Cascaded Deep Hashing for Large-Scale Image Retrieval

Jun Lu[1] and Li Zhang[1,2(✉)]

[1] School of Computer Science and Technology and Joint International, Research Laboratory of Machine Learning and Neuromorphic Computing, Soochow University, Suzhou 215006, Jiangsu, China
zhangliml@suda.edu.cn
[2] Provincial Key Laboratory for Computer Information Processing Technology, Soochow University, Suzhou 215006, Jiangsu, China

Abstract. It is very crucial for large-scale image retrieval tasks to extract effective hash feature representations. Encouraged by the recent advances in convolutional neural networks (CNNs), this paper presents a novel cascaded deep hashing (CDH) method to generate compact hash codes for highly efficient image retrieval tasks on given large-scale datasets. Specifically, we ingeniously utilize three CNN models to learn robust image feature representations on a given dataset, which solves the issue that categories with poor feature representation have a fairly low retrieval precision. Experimental results indicate that CDH outperforms some state-of-the-art hashing algorithms on both CIFAR-10 and MNIST datasets.

Keywords: Image retrieval · Convolutional neural networks · Hash code
Image representation

1 Introduction

In recent years, multimedia data including images have being produced on the Internet every day, making it extremely hard to retrieve similar data from a large-scale database. Content-based image retrieval (CBIR) is a popular image retrieval method, which searches for similar images according to compare the content of images [1–3]. The main steps in CBIR include image representation and similarity measurement. Along this research track, the most challenging issue is to improve the "semantic gap" between the pixel-level information captured by machines and semantics from human perceptions [3, 4].

Recent studies [5–8] revealed that the deep features obtained by convolutional neural networks (CNNs) are more suitable for computer vision tasks, which is a significant breakthrough compared with traditional methods using hand-crafted features [1, 2, 9]. Better effect of deep features gives the credit to advantages of deep CNNs which can learn high-level abstractions in images. But deep features are high-dimensional, which makes it unwise to directly compute the similarity between two high-dimensional vectors. For a large scale image database, it is an undesirable method would consume a lot of time and computing resources.

L. Cheng et al. (Eds.): ICONIP 2018, LNCS 11306, pp. 419–429, 2018.
https://doi.org/10.1007/978-3-030-04224-0_36

Hashing approaches have been turned out to be more appropriate when images need to be retrieved from a large-scale image database, because of its fast speed for searching process and low memory costs [10–17]. Projecting the high-dimensional data into a low-dimensional space, hashing methods can generate compact binary codes that approximately preserve the data structure in the original space. Binary codes are easy to store and compare, which dramatically reduces the computational and memory cost. Hashing algorithms consist of two groups: data-independent and data-dependent methods [10–17].

Most of early researchers pay more attention to data-independent methods which employ random hash functions to map data points to similar hash codes. The most representative one is the locality-sensitive hashing (LSH) [10] and its variants [11], which use random projections to produce binary codes. However, data-independent methods are unpractical because they would produce long codes.

Fortunately, data-dependent hashing methods through machine learning have shown their effectiveness in overcoming the issue mentioned above [12–17]. The data-dependent methods can better access compact and short hash codes from the large-scale data. In general, these techniques are made up of two parts: (1) Generating visual descriptor feature vectors from images; and (2) Encoding vectors into binary hash codes by implementing projection and quantization steps. Existing data-dependent hash methods can be further split into supervised (semi-supervised) and unsupervised methods. The unsupervised methods only utilize the training data without labels to acquire hash functions, which encode neighborhood relation of samples from a certain metric space into the Hamming space [12, 13]. For instance, Spectral Hashing (SH) [12] tries to preserve the similarity structures defined in the original space.

Supervised methods boost hash codes by taking advantage of label information to learn more complex semantic similarity [14–17]. In the inspiration of deep learning, some researchers utilized deep architectures for hash learning under the supervised framework. Xia et al. [15] proposed a hashing method based on the supervised data to acquire binary hashing codes through deep learning. Although this approach is proven effective, it consumes too much computational time and considerable storage space for the input of a pair-wised similarity matrix of data. Very recently, Lin et al. [16] put forward an effective method that based on a deep CNN model to learn simultaneously binary codes and image representation when the image data are labeled.

There is such a phenomenon in this method that the retrieval performance is closely related with the classification accuracy of deep CNN models. The categories which can be recognized well by a CNN model also have a high retrieval performance, but the categories with low classification accuracies have a fairly low retrieval precision. Thus, the images recognized bad could reduce the efficiency of image retrieval.

In order to address the issue mentioned above, a novel and effective cascaded deep hashing (CDH) algorithm based on multiple CNNs is developed for the task of large-scale image retrieval. Different from other supervised methods (such as [16]), we use three CNNs, a global CNN and two local CNNs, to generate binary codes. The global CNN is used to recognize the label of images and generate candidate binary codes. The two local CNNs can improve the representation ability of deep features, especially the categories with poor classification ability.

The rest of this paper is as follows. Section 2 elaborates on details of CDH. Section 3 compares CDH with several state-of-the-art methods and reports experimental results. Finally, we conclude this paper in Sect. 4.

2 Our Method

Recent studies have proved that deep hashing methods using CNN can achieve better results in content-based image retrieval [15, 16]. But the precision of image retrieval depends on the classification accuracy of CNN models, and the categories with low classification accuracies have a fairly low retrieval precision. That means hash-like binary codes learned from deep features of poor representation are inefficient for image retrieval tasks in this case. In order to improve this situation, we present a cascaded deep hashing (CDH) method for hash code learning.

We expect that CDH could raise the classification accuracy of the categories with poor classification ability, which makes the hash-like binary codes of all categories have a good representation ability to be used for retrieving.

2.1 Cascaded Models

Consider a large-scale image database consisting of c categories as $X = \{X_i\}_{i=1}^{c}$, where X_i represents the set of *ith* category. Let the label set of images be $Y = \{1, 2, \cdots, c\}$. Further, we partition X into two subsets, the validation set $X_{validation}$ and the training set X_{train}.

Figure 1 shows the training framework of CDH. From Fig. 1, we can see that the CDH model includes three CNNs, one global model CNN_1, and two local models CNN_2 and CNN_3. The training data for CNN_1 is the whole training set X_{train}. The training data for both CNN_2 and CNN_3 are subsets of X_{train}, which are dependent on the classification results of CNN_1.

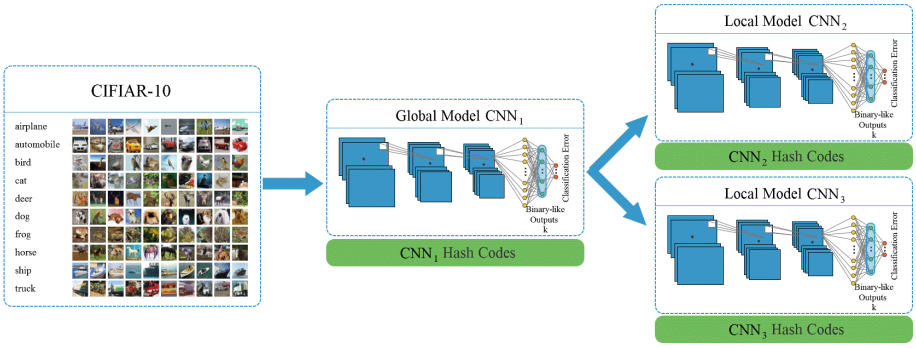


Fig. 1. The training framework of CDH

First, we determine the structure of CNNs a priori, so that the network would have a good classification ability on X_{train}. A typical CNN architecture is given in Fig. 2, which is usually composed of convolution layers, pooling layers and fully connected layers. Then the global model CNN_1 can be obtained by training this network on the training set X_{train}. Second, we apply CNN_1 to the subset $X_{validation}$ and calculate the classification accuracy of each class in $X_{validation}$. Let $CP = \{p_1, p_2, \cdots, p_c\}$ be the set of classification accuracy for all class, where p_i is the classification accuracy for class i. Sort elements in the set CP in descending order and define the sorted CP as $CP_S = \{p_{s_1}, p_{s_2}, \cdots, p_{s_c}\}$.

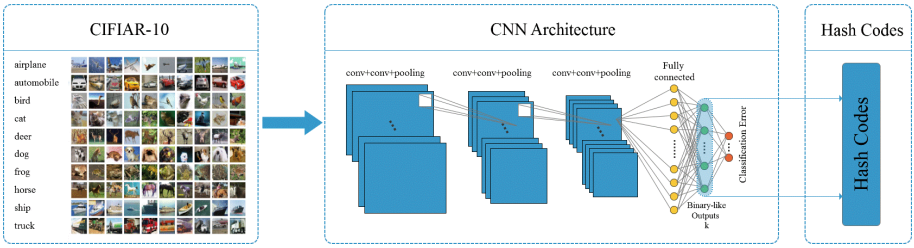


Fig. 2. A typical CNN architecture

Next, we consider how to divide the set X into two subsets X_{good} and X_{bad}, where X_{good} consists of samples belonging to categories have higher classification accuracies, and $X_{bad} = X - X_{good}$. Thus, we need to determine a threshold to separate categories at first. If

$$i^* = \underset{i=2,\cdots,c-1}{\operatorname{argmax}} (p_{s_i} - p_{s_{i-1}}) \tag{1}$$

we can partition CP_S to two subsets $\{p_{s_1}, p_{s_2}, \cdots, p_{s_{i*}}\}$ and $\{p_{s_{i*+1}}, \cdots, p_{s_c}\}$. Therefore, the corresponding subset $Y_{good} = \{s_1, \cdots, s_{i*}\} \subseteq Y$ represents the set of categories with good classification ability. The remaining categories construct the subset $Y_{bad} = Y - Y_{good}$. Correspondingly, the samples in X_{good} belong to the classes in Y_{good}, and those in X_{bad} to the classes in Y_{bad}.

Then, we train the same CNN on X_{good} and X_{bad} to obtain the local model CNN_2 and CNN_3, respectively.

2.2 Learning Binary Codes with Cascaded CNN Models

Lin analyzed the deep CNN and showed that the final outputs of the classification layer rely on a set of k hidden attributes with each attribute on or off [16]. It means images having the same label would induce similar binary activations. According to the above point of view, it is an effective way to learn hash-like binary codes by binarizing the k activations by a threshold $\theta \in \mathbb{R}$. As shown in Fig. 2, we set the latent layer with k nodes in front of the output layer in the network.

For an image $\mathbf{x}_i$, we denote the output vector of the latent layer by $\mathbf{o}_i = [o_1^i, \ldots, o_k^i]^T \in \mathbb{R}^k$. Then, the binary codes of image $\mathbf{x}_i$ can be represented as $\mathbf{h}_i = [h_1^i, h_2^i, \cdots, h_k^i] \in \{0, 1\}^k$, where

$$h_j^i = \begin{cases} 1, & \text{if } o_j^i \geq \theta \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

We must decide an appropriate value for the threshold θ for different database, which makes the binary codes more effective for image retrieval.

Using the above mentioned method, we use CNN_1, CNN_2 and CNN_3 to generate hash codes for X, X_{good} and X_{bad}, respectively. Let H_{global}, H_{good} and H_{bad} represent the hash code sets that are obtained from X, X_{good} and X_{bad}, respectively.

2.3 Image Retrieval

For a query image, our goal is to search similar images from the given dataset. The method in [16] directly retrieves the query image in one hash code database, or H_{global}. However, we should strength the representation ability of the images, especially those recognized bad by CNN_1. To fulfill this idea, we use a cascaded search method to retrieval the similar images. Figure 3 shows the retrieval process.

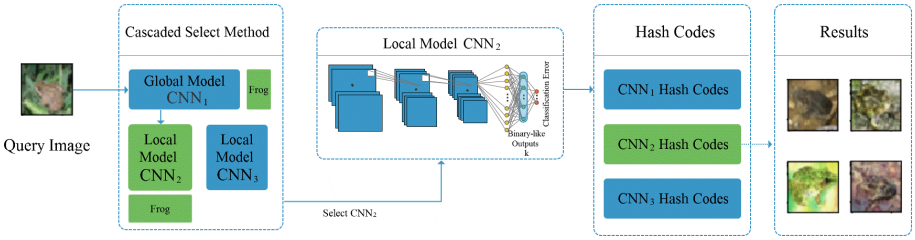


Fig. 3. The retrieval process of CDH

Given a query image $\mathbf{x}$, we input it into CNN_1 and receive the output as the prediction result denoted as y_{global}. If y_{global} is a component of Y_{good}, we use CNN_2 to get the prediction label of $\mathbf{x}$ and define it as y_{good}. If y_{global} is in Y_{bad}, we input $\mathbf{x}$ to CNN_3 to obtain the prediction y_{bad}.

Since we have generated three hash code sets: H_{global}, H_{good} and H_{bad}, from which we need to select an appropriate hash code set for image retrieval. The final hash code set H_{goal} is defined as

$$H_{goal} = \begin{cases} H_{good}, & if\ y_{global} = y_{good} \\ H_{bad}, & if\ y_{global} = y_{bad} \\ H_{global}, & otherwise \end{cases} \tag{3}$$

Once H_{goal} is determined, we can generate the hash code $\mathbf{h}$ for $\mathbf{x}$ using the corresponding CNN model. For example, if $H_{goal} = H_{good}$, then we use the local model CNN_2 to generate $\mathbf{h}$. Moreover, retrieval is carried out in H_{goal}.

Suppose we need to search out t images that are most similar to $\mathbf{x}$. The Hamming distance between the hash code of query image and that of any training sample is taken as their similarity. The smaller the Hamming distance is, the higher level the similarity of the two images is. The candidates are ranked in ascending. We select the top t images as the results of retrieval.

3 Experiments

To verify the effectiveness of our proposed method, we perform experiments on two image datasets, MNIST [18] and CIFAR-10 [19]. In the following, we first describe datasets and experimental settings, and then analyze experimental results.

3.1 Datasets

MNIST Dataset [18] contains 70 K 28×28 gray scale images belonging to 10 categories of handwritten Arabic numerals from 0 to 9. There are 60,000 training images, and 10,000 test images.

CIFAR-10 Dataset [19] contains 60 K 32×32 color tiny images which are categorized into10 classes (6 K tiny images per class). Each image belongs to one of the 10 classes in a single-label dataset.

3.2 Experimental Setting

The basic network is made up of three convolution-pool layers and three fully connected layers sequencely. The size of filters in convolution layers is 3×3 and the stride is 1. There are 64, 64, and 128 filters in the three convolution layers, respectively. Each convolution layer follows a pool layer with a stride of 2. Besides, the first fully connected layer contains 500 nodes, the second (latent layer) has k (the hash code length) nodes and the third (output layer) has c nodes (the label number).

To illustrate the effectiveness of our retrieval method, we compare CDH with six typical hashing methods: DLBHC [16], CNNH+ [15], KSH [17], BRE [14], LSH [11], and SH [12]. We evaluate the retrieval procedure by a Hamming ranking-based criterion. Given a query image, we find the t images with the smallest Hamming distance between it and training samples. The average precision (AP) for this query image is as

$$Precison@t = \frac{\sum_{i=1}^{t} \mathrm{Rel}(i)}{t} \tag{4}$$

where $Rel(i)$ is the ground truth relevance between a query $\mathbf{x}$ and the *ith* ranked image [16]. Here, we consider only the category label in measuring the relevance so $Rel(i) \in \{0, 1\}$, where $Rel(i) = 1$ if the query and the *ith* image have the same label; otherwise $Rel(i) = 0$. The mean retrieval precision (MRP) is used to measure the retrieval ability of these methods, which is the mean of AP on all query images.

3.3 Experimental Results on CIFAR-10 Dataset

A. Performance of Image Classification

When training the CNN model on CIFAR-10, the output layer is set as 10-way softmax to predict 10 object categories. In the latent layers, we fit the nodes of neurons k range from 16 to 64 to measure the performance of the latent layer embedded in the deep CNN model. The stochastic gradient descent (SGD) method is adopted to train CNN with 150 iterations and a learning rate of 0.01 on the CIFAR-10 dataset.

Table 1 gives MRP of three models. As shown in Table 1, the local models can effectively improve the classification performance of X_{good} and X_{bad} compared with the global model.

Table 1. MPR (%) of three models on data partition of CIFAR-10 dataset.

Data subset	Model		
	CNN_1	CNN_2	CNN_3
X_{train}	88.96	–	–
X_{good}	91.53	93.35	–
X_{bad}	82.21	–	88.96

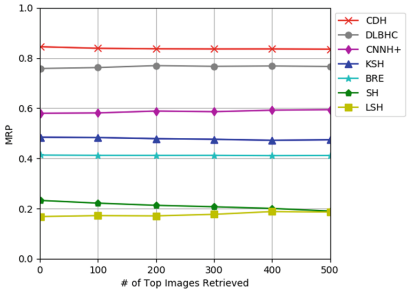


Fig. 4. MRP vs. top t images with 48 bits on CIFAR10

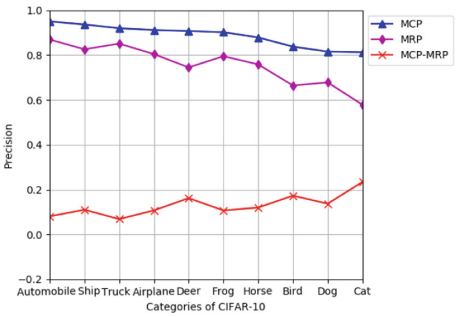
B. Performance of Images Retrieval

In this experiment, we map images to the hash codes from 16 to 64 for image retrieval measured with the hamming distance. To compare with traditional hashing approaches in hand-craft representation, 512-dimensional generalized search tree (GIST) features are extracted from each image [20].

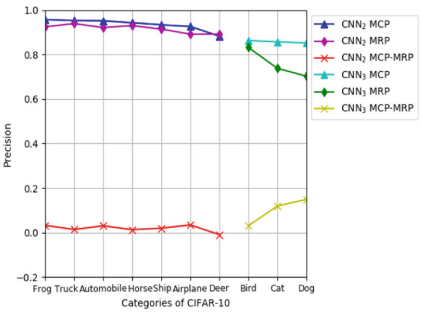
Table 2 shows the MRP of the top 500 returned images with different lengths of hash codes, where the best results are in bold. Figure 4 shows MRP regarding to various numbers of the top images received from compared methods. From experimental results, we can see that CDH obviously has the best experimental results among the compared methods, including unsupervised and supervised ones.

We also investigate more details for the relationship between classification accuracy and retrieval accuracy. We take the hash codes of 48 as an example, as shown in Fig. 5.

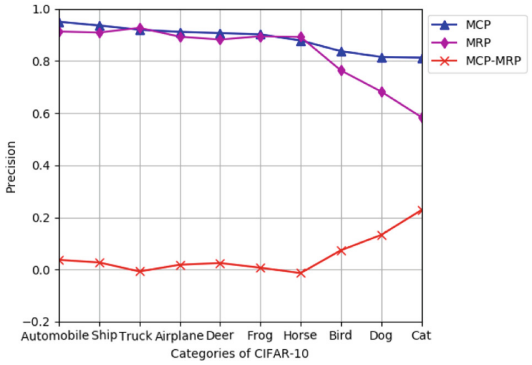
In Fig. 5(a), we can see that categories with slightly lower classification accuracy obtained by the global model CNN_1 have the bigger difference between the mean classification accuracy (MCP) and the mean retrieval accuracy. Figure 5(b) shows that we can improve the classification accuracy to enhance the deep feature representation ability with the help of local models CNN_2 and CNN_3. Inspection on Fig. 5(b) indicates that those categories with poor classification accuracy also receive pretty good retrieval accuracy by CNN_3. Figure 5(c) reports that CDH can generally obtain higher retrieval accuracy in the cascade way.



(a) Relationship on CNN_1



(b) Relationship on CNN_2 *and* CNN_3



(c) Relationship on *Cascaded Model*

Fig. 5. Relationship between mean classification prediction (MCP) and MRP of each category on different models.

Table 2. MAP (%) with various number of bits on the CIFAR-10 dataset

Method	16 bits	32 bits	48 bits	64 bits
CDH	**82.08**	**82.96**	**83.40**	**83.78**
DLBHC	73.39	74.18	75.16	75.99
CNNH+	58.63	58.94	59.31	59.98
KSH	41.03	41.78	42.07	42.22
BRE	19.22	19.88	20.43	20.51
SH	19.63	19.87	19.99	20.06
LSH	15.66	16.21	16.44	16.51

Figure 6 shows the top images retrieved by our method CDH and the state-of-the-art method DLBHC. CDH can successfully retrieve images with relevant categories and similar appearance. It can be easily found that the images retrieved by CDH are more appearance-relevant according to our empirical eyeball checking, which makes CDH have better performance.

3.4 Experimental Result on MNIST Dataset

A. Performance of Image Classification

To transfer the deep CNN to the dataset of MNIST, we modify the latent layer to 10-way softmax to predict 10 digit classes and k is also set from 16 to 64. We then train our cascaded model on the MNIST dataset. In Table 3, we list the classification accuracy of three models on different parts of MNIST.

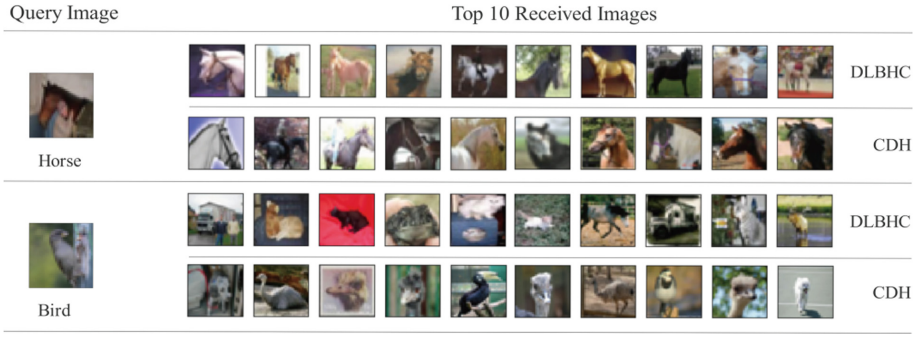


Fig. 6. The retrieval process of CDH

Table 3. MPR (%) of three models on data partition of MNIST dataset.

Data subset	Model		
	CNN_1	CNN_2	CNN_3
X_{train}	99.41	–	–
X_{good}	99.49	99.72	–
X_{bad}	99.23	–	99.51

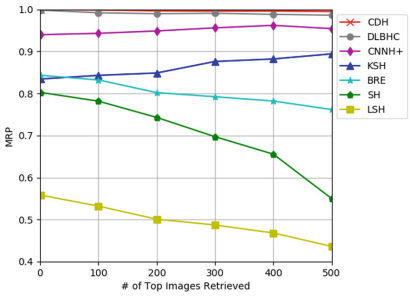


Fig. 7. MRP vs. top t images with 48 bits on MNIST.

B. Performance of Images Retrieval

In order to make a comparison fairly with other hashing methods, we unify the evaluation method that retrieves the relevant images by hash codes from 16 to 64 and using the Hamming distance. We still use the 512-d GIFT features for traditional hashing learning approach. We can see the precision (MRP) of top 500 returned image with different lengths of hash codes in Table 4, where the best results are in bold. It can be seen that our method has excellent results no matter how many images are retrieved. Figure 7 gives the relationship curves of the precision (MRP) vs. the number of the top retrieved samples. CDH can also stand out when compared with other fine methods.

Table 4. MAP (%) with various number of bits on the MNIST dataset

Method	16 bits	32 bits	48 bits	64 bits
CDH	**99.53**	**99.54**	**99.56**	**99.57**
DLBHC	98.12	98.34	98.63	98.83
CNNH+	90.34	90.89	91.23	91.89
KSH	84.93	86.23	88.57	90.04
BRE	72.33	74.58	76.62	78.92
SH	48.91	49.98	51.12	53.43
LSH	42.14	44.33	45.58	46.63

4 Conclusions

We present a cascaded framework to generate compact and short hashing codes for large-scale image retrieval. We use multiple CNN models to boost feature expression ability of images, so that our hash-like binary codes are more suitable for image retrieval. Experimental results show that CDH has superior performance over the previous best retrieval results, which has an elevation of about 8% and 1% on the CIFAR-10 and MNIST datasets, respectively.

Acknowledgments. This work was supported in part by the National Natural Science Foundation of China under Grants No. 61373093, No. 61402310, No. 61672364 and No. 61672365, by the Soochow Scholar Project of Soochow University, by the Six Talent Peak Project of Jiangsu Province of China, by the Postgraduate Research & Practice Innovation Program of Jiangsu Province (No. SJCX18_0846), and by the Graduate Innovation and Practice Program of colleges and universities in Jiangsu Province.

References

1. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. Int. J. Comput. Vis. **60**(2), 91–110 (2004)
2. Qiu, G.: Indexing chromatic and achromatic patterns for content-based colour image retrieval. Pattern Recogn. **35**(8), 1675–1686 (2002)
3. Smeulders, A.W.M., Worring, M., Santini, S., Gupta, A., Jain, R.: Content-based image retrieval at the end of the early years. IEEE Trans. Pattern Anal. Mach. Intell. **22**(12), 1349–1380 (2000)
4. Wan, J., et al.: Deep learning for content-based image retrieval: a comprehensive study. In: Proceedings of the 22nd ACM International Conference on Multimedia, pp. 157–166. ACM (2014)
5. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems, pp. 1097–1105 (2012)
6. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 580–587 (2014)

7. Oquab, M., Bottou, L., Laptev, I., Sivic, J.: Learning and transferring mid-level image representations using convolutional neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1717–1724 (2014)
8. Razavian, A.S., Azizpour, H., Sullivan, J., Carlsson, S.: CNN features off-the-shelf: an astounding baseline for recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 806–813 (2014)
9. Bay, H., Tuytelaars, T., Van Gool, L.: SURF: speeded up robust features. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006. LNCS, vol. 3951, pp. 404–417. Springer, Heidelberg (2006). https://doi.org/10.1007/11744023_32
10. Gionis, A., Indyk, P., Motwani, R.: Similarity search in high dimensions via hashing. In: International Conference on Very Large Data Bases, pp. 518–529. Morgan Kaufmann Publishers Inc. (1999)
11. Raginsky, M., Lazebnik, S.: Locality-sensitive binary codes from shift-invariant kernels. In: Advances in Neural Information Processing Systems, pp. 1509–1517 (2009)
12. Weiss, Y., Torralba, A., Fergus, R.: Spectral hashing. In: Advances in Neural Information Processing Systems, pp. 1753–1760 (2008)
13. Gong, Y., Lazebnik, S., Gordo, A., Perronnin, F.: Iterative quantization: a procrustean approach to learning binary codes for large-scale image retrieval. IEEE Trans. Pattern Anal. Mach. Intell. **35**(12), 2916–2929 (2013)
14. Kulis, B., Darrell, T.: Learning to hash with binary reconstructive embeddings. In: Advances in Neural Information Processing Systems, pp. 1042–1050 (2009)
15. Xia, R., Pan, Y., Lai, H., Liu, C., Yan, S.: Supervised hashing for image retrieval via image representation learning. In: AAAI (2014)
16. Lin, K., Yang, H.F., Hsiao, J.H., Chen, C.S.: Deep learning of binary hash codes for fast image retrieval. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 27–35 (2015)
17. Liu, W., Wang, J., Ji, R., Jiang, Y.G., Chang, S.F.: Supervised hashing with kernels. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2074–2081 (2012)
18. Lecun, Y., Cortes, C.: The MNIST database of handwritten digits. http://yann.lecun.com/exdb/mnist/ (2010)
19. Krizhevsky, A., Hinton, G.: Learning multiple layers of features from tiny images. Technical report 1 (4), p. 7. University of Toronto (2009)
20. Hellerstein, J.M.: Generalized search tree. In: Liu, L., Özsu, M.T. (eds.) Encyclopedia of Database Systems, pp. 1222–1224. Springer, Boston (2009). https://doi.org/10.1007/978-1-4899-7993-3