Deep Self-taught Hashing for Image Retrieval

Ke Zhou
Huazhong University of Science and Technology
k.zhou@hust.edu.cn

Yu Liu
Huazhong University of Science and Technology
lightyear416@163.com

Jinkuan Song
University of Trento
jingkuan.song@unitn.it

Linyu Yan
Hubei University of Technology
yanranyaya@hust.edu.cn

Fuhao Zou
Huazhong University of Science and Technology
fuhao_zou@hust.edu.cn

Fumin Shen
University of Electronic Science and Technology of China
fumin.shen@gmail.com

ABSTRACT

Hashing is a promising technique to tackle the problem of scalable retrieval, and it generally consists two major components, namely hash code generation and hash functions learning. The majority of existing hashing fall under the shallow model, which is intrinsically weak on mining robust visual features and learning complicated hash functions. In view of the superiority of deep structure, especially the Convolutional Neural Networks (CNNs), on extracting high level representation, we propose a deep self-taught hashing (DSTH) framework to combine deep structures with hashing to improve the retrieval performance by automatically learning robust visual features and hash functions. By employing CNNs, more robust and discriminative features of the images can be extracted to benefit the hash codes generation. Then, we apply CNNs and Multilayer Perceptron under deep learning scheme to learn hash function in supervised process by using the generated hash codes as labels. The experimental results have shown that the DSTH is superior to several state-of-the-art algorithms.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval; I.5.2 [**Pattern Recognition**]: Design Methodology; classifier design and evaluation

Keywords

Data Hashing; Deep Learning; Self-taught; Convolutional Neural Networks

1. INTRODUCTION

Hashing is a promising technique in terms of similarity search over large scale dataset. Actually, it is a special way of dimensionality reduction, mapping high dimensional feature to compact hash code. Since the Hamming distance between two binary hash codes can be computed efficiently by using bit XOR operation and counting the number of non-zero bits, an ordinary PC today would be able to do millions of Hamming distance computation in just a few milliseconds. As a result, hashing shows incomparable superiority in fast similarity search. Currently, the research on hashing confronts two challenge: first, how to precisely extract representative image features; second, how to tackle the problem of semantic gap, leading to accurate transformation from image feature to hash code.

Aiming at ensuring the effectiveness of hashing, hash code should preserve the property of discrimination, i.e., objects with similar semantic should be mapped into similar hash codes and vise versa. Several data-aware hashing methods have been proposed by introducing the machine learning tricks into the field of hashing to enhance the effectiveness of hash codes [2, 5, 10]. Self-taught hashing (STH) [7] is proposed and considered as one of state-of-the-art works. However, it suffers from overfitting problem since the operations of generating hash codes for training data and hash function for testing data are independently handled, which leads to poor generalization ability. Minimal loss hashing (MLH) [8] have shown higher search accuracy than unsupervised hashing approaches, but they all impose difficult optimization and slow training mechanisms. Spectral hashing (SpH) [6] uses a separable Laplacian eigenfunction (LE) formulation that ends up assigning more bits to directions along which the data has a greater variance. However, this approach is somewhat heuristic and relies on an unrealistic assumption that the data is uniformly distributed in a high-dimensional rectangle. To summarize, the majority of existing hashing fall under the shallow model, which perform poor in discovering semantic information. The drawback derives from two aspects: (1) For feature extraction, existing hashing methods are mainly based on hand-crafted feature, such as Color Histogram, GIST, SIFT, BoW, etc. However, those features are limited in aspect of reflecting image semantic information, because they represent the semantical content in just one aspect, either global or local view. (2) For semantic preservation, shallow model intrinsically could not explore the high level semantic information contained in feature data.

To avoid the shortcoming of hashing method of shallow model, a few deep hashing method have been proposed. Comparing to shallow learning based hashing, deep learning

MM'15, October 26-30, 2015, Brisbane, Australia.
© 2015 ACM. ISBN 978-1-4503-3459-4/15/10 ...$15.00.
DOI: http://dx.doi.org/10.1145/2733373.2806320.

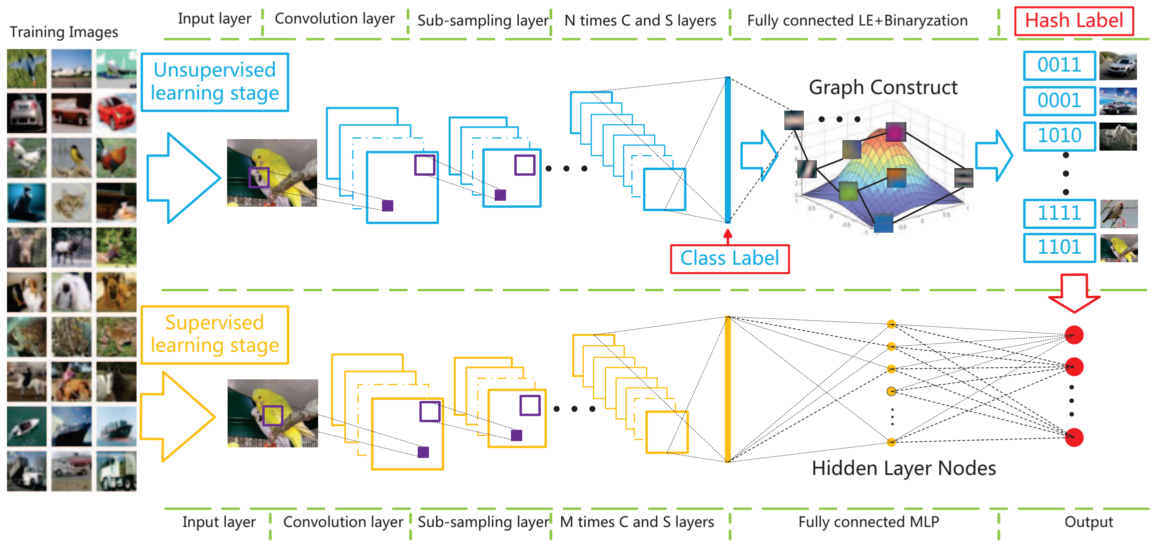

Figure 1: DSTH framework

has shown good performance on image applications. Convolutional neural networks(CNNs), have already achieved great success in various visual tasks such as image classification, retrieval and object detection [1] due to their powerful capability on mining high level semantic image representation. Through assuming the stationarity of statistics and the locality of pixel dependencies in images, CNNs have much fewer connections and parameters than standard deep neural networks, making it easier to train while the performance is nearly not affected. There are a few hashing methods that also use deep models. Given approximate hash codes learned from pairwise similarity matrix decomposition, Xia *et al.* [3] learn hash functions through using CNNs to fit the learned hash codes and class labels. Xia *et la.* [3] proposed a supervised hashing method, which firstly factorizes the pairwise semantic similarity matrix into approximate hash codes for the training images, and then trains a deep convolutional network with the approximate hash codes as well as the image tags. However, the in-sample hash codes are trained using the classification labels, which are coarse on measuring similarity among objects and makes the hashing model inaccurate for image representation.

Inspired by the successes of deep learning, in this paper, we explore deep learning techniques with application to image representation and establish a deep learning framework which learns hash functions by itself and yields accurate hash codes using extracted discriminative features. Meanwhile, we relate the hash value on each dimensional to representative features respectively. Although the framework seems similar to STH, we have two fundamental differences: (1) The image feature for training is automatically generated, while STH extracted hand-crafted visual features which do not necessarily preserve the accurate semantic relationship among images, degrading the performance of learning hashing model; (2) We learn hashing model by applying deep learning structure, while STH applied shallow SVM learning. The rest of this paper is organized as follows: In section 2, we illustrate our deep learning hashing framework. Extensive experimental results are presented in Section 3. Finally, we provide a conclusion in Section 4.

2. APPROACH

We propose an unsupervised and supervised learning algorithm under deep learning scheme, called deep self-taught hashing (DSTH), which is composed of unsupervised and supervised learning stages. The framework is shown in Fig. 1. In unsupervised learning stage, we adopt CNN and combination of LE and Binarization to learn hash labels with unsupervised constraints. In supervised learning stage, we use CNNs again with the hash labels acquired in the unsupervised learning stage, and apply Multi-layer Perceptron (MLP) to approximate acquired hash value. Finally, we get the hashing model learned from deep learning framework, aiming to efficiently map new images into hash codes for large scale image retrieval.

2.1 Unsupervised learning stage

In unsupervised learning stage, we mainly focus on generating hash labels. First, to capture discriminative fine features, we introduce CNNs to promote fine grained feature extraction with class labels. Second, we apply LE for dimension reducing and binaryzation for generating hash lables.

CNNs exploit spatially-local correlation by enforcing a local connectivity pattern between neurons of adjacent layers. Additionally, weight sharing increases learning efficiency by greatly reducing the number of free parameters to be learned. The constraints on the model enable CNNs to achieve better generalization on vision problems. In our algorithm, we adopt three layers, which contain convolution layer and subsampling layer respectively, to generate fine features.

After that, we use LE and binarization to transform the feature matrix consists of fully connected fine features to hash labels which retain relative distances among data from high dimension to low dimension. In practice, we collect n fully connected picture features as m-dimensional vectors $\{x_i\}_{i=1}^n \in \mathbb{R}^m$. We use x_i and y_i to represent ith sample and its hash label where $y_i \in \{-1, +1\}^l$ with l desired length of code. We set y_i^ρ is the ρ-th element of y_i where $+1$ if the ρ-th bit of code is on, or -1 otherwise.The hash code for the i-th image in n samples set can be represented as $[y_1, \dots, y_n]^T$.

An excellent semantic hash for image retrieval should preserve Hamming distance of mapped codes to protect feature similarity from high dimension to low dimension. This method is based on graph structure and we focus on the local similarity one [9]. Therefore, we apply k-Nearest Neighbor (KNN) algorithm to construct data graph. Our $n \times n$ local similarity matrix W is

$$W_{ij} = \begin{cases} 0 & \text{if } N_k(x_i, x_j) \text{ is false,} \\ \dfrac{x_i^T x_j}{\|x_i\| \cdot \|x_j\|} & \text{otherwise.} \end{cases} \tag{1}$$

where $N_k(x_i, x_j)$ represents the ith and the jth samples are neighbours in k-nearest set. Furthermore, we apply diagonal matrix

$$D_{ii} = \sum_{j=1}^{n} W_{ij} \tag{2}$$

Meanwhile, we use the number of different bits for calculating Hamming distance between y_i and y_j as

$$H_{ij} = \|y_i - y_j\|^2 / 4 \tag{3}$$

As in SpH, we define an object function ζ to minimize the weighted average Hamming distance.

$$\zeta = \sum_{i=1}^{n} \sum_{j=1}^{n} W_{ij} H_{ij} \tag{4}$$

To calculate ζ, we transform it to $\xi = tr(Y^T LY)/4$, where $L = D - W$ is Laplacian matrix and $tr(\cdot)$ means trace of matrix. The last, we transform ξ to LapEig problem ψ with slacking constraint $y_i \in \{-1, +1\}^t$, and obtain the optimal t-dimensional real-valued vector $\tilde{y}$ to represent each sample. ψ is the following:

$$\psi = arg \min_{\tilde{Y}} \quad Tr(\tilde{Y^T} D\tilde{Y}) \quad s.t. \begin{cases} \tilde{Y^T} D\tilde{Y} = I \\ \tilde{Y^T} D1 \ = 0 \end{cases} \tag{5}$$

where $Tr(\tilde{Y^T} L\tilde{Y})$ gives the real relaxation of the weighted average Hamming distance $Tr(T^T LY)$. The solution of this optimisation problem is given by $\tilde{Y} = [v_1, \ldots, v_t]$ whose columns are the t eigenvectors corresponding to the smallest eigenvalues of following generalised eigenvalue problem. The solution of ψ can be transformed to

$$Lv = \lambda Dv \tag{6}$$

where vector v are the t eigenvectors corresponding to the smallest eigenvalues (nonzero) of it.

Then, we convert the t-dimensional real-valued vectors $\tilde{y_1}, \ldots, \tilde{y_n}$ into binary codes according to threshold. We set ε to present default threshold and y_i^p equivalent to p-th element of $\tilde{y_i}$. The value of y_i^p is

$$y_i^p = \begin{cases} +1 & y_i^p \geqslant \varepsilon, \\ -1 & \text{otherwise.} \end{cases} \tag{7}$$

2.2 Supervised learning stage

In supervised learning stage, we mainly focus on using the hash labels acquired in the unsupervised learning stage to get hashing model. First, using hash labels acquired in unsupervised learning stage, we employ CNNs again to receive fine grained features.

Then, we take advantage of single-hidden-layer MLP, which is an artificial neural network (ANN) and consists of input layer, hidden layer and output layer. It can adjust input to approximate output according to change weights on different nodes in hidden layer.

Formally, we set a function $f : \mathbb{R}^I \to \mathbb{R}^O$, where I is the size of input vector x and O is the size of the output vector $f(x)$. The formulation is

$$f(x) = b^{(2)} + W^{(2)} h(b^{(1)} + W^{(1)} x) \tag{8}$$

where b is bias vector, W is weight matrix and $h(x)$ is function in hidden layer. In addition, the core of $h(x)$ is logistic function $sigmoid(\alpha) = 1/(1 + e^{-\alpha})$.

In the learning process, MLP converges according to perceptron rules. We set trainset $D = \{(I_1, O_1), \ldots, (I_m, O_m)\}$, where I_i denotes ith input and O_i denotes ith objective output. The perceptron function is

$$E(W) = -\sum_{I_i \in M} (W^T I_i) O_i \tag{9}$$

where M is a set of input vectors who were classified wrong. Moreover, this function subjects to

$$O_i = \begin{cases} +1 & W^T I_i \geqslant 0, \\ -1 & W^T I_i \leqslant 0. \end{cases} \tag{10}$$

Therefore, $E(W)$ is summary of positive numbers. If all input vectors were classified correctly, $E(W) = 0$. In the practice, we use upper bound of $E(W)$ and times of iteration to control MLP convergence.

The reason why we select single-hidden-layer MLP to continue learning hash label after CNNs because CNNs is also a kind of transformation model of MLP. Therefore, we can construct integrate deep learning framework of ANN on the multi-output condition. However, single-hidden-layer MLP has a disadvantage of slow convergence rate and easy to fall into local optimum with effect to final result from initial condition. In the experiment, we will reset some cases by default condition to ensure computing efficiency. Sometimes, if we may never obtain W satisfied the upper bound of $E(W)$, we will stop iteration by default iteration times.

3. EXPERIMENT

To evaluate the retrieving performance of our approach DSTH, we test our approach and compare it with SpH, STH, ITQ and CNNH on the MNIST digit dataset and STL-10 dataset in this section. We use precision-recall curve to measure the performance of our approach. In addition,we choose 10 samples respectively and randomly in each category to calculate *precison* and select mean of their values of *precison* as final result. Meanwhile,we apply $k = 25$ for retrieving original nearest neighbours.

3.1 MNIST Digits

MNIST is a handwriting image set which contains 60000 training images and 10000 test images, each of size 28×28 pixels, with a label from 0 to 9. It is used as baseline for measurement. We randomly select 30,000 images and corresponding labels for establishing CNN models, then apply combination of LE and binarization to generate hash labels. After that, the hash labels are used to tune the CNN models and learn MLP mapping matrix. Finally, we map 10,000 test images into hash codes using the learned hash model, and evaluate the accuracy of retrieval. Figure 2 shows the 64-bit

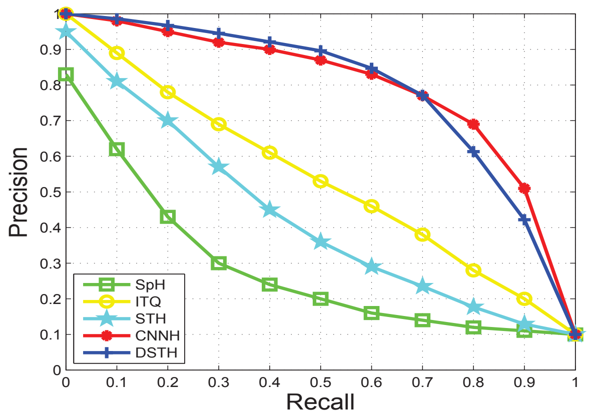

Figure 2: Precision-Recall curve of the comparative algorithm on MNIST with 64bits

codes retrieving accuracy of our approach DSTH compared to other algorithms.

From Figure 2, it is obvious that DSTH and CNNH are better than other algorithms on P-R curve. The difference between them is whether using CNNs to extract feature. Meanwhile, we find that curves of DSTH and CNNH are twined, and the former is better until recall achieves 0.7. When the recall exceeds 0.7, both curves decrease dramatically. The superiority of our approach mainly derives from the fine grained analysis on images, which yields fine grained image representation. However,due to low dimension of MNIST, the structure of deep learning is not deep enough to get excellent hash labels, making the retrieval results not good enough. Therefore, we select another image set STL-10 with high resolution to highlight the capability of our approach.

3.2 STL-10 dataset

STL-10 is a sub-set of ImageNet dataset for image recognition, which consist of 5000 training images (500 images per class), 8000 test images (800 images per class) and 100000 unlabeled images, each of size 96×96 pixels, in 10 classes. We select 10000 of 100000 unlabeled images randomly to obtain hash model and test. We calculate 48 bits for comparison. Figure 3 shows the quantitative evaluation of different approaches with 48 bits.

According to the Figure 3, it can be observed that DSTH performs best over the STL-10 dataset(which have large size images) and the second is CNNH. The gain of performance maybe stem from two reasons: (1) The visual feature generated by CNN performs better than hand-crafted feature; (2) The deep hash function have superior generation ability than SVM shallow learning[4]. For image retrieval, the larger the image is, the more accurate the image feature should be. In our scheme, we promote deep CNNs layer architecture in both unsupervised learning and supervised learning to mine the high level semantic features, using fine grained hash labels to learn hashing model for image representation. In addition, using hash labels acquired in unsupervised stage to tune the deep neuron net yields accurate hashing models, which map new image samples to be precise hash codes.

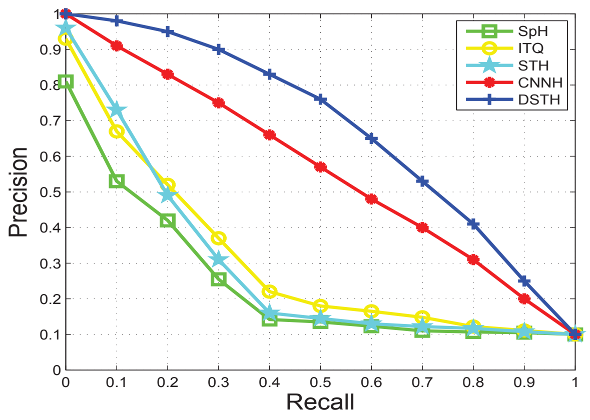

Figure 3: Precision-Recall curve of the comparative algorithm on STL-10 with 48bits

4. CONCLUSION

In this paper, we propose Deep Self-taught Hashing(DSTH) learning algorithm on deep learning framework and introduce hash labels derived from image feature into hashing model learning. First, using CNNs in unsupervised learning stage to generate hash labels successfully maximized the value of image itself, leading to much more accurate image representation than hand-crafted extracted feature used in shallow model. Second, in terms of both feature extraction and hash label generation, utilizing deep architecture simultaneously to promote the hash models leads to excellent performance on big size images dataset.

5. ACKNOWLEDGEMENT

Thanks for the funding supported by the National Basic Research Program (973 Program) of China under Grant 2011CB302305, National Natural Science Foundation of China under Grant 61232004.

6. REFERENCES

[1] J. Wang, Y. Song, T. Leung, C. Rosenberg, and Y. Wu. Learning fine-grained image similarity with deep ranking. CVPR, 2014.

[2] Y. Yang, Y. Yang, Z. Huang, H. Shen and F. Nie. Tag Localization with Spatial Correlations and Joint Group Sparsity. CVPR, 2011.

[3] R. Xia, Y. Pan, H. Lai, C. Liu, S. Yan. Supervised Hashing for Image Retrieval via Image Representation Learning. AAAI, 2014.

[4] Bengio Y. Learning deep architecture for AI. Foundations and Trends in Machine Learning, 2009, 2(1): 1-127.

[5] J. Song, Y. Yang, Z. Huang, H. T. Shen, and R. Hong. Multiple feature hashing for real-time large scale near-duplicate video retrieval. In *ACM Multimedia*, 2011.

[6] Y. Weiss, A. Torralba, R. Fergus, Spectral hashing, NIPS, 2008.

[7] D. Zhang, J. Wang, D. Cai, J. Lu, Self-taught hashing for fast similarity search, SIGIR, 2010.

[8] M. E. Norouzi, D. J. Fleet, Minimal loss hashing for compact binary codes, ICML, 2011.

[9] L. Gao, J. Song, F. Nie, Y. Yan, N. Sebe, and H. T. Shen. Optimal graph leaning with partial tags and multiple features. In *CVPR*, 2015.

[10] J. Wang, H. T. Shen, J. Song, and J. Ji. Hashing for similarity search: A survey. *CoRR*, abs/1408.2927, 2014.