DEEP HASH LEARNING FOR EFFICIENT IMAGE RETRIEVAL

Xuchao Lu[⋆]*, Li Song*[⋆†]*, Rong Xie*[⋆†]*, Xiaokang Yang*[⋆†]*, Wenjun Zhang*[⋆†]

[⋆]Institute of Image Communication and Network Engineering,
Shanghai Jiao Tong University, Shanghai, China
[†]Future Medianet Innovation Center, Shanghai, China
E-mail: lxc1992, song_li, xierong, xkyang, zhangwenjun@sjtu.edu.cn

ABSTRACT

Hashing method is a widely used method for content-based image retrieval. For more complicated semantic similarity of images, supervised hashing methods based on hand-crafted features show its limitations. Convolutional neural network (CNN) has powerful automatic feature learning ability. For this reason, CNN based deep hashing methods outperform previous methods. In this paper, we propose a new deep supervised hashing method for efficient image retrieval. We design a novel deep network with a hash layer as the output layer. An algorithm is proposed to generate optimal target hash code for training. We perform point-wise training for simultaneous feature extracting and hash function learning. Experiments on standard image retrieval benchmarks show that our method outperforms other state-of-the-art methods including unsupervised, supervised and deep hashing methods.

Index Terms— Image Retrieval, CNN, Hashing Learning

1. INTRODUCTION

With growing demands for browsing, searching and retrieving images from a large image database, content-based image retrieval has become a hot topic. The hashing method is the most successful method, like [1], [2] and [3]. Images are mapped to compact binary hash codes according to extracted image features. Since nearby hash codes indicate similar contents, hashing methods retrieve images with similar hash codes to query image's. The binarized representation of images leads to great efficiency improvement on storage and computation compared to standard techniques operating in Euclidean space [4].

The critical part of existing hashing methods is the features they use to represent the image. Convolutional neural network (CNN) has proved its remarkable performance in tasks highly depending on feature extracting, like image classification, natural language processing, and video analysis. CNN based methods outperform previous leading methods in these areas, which shows that CNN can learn robust features representing the semantic information of images.

A very natural idea is to use deep learning for learning compact binary hash codes. Following semantic hashing [3], deep hashing methods using CNN show great performance in content-based image retrieval. In this paper, we propose a new supervised deep hashing method for compact hash code learning to perform content-based image retrieval. Our method is an end-to-end learning framework with three main steps. The first step is to generate optimal target hash code from point-wise label information. The second step is to learn image features and hash function simultaneously through the training process of carefully designed deep network. The third step is to map image pixels to compact binary codes through hash function and perform image retrieval. Our method reaches state-of-the-art performance on MNIST and CIFAR-10 datasets.

The rest of this paper is organized as follows: Section 2 discusses the related works of hashing methods. The details of our method are in Section 3. The experimental results are provided in Section 4. Finally, Section five gives the conclusion.

2. RELATED WORKS

Hashing methods include data-independent methods [1] and data-dependent methods [5][2][6][7][8]. Methods of the first category are proposed in earlier days. The most representative ones are locality-sensitive hashing (LSH) [1] and the variants of it. The hash function is not related to training data. Instead, they do random projections to map images into a feature space. The second category learns the hash function from the training data. Because of the extra information, data-dependent methods outperform data-independent ones.

Data-dependent methods can be further divided into unsupervised methods and supervised methods. Unsupervised methods include spectral hashing (SH) [5], iterative quantization (ITQ) [7], etc. These methods learn hash functions from unlabelled training sets. To deal with more complicated im-

This work is supported by NSFC (61671296, 61521062 and U1611461), the national key research and development program of China(BZ0300013).

978-1-5090-6067-2/17/$31.00

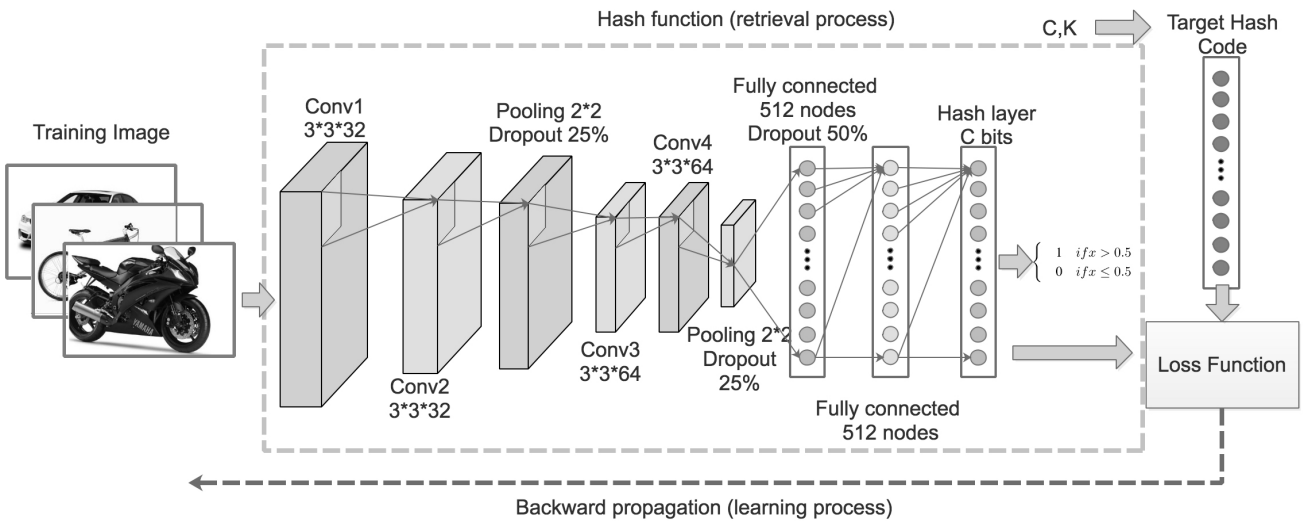

Fig. 1. The proposed framework for CIFAR-10 database. First, target hash code set is generated based on hash length C, image category number K. Then deep network is trained with raw images and target hash code. Finally, image retrieval is processed with the hash function, which is the trained network concatenated with a sgn function.

age database, supervised methods are proposed to learn better hash function from label information of training images. For example, supervised hashing with kernels (KSH) [2] requires a limited amount of supervised information and achieve high quality hashing. Minimal loss hashing (MLH) [6] is based on structured prediction with latent variables and a hinge-like loss function. And binary reconstructive embeddings (BRE) [8] develops an algorithm for learning hash functions based on explicitly minimizing the reconstruction error between the original distances and the Hamming distances.

Hashing methods mentioned above use hand-crafted features which are not powerful enough for more complicated semantic similarity. Moreover, the feature extracting procedure is independent of hash function learning. Recently, CNN based hashing methods called deep hashing methods are proposed to issue these problems. CNN can learn more representative features over hand-crafted features. Furthermore, most deep hashing methods perform feature learning and hash function learning simultaneously and show great improvement compared to previous methods. Several deep hashing methods have been proposed and proved to have better accuracy in content-based image retrieval. For example, CNNH [9] proposes a two-stage deep hashing method. It simultaneously learn features and hash functions based on learned approximate hash codes. Deep pairwise-supervised hashing (DPSH) [10] performs learning based on pairwise labels. [11] poses hash learning as a problem of regularized similarity learning and simultaneously learn hash function and image features through triplet samples. Our approach proposed in this paper outperforms the above methods.

3. PROPOSED METHOD

Given N training images $X = \{x_1, x_2, ..., x_N\}$ belonging to K categories, x_i is in the form of raw RGB values. Label information is noted as $L = \{l_1, l_2, ..., l_N\}$, $l_i \in \{1, 2, ..., K\}$. Our goal is to learn a function $H(x)$ mapping input images to compact binary codes $b_i = H(x_i)$, $b_i \in \{0, 1\}^C$, C stands for the hash length. The hash function $H(x)$ satisfies:

(1) b_i and b_j are similar in the Hamming space when $l_i = l_j$

(2) b_i and b_j are far away in the Hamming space when $l_i \neq l_j$

Figure 1 shows the proposed framework, we use CIFAR-10 database to illustrate. Our framework contains a CNN model as the main component. Normally the last layer of CNN is a Softmax classification layer. We replace it with a hash layer of C nodes for simultaneous learning of compact binary codes and image features. Since the output layer of CNN model has been changed, we need new output information to replace the labels. A target hash code generation component is proposed to generate optimal hash code for training. Finally, the hash function $H(x)$ is the trained model concatenated with a revised sgn function.

3.1. Target hash code generation

We replace the last layer of original CNN classification model with a fully connected layer called hash layer which has C nodes. We believe that the last layer contains the most compact information and hash code should be derived directly from this layer. The key point is to design a good target hash

Algorithm 1 Optimal hash code generation

Input: binary code length C, number of categories K

Output: code set $\{m_1, m_2, ..., m_K\}, m_i \in \{0,1\}^C$ satisfies $min(Hamming(m_i, m_j)) \geq H$

```
codeset.add(0)
for (i=1,2,...,2^C − 1) do
    flag=0
    for j=codeset[0],codeset[1]... do
        if Hamming(i,j)<H then
            flag=1
            break
    if flag==0 then
        codeset.add(i)
return codeset
```

Repeat: Perform the algorithm with $H = 1, 2, 3...$ until the length of code set is larger than K. Choose K codewords from the code set with largest H, that will be the target hash code set.

code set for the training of the whole network. The target hash code generation component is for this purpose.

Some models like [12] add a C-node latent layer before the CNN's last layer to generate hash codes. The good part is that the label information can be directly used for training. However, the learned weights after the latent layer become redundant. Furthermore, the target hash code generation component makes our learning a point-wise manner. We require no pair-wise inputs like [9], and the training speed is much faster.

Since the training images are in K categories, our target is to find a binary code set with K codewords. The minimum hamming distance between any two codewords should be as large as possible. In a more specific way, given binary code length C and codeword number K, we want to find a code set $M = \{m_1, m_2, ..., m_K\}, m_i \in \{0,1\}^C$, whose minimum Hamming distance is maximized. This optimization problem can first be divided to smaller jobs: given code length C and minimum Hamming distance H, find a code set with more than K code words. After that, repeat this process with larger H until no code set can be found. The last solvable H is the maximized minimum Hamming distance. The whole process is described in algorithm 1. Please note that this optimization problem is a complicated problem with no fixed result. For different C and H, the scale of code set may not be a certain number [13]. We have proved that our algorithm is able to at least find a second optimal solution. For example, consider a 24-bit code set for a 12-category dataset, the best solution is a code set whose minimum Hamming distance is 13 bits. Our algorithm will find one with the minimum Hamming distance of 12 bits.

For instance, given code length $C = 12$ for a dataset with $K = 10$ categories. With our algorithm, the minimum hamming distance $H = 6$ results in code set M with 16 codewords. We further try $H = 7$ and it results in a set with 4 codewords, which fails to meet our need. Then we randomly choose 10 codewords from the former set $M(C = 12, H = 6)$ and the target hash code is constructed as table 1 shows.

Table 1. Target 12-bit hash code set for a 10-category dataset

Label	decimal code	target hash code
1	0	000000000000
2	63	000000111111
3	455	000111000111
4	504	000111111000
5	1611	011001001011
6	1652	011001110100
7	1932	011110001100
8	1971	011110110011
9	2709	101010010101
10	2730	101010101010

3.2. Learning hash function

With the label information $L = \{l_1, l_2, ..., l_N\}$ and target hash code set $M = \{m_1, m_2, ..., m_K\}$, we can construct our new training set T with N training samples, x_i is raw RGB value of training images and m_{l_i} is the target hash code:

$$T = \{(x_1, m_{l_1}), (x_2, m_{l_2}), ..., (x_N, m_{l_N})\}$$

CNN has the powerful ability to learn image features through the concatenation of convolution layer and fully connected layer. Take CIFAR-10 training as an example, we adopt a widely used simple CNN model for CIFAR-10 for fast retrieval. As shown in figure 1, we use 32,32,64,64 3×3 convolution kernels for the convolution layers. 2×2 maxpooling and 25% dropout are added after 2nd and 4th convolution layer. Following convolution layers are two fully connected layers with 512 nodes and a 50% dropout after the first one. All these layers are activated with ReLU function for adding non-linearity. The hash layer is a fully connected layer with C nodes, depending on the length of target hash code. For larger C, the network can be trained to learn more features from the input image and lead to better performance. Each node implies a hidden feature of the input image. Sigmoid function ranges in $(0, 1)$ and for most cases lies in $(0, 0.1) \bigcup (0.9, 1)$. It is very suitable for indexing the output to binary codes.

Target hash code includes all the information needed to learn features from images, so loss function need not specially designed, simple mean squared error (MSE) loss function works well. For training optimizer, we choose ADADELTA for its good balance in speed and convergence point. Without huge modifications on the CNN model, our proposed model can learn a robust hush function fast in hundreds of training epochs.

3.3. Image retrieval

Our trained network accepts an input image x in raw pixels and gives an output $y \in (0,1)^C$. To convert output y to binary hash codes $h \in \{0,1\}^C$, we redesign the sgn function:

$$sgn = \begin{cases} 1 & if \quad x > 0.5 \\ 0 & if \quad x \leq 0.5 \end{cases}$$

Finally, we get our hash function:

$$H(x) = sgn(y) = h$$

where y is the output of our proposed model.

For image retrieval, we regard training images as image database and test images as query images. Image retrieval process searches top A most similar images from the database. Three steps are performed to do the image retrieval.

Step 1. Map N training images to hash codes $H_{db} = \{H(x_1), H(x_2), ..., H(x_N)\} = \{h_1, h_2, ..., h_N\}$.

Step 2. For each query image x_q, first calculate $h_q = H(x_q)$, then retrieve A images by the rank of the Hamming distance (h_q, x_i).

Step 3. Compare the similarity of retrieved images and the query image. Then evaluate the performance according to the result.

4. EXPERIMENTS

In this part, we state our experiment settings and the compared results with several state-of-the-art hashing methods on two widely-used datasets MNIST and CIFAR10. Please note that pre-trained AlexNet architecture [14] could have more powerful feature extraction because of deeper layers and more kernels. However, CNN architecture is not the focus of our research, so we adopt relatively simple network for each dataset for fast training and easy implementation.

4.1. Results on MNIST

The MNIST dataset consists 70000 28×28 gray scale images belonging to 10 categories of handwritten Arabic numerals from 0 to 9.

For MNIST, we use 32,32 3×3 convolution kernels for the convolution layers. 2×2 maxpooling and 25% dropout are added after the 2nd convolution layer. Following convolution layers are two fully connected layer with 128 nodes and a 50% dropout after the first fully connected layer. The last layer is the hash layer and the number of nodes is adjustable with hash length. Training the model uses Adadelta optimizer with mean squared error loss function. The whole training process lasts around 120s for 100 epochs training on a GTX1060 6GB graphic processing unit.

Our proposed method is compared to state-of-the-art hashing methods including data independant method LSH [1], two unsupervised methods SH [5], ITQ [7], four supervised methods KSH [2], MLH[6], BRE[8], and ITQ-CCA[7], and CNN based deep hashing method CNNH[9] and its variant CNNH+[9].

Fig. 2. Top 12 retrieved images of two query image samples. The dataset is CIFAR-10 and hash length is 24 bits. As shown in the figure, the precision of high-rank retrieved images is very high.

We follow the experiment configurations of [9] and derive results from the same resource. We randomly select 100 images per class and total 1000 images as test query images. For the unsupervised methods, use all the rest images as training set. And for supervised methods include CNNH, CNNH+ and ours, select 5000 images (500 images per class) as the training set.

To evaluate the performance of retrieval, we use Mean Average Precision (MAP). For each query image, we calculate the average precision of retrieved images. MAP is the mean value of these average precisions. Please note that, for each query image, the correctness of high ranking retrieved image counts more. The MAP result of our test is shown in table 2. We can find that our method outperforms other methods in gray-scale image retrieval.

4.2. Results on CIFAR-10

CIFAR-10 dataset consists of 60000 32×32 images belonging to 10 categories including airplane, automobile, bird etc.The layer information of CIFAR-10 implementation is stated in section 3.2. The whole training process lasts around 2 hours for 300 epochs training on a GTX1060 6GB graphic processing unit.

We compare our method with two more CNN based deep hash methods DHN [15] and DNNH [16], and we also follow their experiment configuration. We randomly select 100 images per class as query set. For the unsupervised methods, use all the rest images as training set. For supervised methods, 5000 images (500 images per class) are randomly selected as the training set. Two images are considered to be similar if they have the same label. The top 12 retrieved

Table 2. MAP of image retrieval on MNIST dataset with 4 different bit length. We use 1000 query images and calculate the MAP within top 5000 returned neighbors.

Method	MNIST(MAP)			
	12bits	24bits	32bits	48bits
Ours	**0.980**	**0.984**	**0.984**	**0.990**
CNNH+	0.969	0.975	0.971	0.975
CNNH	0.957	0.963	0.956	0.960
KSH	0.872	0.891	0.897	0.900
ITQ-CCA	0.659	0.694	0.714	0.726
MLH	0.472	0.666	0.652	0.654
BRE	0.515	0.593	0.613	0.634
SH	0.265	0.267	0.259	0.250
ITQ	0.388	0.436	0.422	0.429
LSH	0.187	0.209	0.235	0.243

images of two query images are shown in figure 2 as an illustraion. The MAP results are in table 3. We can see that our method is better than other methods including unsupervised methods, supervised methods and deep methods with feature learning.

Table 3. MAP of image retrieval on CIFAR-10 dataset with 4 different bit length. We use 1000 query images and calculate the MAP within top 5000 returned neighbors.

Method	CIFAR-10(MAP)			
	12bits	24bits	32bits	48bits
Ours	**0.612**	**0.648**	**0.658**	**0.680**
DHN	0.555	0.594	0.603	0.621
DNNH	0.552	0.566	0.558	0.581
CNNH+	0.465	0.521	0.521	0.532
CNNH	0.439	0.511	0.509	0.522
KSH	0.303	0.337	0.346	0.356
ITQ-CCA	0.264	0.282	0.288	0.295
MLH	0.182	0.195	0.207	0.211
BRE	0.159	0.181	0.193	0.196
SH	0.131	0.135	0.133	0.130
ITQ	0.162	0.169	0.172	0.175
LSH	0.121	0.126	0.120	0.120

Following [10], we further do comparison to some deep hashing methods with different experiment settings including DSCH, DRSCH [11], DSRH [17] and DPSH [10], the results are directly derived from [10]. More specifically, we use 10000 test images as query set and 50000 images as training set. The MAP values are calculated according to top 50000 returned neighbors and are shown in table 4. We can find out that our method still leads the MAP results.

Table 4. MAP of image retrieval on CIFAR-10 dataset with 4 different bit length. We use 10000 query images and calculate the MAP within top 50000 returned neighbors.

Method	CIFAR-10(MAP)			
	16bits	24bits	32bits	48bits
Ours	**0.822**	**0.821**	**0.833**	**0.862**
DPSH	0.763	0.781	0.795	0.807
DRSCH	0.615	0.622	0.629	0.631
DSCH	0.609	0.613	0.617	0.620
DSRH	0.608	0.611	0.617	0.618

5. CONCLUSION

In this paper, we present a novel end-to-end hash learning network. We first propose an algorithm to form an optimal target hash code set with given bit length and number of categories. The network is redesigned based on CNN classification model. It is trained with raw images and generated target hash codes. Our training can be carried out very fast because of the point-wise training behavior. Experiment results on standard image retrieval benchmarks show that our method outperforms the state-of-the-art.

6. REFERENCES

[1] Mayur Datar, Nicole Immorlica, Piotr Indyk, and Vahab S Mirrokni, "Locality-sensitive hashing scheme based on p-stable distributions," in *Proceedings of the twentieth annual symposium on Computational geometry*. ACM, 2004, pp. 253–262.

[2] Wei Liu, Jun Wang, Rongrong Ji, Yu-Gang Jiang, and Shih-Fu Chang, "Supervised hashing with kernels," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012, pp. 2074–2081.

[3] Ruslan Salakhutdinov and Geoffrey Hinton, "Semantic hashing," *International Journal of Approximate Reasoning*, vol. 50, no. 7, pp. 969–978, 2009.

[4] Wei Zhang, Xiaochun Cao, Rui Wang, Yuanfang Guo, and Zhineng Chen, "Binarized mode seeking for scalable visual pattern discovery," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[5] Yair Weiss, Antonio Torralba, and Rob Fergus, "Spectral hashing," in *Advances in neural information processing systems*, 2009, pp. 1753–1760.

[6] Mohammad Norouzi and David M Blei, "Minimal loss hashing for compact binary codes," in *Proceedings of the 28th international conference on machine learning (ICML-11)*, 2011, pp. 353–360.

[7] Yunchao Gong, Svetlana Lazebnik, Albert Gordo, and Florent Perronnin, "Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 12, pp. 2916–2929, 2013.

[8] Brian Kulis and Trevor Darrell, "Learning to hash with binary reconstructive embeddings," in *Advances in neural information processing systems*, 2009, pp. 1042–1050.

[9] Rongkai Xia, Yan Pan, Hanjiang Lai, Cong Liu, and Shuicheng Yan, "Supervised hashing for image retrieval via image representation learning.," in *AAAI*, 2014, vol. 1, p. 2.

[10] Wu-Jun Li, Sheng Wang, and Wang-Cheng Kang, "Feature learning based deep supervised hashing with pairwise labels," *arXiv preprint arXiv:1511.03855*, 2015.

[11] Ruimao Zhang, Liang Lin, Rui Zhang, Wangmeng Zuo, and Lei Zhang, "Bit-scalable deep hashing with regularized similarity learning for image retrieval and person re-identification," *IEEE Transactions on Image Processing*, vol. 24, no. 12, pp. 4766–4779, 2015.

[12] Kevin Lin, Huei-Fang Yang, Jen-Hao Hsiao, and Chu-Song Chen, "Deep learning of binary hash codes for fast image retrieval," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2015, pp. 27–35.

[13] Morris Plotkin, "Binary codes with specified minimum distance," *IRE Transactions on Information Theory*, vol. 6, no. 4, pp. 445–450, 1960.

[14] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[15] Han Zhu, Mingsheng Long, Jianmin Wang, and Yue Cao, "Deep hashing network for efficient similarity retrieval.," in *AAAI*, 2016, pp. 2415–2421.

[16] Hanjiang Lai, Yan Pan, Ye Liu, and Shuicheng Yan, "Simultaneous feature learning and hash coding with deep neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3270–3278.

[17] Fang Zhao, Yongzhen Huang, Liang Wang, and Tieniu Tan, "Deep semantic ranking based hashing for multi-label image retrieval," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1556–1564.