tian, Q.: Super-bit locality-sensitive hashing. In: International Conference on Neural Information Processing Systems, pp. 108–116 (2012)
7. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: NIPS, pp. 1097–1105 (2012)
8. Kulis, B., Grauman, K.: Kernelized locality-sensitive hashing for scalable image search **30**(2), 1895–1900 (2009)
9. Lai, H., Pan, Y., Liu, Y., Yan, S.: Simultaneous feature learning and hash coding with deep neural networks. In: CVPR, pp. 3270–3278 (2015)
10. Li, W.J., Wang, S., Kang, W.C.: Feature learning based deep supervised hashing with pairwise labels. arXiv preprint arXiv:1511.03855 (2015)
11. Liu, H., Wang, R., Shan, S., Chen, X.: Deep supervised hashing for fast image retrieval. In: CVPR, pp. 2064–2072 (2016)
12. Liu, L., Shao, L., Shen, F., Yu, M.: Discretely coding semantic rank orders for supervised image hashing. In: CVPR, pp. 5140–5149 (2017)
13. Mu, Y., Yan, S.: Non-metric locality-sensitive hashing. In: Twenty-Fourth AAAI Conference on Artificial Intelligence, pp. 539–544 (2010)
14. Norouzi, M., Fleet, D.J.: Minimal loss hashing for compact binary codes. In: ICML, pp. 353–360 (2011)
15. Shen, F., Shen, C., Liu, W., Tao Shen, H.: Supervised discrete hashing. In: CVPR, pp. 37–45 (2015)
16. Weiss, Y., Torralba, A., Fergus, R.: Spectral hashing. In: NIPS, pp. 1753–1760 (2009)
17. Wu, D., Lin, Z., Li, B., Ye, M., Wang, W.: Deep supervised hashing for multi-label and large-scale image retrieval. In: ICMR, pp. 150–158. ACM (2017)

18. Xia, R., Pan, Y., Lai, H., Liu, C., Yan, S.: Supervised hashing for image retrieval via image representation learning. In: AAAI Conference on Artificial Intelligence (2014)
19. Yao, T., Long, F., Mei, T., Rui, Y.: Deep semantic-preserving and ranking-based hashing for image retrieval. In: IJCAI, pp. 3931–3937 (2016)
20. Zhu, H., Long, M., Wang, J., Cao, Y.: Deep hashing network for efficient similarity retrieval. In: AAAI, pp. 2415–2421 (2016)