This CVPR2015 extended abstract is the Open Access version, provided by the Computer Vision Foundation.

Simultaneous Feature Learning and Hash Coding with Deep Neural Networks

Hanjiang Lai[1], Yan Pan[2], Ye Liu[3], Shuicheng Yan[1]
[1]Department of Electronic and Computer Engineering, National University of Singapore, Singapore. [2]School of Software, Sun Yan-Sen University, China.
[3]School of Information Science and Technology, Sun Yan-Sen University, China.

In this paper, we focus on learning-based hashing, an emerging stream of hash methods that learn similarity-preserving hash functions to encode input data points (e.g., images) into binary codes.

Many learning-based hashing methods have been proposed, e.g., [3, 5]. The existing learning-based hashing methods can be categorized into unsupervised and supervised methods, based on whether supervised information (e.g., similarities or dissimilarities on data points) is involved. Compact bitwise representations are advantageous for improving the efficiency in both storage and search speed, particularly in big data applications. Compared to unsupervised methods, supervised methods usually embed the input data points into compact hash codes with fewer bits, with the help of supervised information.

In the pipelines of most existing hashing methods for images, each input image is firstly represented by a vector of traditional hand-crafted visual descriptors (e.g., GIST, HOG), followed by separate projection and quantization steps to encode this vector into a binary code. However, such fixed hand-crafted visual features may not be optimally compatible with the coding process. In other words, a pair of semantically similar/dissimilar images may not have feature vectors with relatively small/large Euclidean distance. Ideally, it is expected that an image feature representation can sufficiently preserve the image similarities, which can be learned during the hash learning process. Very recently, Xia *et al.* [5] proposed CNNH, a supervised hashing method in which the learning process is decomposed into a stage of learning approximate hash codes from the supervised information, followed by a stage of simultaneously learning hash functions and image representations based on the learned approximate hash codes. However, in this two-stage method, the learned approximate hash codes are used to guide the learning of the image representation, but the learned image representation cannot give feedback for learning better approximate hash codes. This one-way interaction thus still has limitations.

In this paper, we propose a "one-stage" supervised hashing method via a deep architecture that maps input images to binary codes. As shown in Figure , the proposed deep architecture has three building blocks: 1) shared stacked convolution layers to capture a useful image representation, 2) divide-and-encode modules to divide intermediate image features into multiple branches, with each branch corresponding to one hash bit, (3) a triplet ranking loss [4] designed to preserve relative similarities.

Triplet Ranking Loss and Optimization In the proposed deep architecture, we propose to use a variant of the triplet ranking loss in [4] to preserve the relative similarities of images. Specifically, given the training triplets of images in the form of (I, I^+, I^-) in which I is more similar to I^+ than to I^-, the goal is to find a mapping $\mathcal{F}(.)$ such that the binary code $\mathcal{F}(I)$ is closer to $\mathcal{F}(I^+)$ than to $\mathcal{F}(I^-)$. Accordingly, the triplet ranking hinge loss is defined by

$$\begin{aligned} &\ell_{triplet}(\mathcal{F}(I), \mathcal{F}(I^+), \mathcal{F}(I^-)) \\ =&\max(0, ||\mathcal{F}(I) - \mathcal{F}(I^+)||_2^2 - ||\mathcal{F}(I) - \mathcal{F}(I^-)||_2^2 + 1) \\ &s.t.\ \mathcal{F}(I),\ \mathcal{F}(I^+),\ \mathcal{F}(I^-) \in [0,1]^q. \end{aligned} \tag{1}$$

Shared Sub-Network with Stacked Convolution Layers With this modified triplet ranking loss function (1), the input to the proposed deep architecture are triplets of images, i.e., $\{(I_i, I_i^+, I_i^-)\}_{i=1}^n$, in which I_i is more similar to I_i^+ than to I_i^- $(i = 1, 2, ...n)$. As shown in Figure , we propose to use a shared sub-network with a stack of convolution layers to automatically learn a unified representation of the input images. Through this sub-network, an input triplet (I, I^+, I^-) is encoded to a triplet of intermediate image features (x, x^+, x^-), where x, x^+, x^- are vectors with the same dimension.

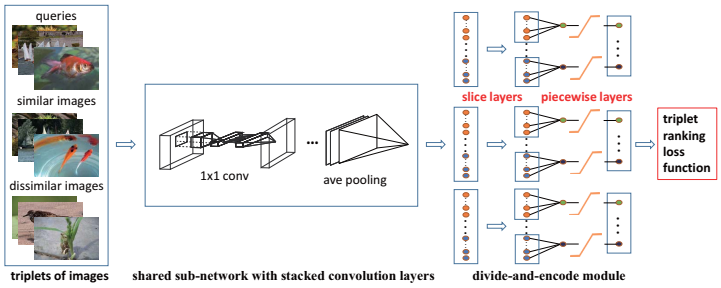

Figure 1: Overview of the proposed deep architecture for hashing. The input to the proposed architecture is in the form of triplets, i.e., (I, I^+, I^-) with a query image I being more similar to an image I^+ than to another image I^-. Through the proposed architecture, the image triplets are first encoded into a triplet of image feature vectors by a shared stack of multiple convolution layers. Then, each image feature vector in the triplet is converted to a hash code by a divide-and-encode module. After that, these hash codes are used in a triplet ranking loss that aims to preserve relative similarities on images.

In this sub-network, we adopt the architecture of Network in Network [2] as our basic framework, where we insert convolution layers with 1×1 filters after some convolution layers with filters of a larger receptive field. These 1×1 convolution filters can be regarded as a linear transformation of their input channels (followed by rectification non-linearity). As suggested in [2], we use an average-pooling layer as the output layer of this sub-network, to replace the fully-connected layer(s) used in traditional architectures (e.g., [1]).

Divide-and-Encode Module After obtaining intermediate image features from the shared sub-network with stacked convolution layers, we propose a divide-and-encode module to map these image features to approximate hash codes. We assume each target hash code has q bits. Then the outputs of the shared sub-network are designed to be $50q$. The proposed divide-and-encode module firstly divides the input intermediate features into q slices with equal length. Then each slice is mapped to one dimension by a fully-connected layer, followed by a sigmoid activation function that restricts the output value in the range $[0, 1]$, and a piece-wise threshold function to encourage the output of binary hash bits. After that, the q output hash bits are concatenated to be a q-bit (approximate) code.

[1] Alex Krizhevsky, Ilya Sutskever, and Geoff Hinton. Imagenet classification with deep convolutional neural networks. In *Proceedings of Advances in Neural Information Processing Systems*, pages 1106–1114, 2012.

[2] Min Lin, Qiang Chen, and Shuicheng Yan. Network in network. In *Proceedings of the International Conference on Learning Representations*, 2014.

[3] Wei Liu, Jun Wang, Rongrong Ji, Yu-Gang Jiang, and Shih-Fu Chang. Supervised hashing with kernels. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2074–2081, 2012.

[4] Mohammad Norouzi, David J. Fleet, and Ruslan Salakhutdinov. Hamming distance metric learning. In *Proceedings of the Advances in Neural Information Processing Systems*, pages 1–9, 2012.

[5] Rongkai Xia, Yan Pan, Hanjiang Lai, Cong Liu, and Shuicheng Yan. Supervised hashing for image retrieval via image representation learning. In *Proceedings of the AAAI Conference on Artificial Intellignece*, pages 2156–2162, 2014.

This is an extended abstract. The full paper is available at the Computer Vision Foundation webpage.