Deep Hashing Network for Efficient Similarity Retrieval*

Han Zhu, Mingsheng Long, Jianmin Wang and **Yue Cao**
School of Software, Tsinghua University, Beijing, China
Tsinghua National Laboratory for Information Science and Technology
{zhuhan10,caoyue10}@gmail.com {mingsheng,jimwang}@tsinghua.edu.cn

Abstract

Due to the storage and retrieval efficiency, hashing has been widely deployed to approximate nearest neighbor search for large-scale multimedia retrieval. Supervised hashing, which improves the quality of hash coding by exploiting the semantic similarity on data pairs, has received increasing attention recently. For most existing supervised hashing methods for image retrieval, an image is first represented as a vector of hand-crafted or machine-learned features, followed by another separate quantization step that generates binary codes. However, suboptimal hash coding may be produced, because the quantization error is not statistically minimized and the feature representation is not optimally compatible with the binary coding. In this paper, we propose a novel Deep Hashing Network (DHN) architecture for supervised hashing, in which we jointly learn good image representation tailored to hash coding and formally control the quantization error. The DHN model constitutes four key components: (1) a subnetwork with multiple convolution-pooling layers to capture image representations; (2) a fully-connected hashing layer to generate compact binary hash codes; (3) a pairwise cross-entropy loss layer for similarity-preserving learning; and (4) a pairwise quantization loss for controlling hashing quality. Extensive experiments on standard image retrieval datasets show the proposed DHN model yields substantial boosts over latest state-of-the-art hashing methods.

Introduction

While image big data with large volume and high dimension are pervasive in search engines and social networks, it has attracted increasing attention to enable approximate nearest neighbors (ANN) retrieval of images with both computation efficiency and search quality. An advantageous solution is hashing methods (Wang et al. 2014), which transform high-dimensional data into compact binary codes and generate similar binary codes for similar data items. In this paper, we focus on learning to hash methods that build data-dependent hash coding for efficient image retrieval, which have shown better performance than data-independent hashing methods, e.g. Locality-Sensitive Hashing (LSH) (Gionis et al. 1999).

Many learning to hash methods have been proposed to enable efficient ANN search using Hamming distance (Kulis and Darrell 2009; Gong and Lazebnik 2011; Norouzi and Blei 2011; Fleet, Punjani, and Norouzi 2012; Liu et al. 2012; Wang, Kumar, and Chang 2012; Liu et al. 2013; Gong et al. 2013; Yu et al. 2014; Xia et al. 2014; Zhang et al. 2014; Shen et al. 2015; Lai et al. 2015; Erin Liong et al. 2015). Hash learning can be divided into unsupervised methods and supervised methods. While unsupervised methods are more general and can be trained without semantic labels or relevances, they are restricted by the semantic gap dilemma (Smeulders et al. 2000) that high-level semantic description of an object often differs from low-level feature descriptors. Supervised methods can incorporate semantic labels or relevances to mitigate the semantic gap and improve the hashing quality, i.e. achieve accurate search with fewer bits of codes.

Recently, deep learning to hash methods (Xia et al. 2014; Lai et al. 2015) have shown that both feature representation and hash coding can be learned more effectively using deep neural networks (Krizhevsky, Sutskever, and Hinton 2012; Lin, Chen, and Yan 2014), which can naturally encode any nonlinear hashing functions. These deep hashing methods have created state-of-the-art results on many benchmarks. However, a crucial disadvantage of these deep learning to hash methods is that the quantization error is not statistically minimized hence the feature representation is not optimally compatible with binary hash coding. The continuous relaxation (Wang et al. 2014), i.e. solving the discrete optimization of hash codes by more viable continuous optimization, may give rise to two important issues widely ignored by previous learning to hash work: (1) uncontrollable quantization error by binarizing continuous embeddings to binary codes, and (2) large approximation error by adopting ordinary distance between continuous embeddings as the surrogate of Hamming distance between binary codes. Another potential limitation is that they do not adopt principled pairwise loss function to link the pairwise Hamming distances with the pairwise similarity labels, i.e. to classify whether a data pair is similar or dissimilar (pairwise classification) based on the pairwise Hamming distances. Therefore, suboptimal hash coding may be produced by existing deep hashing methods.

In this paper, we simultaneously control the quantization error and close the gap between Hamming distance and its approximate distance for learning high-quality hash codes. To approach this goal, we propose a novel Deep Hashing Network (DHN) architecture for supervised hashing using a

*Corresponding authors: Mingsheng Long and Jianmin Wang.

Bayesian framework. We jointly learn good image representation for hash coding and formally control the quantization error. The DHN model constitutes four key components: (1) a sub-network with multiple convolution-pooling layers to capture image representations; (2) a fully-connected hashing layer to generate compact binary hash codes; (3) a pairwise cross-entropy loss layer for similarity-preserving learning; and (4) a pairwise quantization loss for controlling hashing quality. Extensive experiments on standard image retrieval datasets show the proposed DHN model yields substantial improvements over current state-of-the-art hashing methods.

Related Work

Existing learning to hash methods can be categorized in two categories: unsupervised hashing and supervised hashing. Refer to (Wang et al. 2014) for a comprehensive survey.

Unsupervised hashing methods learn hash functions that can encode input data points to binary codes only using the unlabeled training data. Typical learning criteria include reconstruction error minimization (Salakhutdinov and Hinton 2007; Jegou, Douze, and Schmid 2011), neighborhood preserving as graph-based hashing (Weiss, Torralba, and Fergus 2009; Liu et al. 2011), and quantization error minimization as Iterative Quantization (ITQ) (Gong and Lazebnik 2011).

Supervised hashing explores supervised information (e.g., class labels, relative similarity, or relevance feedback) to learn compact hash coding. Binary Reconstruction Embedding (BRE) (Kulis and Darrell 2009) pursues hash functions by minimizing the squared errors between the distances of data points and the distances of corresponding hash codes. Minimal Loss Hashing (MLH) (Norouzi and Blei 2011) and Hamming Distance Metric Learning (Norouzi, Blei, and Salakhutdinov 2012) learn hash codes by minimizing hinge-like loss functions based on relative similarity of data points. Supervised Hashing with Kernels (KSH) (Liu et al. 2012) is a kernel-based method that builds compact binary codes by minimizing the Hamming distances on similar pairs and maximizing the Hamming distances on dissimilar pairs.

Recent revolution in deep learning shows that deep convolutional neural network (CNN) (Krizhevsky, Sutskever, and Hinton 2012; Lin, Chen, and Yan 2014; Bengio, Courville, and Vincent 2013) can automatically learn effective image representations that yield breakthrough performance on general computer vision tasks. Xia et al. proposed CNNH (Xia et al. 2014) that decomposes the hash learning process into a stage of learning approximate hash codes, followed by a deep-network-based stage of simultaneously fine-tuning the image features and hash functions. Lai et al. improved the two-stage CNNH by proposing DNNH (Lai et al. 2015), a simultaneous feature learning and hash coding deep network such that image representations and hash codes can improve each other in the joint learning process. DNNH has created the latest state-of-the-art results on many benchmarks.

In this work, we further improve DNNH by exploiting two key problems: (1) control the quantization error in a principled way, and (2) devise a more principled pairwise cross-entropy loss to link the pairwise Hamming distances with the pairwise similarity labels. These two improvements constitute the proposed Deep Hashing Network (DHN) approach.

Deep Hashing Network

In similarity retrieval, we are given a training set of N points $\{\boldsymbol{x}_i\}_{i=1}^N$, each represented as D-dimensional feature vector $\boldsymbol{x} \in \mathbb{R}^D$. Some pairs of points are associated with similarity labels s_{ij}, where $s_{ij} = 1$ implies $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are similar and $s_{ij} = 0$ indicates $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are dissimilar. Our goal is to learn nonlinear hashing function $f : \boldsymbol{x} \mapsto \boldsymbol{h} \in \{-1, 1\}^K$ to encode each point $\boldsymbol{x}$ in compact K-bit hash code $\boldsymbol{h} = f(\boldsymbol{x})$ such that the similarity between given pairs is preserved. In supervised hashing, $\mathcal{S} = \{s_{ij}\}$ is usually constructed from the semantic labels within the data points or the relevance feedback from click-through data in image retrieval systems.

In this paper, we propose a deep hashing network (DHN) architecture for hash learning, shown in Figure 1. This architecture accepts input images in a pairwise form $(\boldsymbol{x}_i, \boldsymbol{x}_j, s_{ij})$ and processes them through the deep hashing pipeline: (1) a sub-network with multiple convolution-pooling layers to extract image representations; (2) a fully-connected hashing layer to generate compact hash codes; (3) a pairwise cross-entropy loss layer for similarity-preserving learning; and (4) a pairwise quantization loss for controlling hashing quality.

Model Formulation

We start with AlexNet (Krizhevsky, Sutskever, and Hinton 2012), the deep convolutional neural network (CNN) comprised of five convolutional layers ($conv1$–$conv5$) and three fully connected layers ($fc6$–$fc8$). Each fc layer ℓ learns a nonlinear mapping $\boldsymbol{z}_i^\ell = a^\ell\left(\boldsymbol{W}^\ell \boldsymbol{z}_i^{\ell-1} + \boldsymbol{b}^\ell\right)$, where $\boldsymbol{z}_i^\ell$ is the ℓ-th layer hidden representation of point $\boldsymbol{x}_i$, $\boldsymbol{W}^\ell$ and $\boldsymbol{b}^\ell$ are the weight and bias parameters of the ℓ-th layer, and a^ℓ is the activation function, taken as rectifier units (ReLU) $a^\ell(\boldsymbol{x}) = \max(\boldsymbol{0}, \boldsymbol{x})$ for all hidden layers $conv1$–$fc7$. For hash function learning, we replace the $fc8$ layer of the softmax classifier in the original AlexNet with a new fch layer of K hidden units, which transforms the $fc7$ representation to K-dimensional hash coding by $\boldsymbol{h}_i = \boldsymbol{z}_i^l$, where $l = 8$ is the total number of layers and $\boldsymbol{z}_i^l$ is the hidden representation of the fch layer. To encourage the fch layer representation $\boldsymbol{z}_i^l$ to be binary codes, we first squash its output to be within $[-1, 1]$ by utilizing the hyperbolic tangent (tanh) activation $a^l(\boldsymbol{x}) = \tanh(\boldsymbol{x})$. To guarantee that the fch representation $\boldsymbol{z}_i^l$ will be good hash coding, we must preserve the similarity between given pairs in $\mathcal{S}$ and control the quantization error of binarizing the hidden representation into binary codes.

In this work, we jointly preserve the pairwise similarity and control the quantization error in a Bayesian framework. For a pair of binary codes $\boldsymbol{h}_i$ and $\boldsymbol{h}_j$, there exists a nice linear relationship between their Hamming distance $\mathrm{dist}_H(\cdot, \cdot)$ and inner product $\langle \cdot, \cdot \rangle$: $\mathrm{dist}_H\left(\boldsymbol{h}_i, \boldsymbol{h}_j\right) = \frac{1}{2}\left(K - \langle \boldsymbol{h}_i, \boldsymbol{h}_j \rangle\right)$. Hence in the sequel, we will use the inner product as a good surrogate of the Hamming distance to quantify the pairwise similarity. Given the pairwise similarity labels $\mathcal{S} = \{s_{ij}\}$, the logarithm Maximum a Posteriori (MAP) estimation of hash codes $\boldsymbol{H} = [\boldsymbol{h}_1, \ldots, \boldsymbol{h}_N]$ can be derived as follows,

$$\begin{aligned} \log p\left(\boldsymbol{H}|\mathcal{S}\right) &\propto \log p\left(\mathcal{S}|\boldsymbol{H}\right) p\left(\boldsymbol{H}\right) \\ &= \sum_{s_{ij} \in \mathcal{S}} \log p\left(s_{ij}|\boldsymbol{h}_i, \boldsymbol{h}_j\right) p\left(\boldsymbol{h}_i\right) p\left(\boldsymbol{h}_j\right), \end{aligned} \quad (1)$$

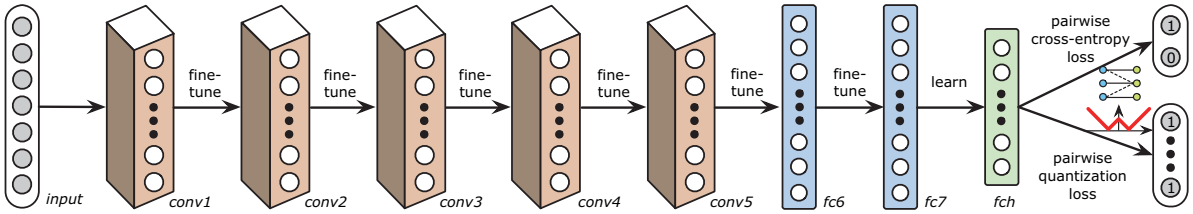


Figure 1: Deep Hashing Network (DHN) with a hash layer fch, a pairwise cross-entropy loss, and a pairwise quantization loss.

where $p(\mathcal{S}|\boldsymbol{H})$ is the likelihood function, and $p(\boldsymbol{H})$ is the prior distribution. For each pair, $p(s_{ij}|\boldsymbol{h}_i, \boldsymbol{h}_j)$ is the conditional probability of similarity label s_{ij} given hash codes $\boldsymbol{h}_i$ and $\boldsymbol{h}_j$, which is defined as the pairwise logistic function,

$$p\left(s_{ij}|\boldsymbol{h}_i, \boldsymbol{h}_j\right) = \begin{cases} \sigma\left(\left\langle \boldsymbol{z}_i^l, \boldsymbol{z}_j^l \right\rangle\right), & s_{ij} = 1 \\ 1 - \sigma\left(\left\langle \boldsymbol{z}_i^l, \boldsymbol{z}_j^l \right\rangle\right), & s_{ij} = 0 \end{cases} = \sigma\left(\left\langle \boldsymbol{z}_i^l, \boldsymbol{z}_j^l \right\rangle\right)^{s_{ij}} \left(1 - \sigma\left(\left\langle \boldsymbol{z}_i^l, \boldsymbol{z}_j^l \right\rangle\right)\right)^{1-s_{ij}} \tag{2}$$

where $\sigma(x) = \frac{1}{1+e^{-x}}$ is the sigmoid function and note that $\boldsymbol{h}_i = \boldsymbol{z}_i^l$. Similar to logistic regression, we can see that the smaller the Hamming distance $\mathrm{dist}_H(\boldsymbol{h}_i, \boldsymbol{h}_j)$ is, the larger the inner product $\langle \boldsymbol{h}_i, \boldsymbol{h}_j \rangle$ will be, and the larger $p(1|\boldsymbol{h}_i, \boldsymbol{h}_j)$ will be, implying that pair $\boldsymbol{h}_i$ and $\boldsymbol{h}_j$ should be classified as "similar"; otherwise, the larger $p(0|\boldsymbol{h}_i, \boldsymbol{h}_j)$ will be, implying that pair $\boldsymbol{h}_i$ and $\boldsymbol{h}_j$ should be classified as "dissimilar". Hence, Equation (2) is a reasonable extension of the logistic regression classifier to the pairwise classification scenario, which is optimal for binary similarity labels $s_{ij} \in \{0, 1\}$.

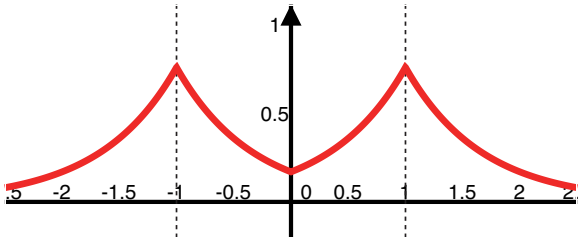


Figure 2: The bimodal Laplacian prior for quantization.

Since discrete optimization of Equation (1) with binary constraints $\boldsymbol{h}_i \in \{-1, 1\}^K$ is very challenging, for ease of optimization, continuous relaxation is applied to the binary constraints, as widely adopted by existing hashing methods (Wang et al. 2014). However, the continuous relaxation will give rise to two important issues widely ignored by previous learning to hash work: (1) uncontrollable quantization error by binarizing continuous embeddings to binary codes, and (2) large approximation error by adopting inner product between continuous embeddings as the surrogate of Hamming distance between binary codes. To control the quantization error and close the gap between Hamming distance and its surrogate for learning high-quality hash codes, in this paper, we propose a novel bimodal Laplacian prior (unnormalized)

双峰laplacian先验

for the continuous representations $\{\boldsymbol{h}_i\}$, which is defined as

$$p(\boldsymbol{h}_i) = \frac{1}{2\epsilon} \exp\left(-\frac{\||\boldsymbol{h}_i| - \mathbf{1}\|_1}{\epsilon}\right), \tag{3}$$

where ϵ is the diversity parameter, and an illustration of the proposed prior is shown in Figure 2. We can observe that the prior puts the largest density on the discrete values $\{-1, 1\}$, which enforces that the learned Hamming embeddings $\{\boldsymbol{h}_i\}$ are assigned to $\{-1, 1\}$ with the largest probability.

By taking Equations (2) and (3) into the MAP estimation in Equation (1), we achieve the DHN optimization problem:

$$\min_{\Theta} C = L + \lambda Q, \tag{4}$$

where $\lambda = 1/\epsilon$ is trade-off parameter between the pairwise cross-entropy loss L and the pairwise quantization loss Q, and $\Theta \triangleq \{\boldsymbol{W}^\ell, \boldsymbol{b}^\ell\}$ denotes the set of network parameters. Specifically, the pairwise cross-entropy loss L is defined as

$$L = \sum_{s_{ij} \in \mathcal{S}} \left(\log\left(1 + \exp\left(\left\langle \boldsymbol{z}_i^l, \boldsymbol{z}_j^l \right\rangle\right)\right) - s_{ij} \left\langle \boldsymbol{z}_i^l, \boldsymbol{z}_j^l \right\rangle\right). \tag{5}$$

Similarly, the pairwise quantization loss can be derived as

$$Q = \sum_{s_{ij} \in \mathcal{S}} \left(\left\||\boldsymbol{z}_i^l| - \mathbf{1}\right\|_1 + \left\||\boldsymbol{z}_j^l| - \mathbf{1}\right\|_1\right), \tag{6}$$

where $\mathbf{1} \in \mathbb{R}^K$ is the vector of ones. As Q is a non-smooth function whose derivative is difficult to compute, we adopt a smooth surrogate (Hyvärinen, Hurri, and Hoyer 2009) of the absolute function $|x| \approx \log\cosh x$, which reduces (6) to

$$Q = \sum_{s_{ij} \in \mathcal{S}} \sum_{k=1}^{K} \left(\log\cosh\left(\left|z_{ik}^l\right| - 1\right) + \log\cosh\left(\left|z_{jk}^l\right| - 1\right)\right). \tag{7}$$

By optimizing the MAP estimation in Equation (4), we can achieve statistically optimal learning of the hash codes, by jointly preserving the pairwise similarity in training data and controlling the quantization error of binarizing continuous embeddings to binary codes. Finally, we can obtain K-bit binary codes by simple quantization $\boldsymbol{h} \leftarrow \mathrm{sgn}(\boldsymbol{z}^l)$, where $\mathrm{sgn}(\boldsymbol{z}^l)$ is the sign function on vectors that for $i = 1, \ldots, K$, $\mathrm{sgn}(z_i^l) = 1$ if $z_i^l > 0$, otherwise $\mathrm{sgn}(z_i^l) = -1$. It is worth noting that, since we have minimized the quantization error in (4) during training, this final binarization step will incur very small loss of retrieval quality as validated empirically.

Learning Algorithm

We derive the learning algorithms for the DHN model in Equation (4), and show rigorously that both pairwise cross-entropy loss and pairwise quantization loss can be optimized efficiently through the standard back-propagation (BP) procedure. For notation brevity, we define the pairwise cost as

$$\begin{aligned} C_{ij} &\triangleq L_{ij} + \lambda Q_{ij} \\ &= \log\left(1+\exp\left(\left\langle \boldsymbol{z}_i^l, \boldsymbol{z}_j^l \right\rangle\right)\right) - s_{ij}\left\langle \boldsymbol{z}_i^l, \boldsymbol{z}_j^l \right\rangle \\ &+ \lambda \sum_{k=1}^{K} \left(\log\cosh\left(\left|z_{ik}^l\right|-1\right) + \log\cosh\left(\left|z_{jk}^l\right|-1\right)\right). \end{aligned} \tag{8}$$

Then we derive the gradient of point-wise cost C_i w.r.t. $\boldsymbol{W}_k^\ell$, the network parameter of the k-th unit in the ℓ-th layer as

$$\begin{aligned} \frac{\partial C_i}{\partial \boldsymbol{W}_k^\ell} &= 2\sum_{j:s_{ij}\in\mathcal{S}} \left(\frac{\partial L_{ij}}{\partial \boldsymbol{W}_k^\ell} + \lambda \frac{\partial Q_{ij}}{\partial \boldsymbol{W}_k^\ell}\right) \\ &= 2\sum_{j:s_{ij}\in\mathcal{S}} \left(\frac{\partial L_{ij}}{\partial \hat{z}_{ik}^\ell} + \lambda \frac{\partial Q_{ij}}{\partial \hat{z}_{ik}^\ell}\right) \frac{\partial \hat{z}_{ik}^\ell}{\partial \boldsymbol{W}_k^\ell} \\ &= 2\delta_{ik}^\ell \boldsymbol{z}_i^{\ell-1}, \end{aligned} \tag{9}$$

where $\hat{\boldsymbol{z}}_i^\ell = \boldsymbol{W}^\ell \boldsymbol{z}_i^{\ell-1} + \boldsymbol{b}^\ell$ is the output of the ℓ-th layer before activation $a^\ell(\cdot)$, and $\delta_{ik}^\ell \triangleq \sum_{j:s_{ij}\in\mathcal{S}} \left(\frac{\partial L_{ij}}{\partial \hat{z}_{ik}^\ell} + \lambda \frac{\partial Q_{ij}}{\partial \hat{z}_{ik}^\ell}\right)$ is the point-wise *residual* term that measures how much the k-th unit in the ℓ-th layer is responsible for the error of point $\boldsymbol{x}_i$ in the network output. For an output unit k, we can directly measure the difference between the network's activation and the true target value, and use it to define the residual δ_{ik}^l as

$$\begin{aligned} \delta_{ik}^l = &\sum_{j:s_{ij}\in\mathcal{S}} \left(\left[\sigma\left(\left\langle \boldsymbol{z}_i^l, \boldsymbol{z}_j^l \right\rangle\right) - s_{ij}\right] z_{jk}^l\right) \dot{a}^l\left(\hat{z}_{ik}^l\right) \\ &+ \lambda \sum_{j:s_{ij}\in\mathcal{S}} \tanh\left(\left|z_{ik}^l\right|-1\right) \operatorname{sgn}\left(z_{ik}^l\right) \dot{a}^l\left(\hat{z}_{ik}^l\right), \end{aligned} \tag{10}$$

where $l=8$ denotes the index of the output layer, and $\dot{a}^l(\cdot)$ is the derivative of the l-th layer activation function. For a hidden unit k in the $(\ell-1)$-th layer, we compute the residual $\delta_{ik}^{\ell-1}$ based on a weighted average of the errors of all the units $k'=1,\ldots,u_\ell$ in the ℓ-th layer that involve $\boldsymbol{z}_i^{\ell-1}$ as an input, which is just consistent with standard BP procedure,

$$\delta_{ik}^{\ell-1} = \left(\sum_{k'=1}^{u_\ell} \delta_{ik'}^\ell W_{k'k}^\ell\right) \dot{a}^{\ell-1}\left(\hat{z}_{ik}^{\ell-1}\right), \tag{11}$$

where u_ℓ is the number of hidden units in the ℓ-th layer. The residuals in all layers can be computed by back-propagation.

An important property of the proposed algorithm is that, only computing the residual of the output layer involves the pairwise summation as in Equation (10). For all hidden layers, all the residuals can be simply computed recursively by Equation (11), which does not involve pairwise summation. Hence we do not need to modify the implementation of BP in all hidden layers $1 \le \ell \le l-1$. We only need modify standard BP by replacing the output residual with Equation (10).

Since the only difference between standard BP and our algorithm is Equation (10), we analyze the computational complexity based on Equation (10). Denote the number of similarity pairs $\mathcal{S}$ available for training as $|\mathcal{S}|$, then it is easy to verify that the computational complexity is linear $O(|\mathcal{S}|)$. In practice, the similarity set $\mathcal{S}$ is very sparse, i.e. $|\mathcal{S}| \ll N^2$, hence we only need to optimize pairwise loss (5) by processing input pairs that are provided with similarity information.

Theoretical Analysis

We elaborate the connection between our work and Iterative Quantization (ITQ) (Gong and Lazebnik 2011), the seminal work that considers the quantization error in an iterative Procrustean procedure. In ITQ, point-wise quantization error is

$$Q_{\text{ITQ}} = \left\|\boldsymbol{h}_i - \operatorname{sgn}\left(\boldsymbol{h}_i\right)\right\|_2. \tag{12}$$

We only consider point-wise quantization error for brevity, while taking the summation for all training points (pairs) yields the overall quantization error similar to Equation (6).

Theorem 1 (Upper Bound)**.** *The proposed quantization loss* (6) *is an upper bound of the ITQ quantization error* (12)*,*

$$\left\|\boldsymbol{h}_i - \operatorname{sgn}\left(\boldsymbol{h}_i\right)\right\|_2 \le \left\||\boldsymbol{h}_i| - \mathbf{1}\right\|_1. \tag{13}$$

Proof. As $\boldsymbol{h}_i$ and $\operatorname{sgn}\left(\boldsymbol{h}_i\right)$ have the same sign, it yields that

$$\begin{aligned} \left\|\boldsymbol{h}_i - \operatorname{sgn}\left(\boldsymbol{h}_i\right)\right\|_2 &= \left\||\boldsymbol{h}_i| - \left|\operatorname{sgn}\left(\boldsymbol{h}_i\right)\right|\right\|_2 \\ &= \left\||\boldsymbol{h}_i| - \mathbf{1}\right\|_2 \\ &\le \left\||\boldsymbol{h}_i| - \mathbf{1}\right\|_1. \text{ (norm inequality)} \end{aligned} \tag{14}$$

□

Theorem 1 reveals that the proposed quantization loss in Equation (6) is a reasonable criterion for hash learning. The loss is easier to optimize in the back-propagation algorithm, since it does not involve a computation-intensive alternating optimization procedure as ITQ (Gong and Lazebnik 2011). Different from all existing methods, the proposed quantization loss is jointly optimized with the pairwise cross-entropy loss in a deep network, which yields better hashing schemes. An important advantage of the proposed loss is that the L_1-loss (6) may encourage sparsity, that is, more hash bits may be enforced to be $\{-1, 1\}$ compared with the L_2-loss (12), giving rise to more compact and discriminative hash codes.

Experiments

We conduct extensive experiments to evaluate the efficacy of the proposed DHN model against several state-of-the-art hashing methods on three widely-used benchmark datasets. The codes and configurations will be made available online.

Evaluation Setup

We conduct extensive empirical evaluation on three public benchmark datasets, **NUS-WIDE**, **CIFAR-10**, and **Flickr**.

- **NUS-WIDE**[1] is a public web image dataset. We follow the settings in (Liu et al. 2011; Lai et al. 2015) and use the subset of 195,834 images that are associated with the 21 most frequent concepts, where each concept consists of at least 5,000 images. We resize all images into 256×256.

[1] http://lms.comp.nus.edu.sg/research/NUS-WIDE.htm

Table 1: Mean Average Precision (MAP) of Hamming Ranking for Different Number of Bits on Three Image Datasets

Method	NUS-WIDE (MAP)				CIFAR-10 (MAP)				Flickr (MAP)			
	12 bits	24 bits	32 bits	48 bits	12 bits	24 bits	32 bits	48 bits	12 bits	24 bits	32 bits	48 bits
DHN	**0.708**	**0.735**	**0.748**	**0.758**	**0.555**	**0.594**	**0.603**	**0.621**	**0.810**	**0.828**	**0.829**	**0.841**
DNNH	0.674	0.697	0.713	0.715	0.552	0.566	0.558	0.581	0.783	0.789	0.791	0.802
CNNH⋆	0.617	0.663	0.657	0.688	0.484	0.476	0.472	0.489	0.749	0.761	0.768	0.776
CNNH	0.611	0.618	0.625	0.608	0.429	0.511	0.509	0.522	0.732	0.734	0.741	0.740
KSH	0.556	0.572	0.581	0.588	0.303	0.337	0.346	0.356	0.690	0.702	0.702	0.706
ITQ-CCA	0.435	0.435	0.435	0.435	0.264	0.282	0.288	0.295	0.513	0.531	0.540	0.555
MLH	0.500	0.514	0.520	0.522	0.182	0.195	0.207	0.211	0.610	0.618	0.629	0.634
BRE	0.485	0.525	0.530	0.544	0.159	0.181	0.193	0.196	0.571	0.592	0.599	0.604
SH	0.433	0.426	0.426	0.423	0.131	0.135	0.133	0.130	0.531	0.533	0.531	0.529
ITQ	0.452	0.468	0.472	0.477	0.162	0.169	0.172	0.175	0.544	0.555	0.560	0.570
LSH	0.403	0.421	0.426	0.441	0.121	0.126	0.120	0.120	0.499	0.513	0.521	0.548

- **CIFAR-10**[2] is a dataset containing 60,000 color images in 10 classes, and each class has 6,000 images in size 32×32.
- **Flickr**[3] consists of 25,000 images collected from Flickr, where each image is labeled with one of the 38 semantic concepts. We resize images of this subset into 256×256.

We follow the experimental protocols in (Lai et al. 2015). In NUS-WIDE and CIFAR-10, we randomly select 100 images per class as the test query set, and 500 images per class as the training set. In Flickr, we randomly select 1000 images as the test query set, and 4000 images as the training set. The similarity pairs for training are randomly constructed using image labels: each pair is considered similar (dissimilar) if they share at least one (none) semantic label.

We follow (Lai et al. 2015) to evaluate the retrieval quality based on four evaluation metrics: Mean Average Precision (MAP), Precision-Recall curves, Precision curves within Hamming distance 2, and Precision curves with respect to different numbers of top returned samples. For fair comparison, all of the methods use identical training and test sets.

We evaluate and compare, in several metrics, the retrieval quality of the proposed **DHN** approach with ten state-of-the-art hashing methods, including three unsupervised methods **LSH** (Gionis et al. 1999), **SH** (Weiss, Torralba, and Fergus 2009) and **ITQ** (Gong and Lazebnik 2011), and seven supervised methods **DNNH** (Lai et al. 2015), **CNNH** (Xia et al. 2014) and its variant **CNNH⋆** (Lai et al. 2015), **KSH** (Liu et al. 2012), **MLH** (Norouzi and Blei 2011), **BRE** (Kulis and Darrell 2009) and **ITQ-CCA** (Gong and Lazebnik 2011).

For the deep learning based methods, including CNNH, CNNH⋆, DNNH and DHN, we directly use the image pixels as input. For the shallow learning based methods, we follow (Liu et al. 2012; Lai et al. 2015; Srivastava and Salakhutdinov 2014) to represent each image in NUS-WIDE by a 500-dimensional bag-of-words vector, to represent each image in CIFAR-10 by a 512-dimensional GIST vector, and to represent each image in Flickr by a 3,857-dimensional vector concatenated by local SIFT feature, global GIST feature, etc. All image features are available at the datasets' website.

[2]http://www.cs.toronto.edu/kriz/cifar.html

[3]http://press.liacs.nl/mirflickr/

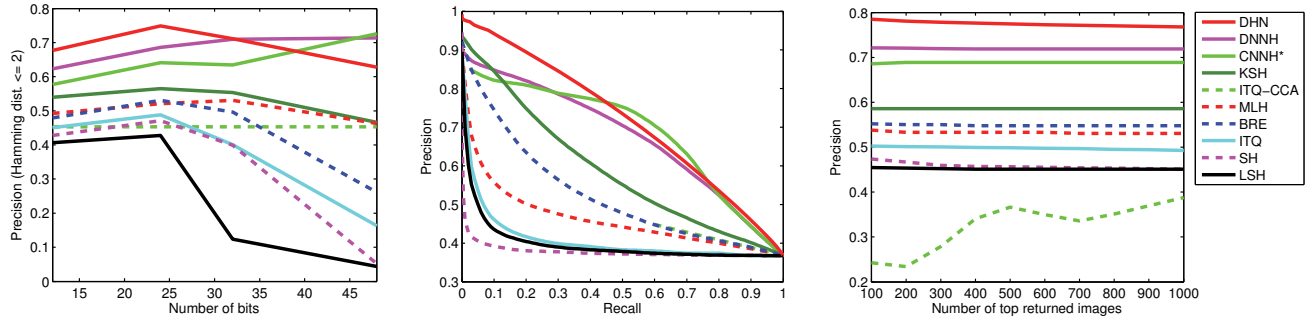
To guarantee that our results directly comparable to most published results, the results of LSH, BRE, ITQ, ITQ-CCA, KSH, MLH and SH on both the NUS-WIDE and CIFAR-10 datasets are directly reported from the latest work (Lai et al. 2015), while the results on the Flickr dataset are obtained by the implementations provided by their authors, following standard cross-validation procedures for model selection.

We implement the DHN model based on the open-source **Caffe** framework (Jia et al. 2014). We employ the AlexNet architecture (Krizhevsky, Sutskever, and Hinton 2012), fine-tune convolutional layers $conv1$–$conv5$ and fully-connected layers $fc6$–$fc7$ that were copied from the pre-trained model, and train hashing layer fch, all via back-propagation. As the fch layer is trained from scratch, we set its learning rate to be 10 times that of the lower layers. We use the mini-batch stochastic gradient descent (SGD) with 0.9 momentum and the learning rate annealing strategy implemented in Caffe, and cross-validate the learning rate from 10^{-5} to 10^{-2} with a multiplicative step-size 10. We choose the quantization penalty parameter λ by cross-validation from 10^{-5} to 10^{2} with a multiplicative step-size 10. We fix the mini-batch size of images as 64 and the weight decay parameter as 0.0005.

Results and Discussion

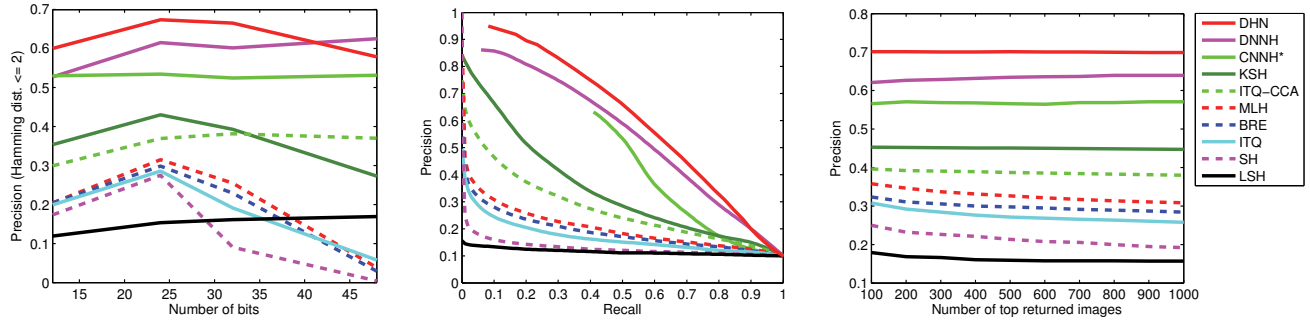
The MAP results of all methods are listed in Table 1, which show the proposed DHN method substantially outperforms all the comparison methods. Specifically, compared to the best baseline using traditional hand-crafted visual features, KSH, we achieve absolute increases of 16.3%, 25.8%, and 12.7% in average MAP for different bits on NUS-WIDE, CIFAR-10, and Flickr respectively. Compared to the state-of-the-art deep-network-based hashing method, DNNH, we achieve absolute increments of 3.8%, 2.9%, 3.6% in average MAP for different bits on the three datasets respectively. DNNH uses a fixed-margin loss and piecewise-linear activation function to train deep networks, which may cause information loss and objective oscillations in back propagation.

Figures 3 and 4 show the results of precision curves within Hamming radius 2, precision-recall curves with 48 bits, and precision curves with 48 bits w.r.t. different numbers of top returned images on NUS-WIDE and CIFAR-10, respectively. The proposed DHN approach generally outperforms



(a) Precision within Hamming radius 2 (b) Precision-recall curve @ 48 bits (c) Precisou curve w.r.t. top-n @ 48 bits

Figure 3: The results of comparison methods on the NUS-WIDE dataset: (a) precision curves within Hamming radius 2; (b) precision-recall curves of Hamming ranking with 48 bits; (c) precision curves w.r.t different numbers of top returned images.



(a) Precision within Hamming radius 2 (b) Precision-recall curve @ 48 bits (c) Precisou curve w.r.t. top-n @ 48 bits

Figure 4: The results of comparison methods on the CIFAR-10 dataset: (a) precision curves within Hamming radius 2; (b) precision-recall curves of Hamming ranking with 48 bits; (c) precision curves w.r.t different numbers of top returned images.

all comparison methods by large margins in the metrics of precision-recall curves of Hamming ranking (Figures 3(b) and 4(b)) and the precision curves w.r.t. different numbers of top returned images (Figures 3(c) and 4(c)). DHN achieves particularly good results at lower recall levels, which is very desirable for precision-oriented image retrieval systems.

The retrieval performance using Hamming ranking within Hamming radius 2 is particularly important for retrieval with binary hash codes, because such Hamming ranking only requires constant time cost. As shown in Figures 3(a) and 4(a), the proposed DHN approach achieves the highest precision within Hamming radius 2 with 24 bits in both datasets, and outperforms the state-of-the-art method, DNNH, when hash bits is no more than 32. In particular, the precision of DHN with 24 bits is better than that of DNNH with 48 bits. These results testify that DHN learns more compact binary codes such that shorter codes can already establish accurate results. It is worth noting that, when using longer hash codes, the Hamming space becomes increasingly sparse and very few data points fall within the Hamming ball with radius 2. This explains why DHN achieves the best performance with relatively shorter hash codes. DNNH mitigates the sparsity issue using a complicated divide-and-encode module to generate several pieces of independent hash codes and combine them. It is interesting to extend the DHN model using this module.

Empirical Analysis

We investigate several variants of DHN: DHN-B is the DHN variant without binarization ($\boldsymbol{h} \leftarrow \mathrm{sgn}(\boldsymbol{z}^l)$ not performed), which may serve as an upper bound of performance. DHN-Q is the DHN variant without the quantization loss ($\lambda = 0$). DHN-E is the DHN variant using widely-adopted pairwise squared loss $L = \sum_{s_{ij} \in \mathcal{S}} \left(s_{ij} - \frac{1}{K} \left\langle \boldsymbol{z}_i^l, \boldsymbol{z}_j^l \right\rangle\right)^2$ (Liu et al. 2012; Xia et al. 2014) instead of the pairwise cross-entropy loss (5). We compare the results of DHN variants in Table 2.

We can observe that, by optimizing the quantization loss (6), DHN incurs small average MAP decreases of 4.18%, 0.01%, and 2.29% when binarizing continuous embeddings of DHN-B. In contrast, without optimizing the quantization loss (6), DHN-Q suffers from very large MAP decreases of 10.8%, 4.20%, and 5.33%, and substantially underperforms DHN. These results validate that the proposed quantization loss (6) can effectively reduce the binarization error and lead to nearly lossless hash coding for high retrieval quality.

Another crucial observation is that, by using the pairwise cross-entropy loss (5), DHN outperforms DHN-E using the pairwise squared loss by very large margins of 9.07%, 8.01%, and 6.64% in average MAP for different bits. The

Table 2: Mean Average Precision (MAP) Results of DHN and Its Variants, DHN-B, DHN-Q, DHN-E on Three Datasets

Method	NUS-WIDE (MAP)				CIFAR-10 (MAP)				Flickr (MAP)			
	12 bits	24 bits	32 bits	48 bits	12 bits	24 bits	32 bits	48 bits	12 bits	24 bits	32 bits	48 bits
DHN-B	**0.760**	**0.779**	**0.788**	**0.789**	**0.606**	**0.599**	0.597	0.592	**0.842**	**0.850**	**0.851**	**0.856**
DHN	0.708	0.735	0.748	0.758	0.555	0.594	**0.603**	**0.620**	0.810	0.828	0.829	0.841
DHN-Q	0.632	0.667	0.683	0.703	0.532	0.551	0.574	0.569	0.784	0.797	0.801	0.804
DHN-E	0.611	0.643	0.664	0.670	0.485	0.512	0.521	0.535	0.751	0.764	0.763	0.766

pairwise squared loss has been widely adopted in previous work (Liu et al. 2012; Xia et al. 2014). However, this loss does not link well the pairwise Hamming distances (taking values in $(-\infty, +\infty)$ when using continuous relaxation) to the pairwise similarity labels (taking binary values {-1,1}). This is a misspecified use of pairwise regression for pairwise classification problems, which gives rise to large similarity search errors. The proposed pairwise cross-entropy loss (5) is derived rigorously from Bayesian learning framework (1) and is well-specified to the pairwise classification problem.

Conclusion

In this paper, we have formally approached the problem of supervised deep hashing in a Bayesian learning framework. The proposed Deep Hashing Network (DHN) architecture simultaneously optimizes the pairwise cross-entropy loss on semantic similarity pairs and the pairwise quantization loss on compact hash codes. Extensive experiments on standard image retrieval datasets show the DHN architecture yields substantial boosts over the state-of-the-art hashing methods.

Acknowledgments

This work was supported by National Natural Science Foundation of China (No. 61502265), National Natural Science Funds for Distinguished Young Scholars (No. 613250154), China Postdoctoral Science Foundation (No. 2015T80088), and Tsinghua National Laboratory (TNList) Special Funds for Big Data Science and Technology.

References

Bengio, Y.; Courville, A.; and Vincent, P. 2013. Representation learning: A review and new perspectives. *TPAMI* 35(8):1798–1828.

Erin Liong, V.; Lu, J.; Wang, G.; Moulin, P.; and Zhou, J. 2015. Deep hashing for compact binary codes learning. In *CVPR*, 2475–2483. IEEE.

Fleet, D. J.; Punjani, A.; and Norouzi, M. 2012. Fast search in hamming space with multi-index hashing. In *CVPR*. IEEE.

Gionis, A.; Indyk, P.; Motwani, R.; et al. 1999. Similarity search in high dimensions via hashing. In *VLDB*, volume 99, 518–529. ACM.

Gong, Y., and Lazebnik, S. 2011. Iterative quantization: A procrustean approach to learning binary codes. In *CVPR*, 817–824.

Gong, Y.; Kumar, S.; Rowley, H.; Lazebnik, S.; et al. 2013. Learning binary codes for high-dimensional data using bilinear projections. In *CVPR*, 484–491. IEEE.

Hyvärinen, A.; Hurri, J.; and Hoyer, P. O. 2009. *Natural Image Statistics: A Probabilistic Approach to Early Computational Vision.*, volume 39. Springer Science & Business Media.

Jegou, H.; Douze, M.; and Schmid, C. 2011. Product quantization for nearest neighbor search. *TPAMI* 33(1):117–128.

Jia, Y.; Shelhamer, E.; Donahue, J.; Karayev, S.; Long, J.; Girshick, R.; Guadarrama, S.; and Darrell, T. 2014. Caffe: Convolutional architecture for fast feature embedding. In *ACM Multimedia*. ACM.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. In *NIPS*.

Kulis, B., and Darrell, T. 2009. Learning to hash with binary reconstructive embeddings. In *NIPS*, 1042–1050.

Lai, H.; Pan, Y.; Liu, Y.; and Yan, S. 2015. Simultaneous feature learning and hash coding with deep neural networks. In *CVPR*. IEEE.

Lin, M.; Chen, Q.; and Yan, S. 2014. Network in network. In *ICLR, 2014 (arXiv:1409.1556)*.

Liu, W.; Wang, J.; Kumar, S.; and Chang, S.-F. 2011. Hashing with graphs. In *ICML*. ACM.

Liu, W.; Wang, J.; Ji, R.; Jiang, Y.-G.; and Chang, S.-F. 2012. Supervised hashing with kernels. In *CVPR*. IEEE.

Liu, X.; He, J.; Lang, B.; and Chang, S.-F. 2013. Hash bit selection: a unified solution for selection problems in hashing. In *CVPR*, 1570–1577. IEEE.

Norouzi, M., and Blei, D. M. 2011. Minimal loss hashing for compact binary codes. In *ICML*, 353–360. ACM.

Norouzi, M.; Blei, D. M.; and Salakhutdinov, R. R. 2012. Hamming distance metric learning. In *NIPS*, 1061–1069.

Salakhutdinov, R., and Hinton, G. E. 2007. Learning a nonlinear embedding by preserving class neighbourhood structure. In *AISTATS*, 412–419.

Shen, F.; Shen, C.; Liu, W.; and Tao Shen, H. 2015. Supervised discrete hashing. In *CVPR*. IEEE.

Smeulders, A. W.; Worring, M.; Santini, S.; Gupta, A.; and Jain, R. 2000. Content-based image retrieval at the end of the early years. *TPAMI* 22(12):1349–1380.

Srivastava, N., and Salakhutdinov, R. 2014. Multimodal learning with deep boltzmann machines. *JMLR* 15:2949–2980.

Wang, J.; Shen, H. T.; Song, J.; and Ji, J. 2014. Hashing for similarity search: A survey. *arXiv preprint arXiv:1408.2927*.

Wang, J.; Kumar, S.; and Chang, S.-F. 2012. Semi-supervised hashing for large-scale search. *TPAMI* 34(12):2393–2406.

Weiss, Y.; Torralba, A.; and Fergus, R. 2009. Spectral hashing. In *NIPS*.

Xia, R.; Pan, Y.; Lai, H.; Liu, C.; and Yan, S. 2014. Supervised hashing for image retrieval via image representation learning. In *AAAI*, 2156–2162. AAAI.

Yu, F. X.; Kumar, S.; Gong, Y.; and Chang, S.-F. 2014. Circulant binary embedding. In *ICML*, 353–360. ACM.

Zhang, P.; Zhang, W.; Li, W.-J.; and Guo, M. 2014. Supervised hashing with latent factor models. In *SIGIR*, 173–182. ACM.