Deep Cauchy Hashing for Hamming Space Retrieval

Yue Cao, Mingsheng Long*, Bin Liu, Jianmin Wang
KLiss, MOE; School of Software, Tsinghua University, China
National Engineering Laboratory for Big Data Software
Beijing Key Laboratory for Industrial Big Data System and Application
{caoyue10,liubinthss}@gmail.com, {mingsheng,jimwang}@tsinghua.edu.cn

Abstract

Due to its computation efficiency and retrieval quality, hashing has been widely applied to approximate nearest neighbor search for large-scale image retrieval, while deep hashing further improves the retrieval quality by end-to-end representation learning and hash coding. With compact hash codes, Hamming space retrieval enables the most efficient constant-time search that returns data points within a given Hamming radius to each query, by hash table lookups instead of linear scan. However, subject to the weak capability of concentrating relevant images to be within a small Hamming ball due to mis-specified loss functions, existing deep hashing methods may underperform for Hamming space retrieval. This work presents Deep Cauchy Hashing (DCH), a novel deep hashing model that generates compact and concentrated binary hash codes to enable efficient and effective Hamming space retrieval. The main idea is to design a pairwise cross-entropy loss based on Cauchy distribution, which penalizes significantly on similar image pairs with Hamming distance larger than the given Hamming radius threshold. Comprehensive experiments demonstrate that DCH can generate highly concentrated hash codes and yield state-of-the-art Hamming space retrieval performance on three datasets, NUS-WIDE, CIFAR-10, and MS-COCO.

1. Introduction

In the big data era, large-scale and high-dimensional media data has been pervasive in search engines and social networks. To guarantee retrieval quality and computation efficiency, approximate nearest neighbor (ANN) search has attracted increasing attention. Parallel to the traditional indexing methods [18] for candidates pruning, another advantageous solution is hashing methods [31] for data compression, which transform high-dimensional media data into compact binary codes and generate similar binary codes for similar data items. This paper will focus on the learning to hash methods [31] that build data-dependent hash encoding schemes for efficient image retrieval, which have shown better performance than the data-independent hashing methods, e.g. Locality-Sensitive Hashing (LSH) [10].

Many learning to hash methods have been proposed to enable efficient ANN search by Hamming ranking of compact hash codes, including both supervised and unsupervised methods [16, 12, 26, 9, 22, 30, 25, 24, 35]. Recently, deep learning to hash methods [33, 17, 28, 8, 36, 19, 21, 4] have shown that deep neural networks can enable end-to-end representation learning and hash coding with nonlinear hash functions. These deep learning to hash methods have shown state-of-the-art search performance. In particular, they prove it crucial to jointly learn similarity-preserving representations and control quantization error of converting continuous representations to binary codes [36, 19, 21, 4].

However, most of the existing methods are tailored to data compression instead of candidates pruning, i.e. they are designed to maximize retrieval performance based on linear scan of hash codes. As linear scan is still costly even using hash codes, we are somewhat deviating from our original intention with hashing, that is, to maximize speedup under acceptable accuracy. With the blossom of powerful hashing methods for linear scan, we should now turn to the Hamming space retrieval [9], which enables the most efficient constant-time search. In Hamming space retrieval, we return data points within a given Hamming radius to each query, by hash table lookups instead of linear scan. Unfortunately, existing hashing methods generally lack the capability of concentrating relevant images to be within a small Hamming ball due to the mis-specified loss functions, thus they may underperform for Hamming space retrieval.

This work presents Deep Cauchy Hashing (DCH), a novel deep hashing model that generates concentrated and compact hash codes to enable efficient and effective Hamming space retrieval. We propose a pairwise cross-entropy loss based on Cauchy distribution, which penalizes significantly on similar image pairs with Hamming distance larger than the given Hamming radius threshold. We further pro-

*Corresponding author: M. Long (mingsheng@tsinghua.edu.cn).

pose a quantization loss based on the Cauchy distribution, which enables learning nearly lossless hash codes. Both loss functions can be derived in a Bayesian learning framework and are well-specified to the Hamming space retrieval. The proposed DCH model can be trained end-to-end by back-propagation. Extensive experiments demonstrate that DCH can generate highly concentrated and compact hash codes and yield state-of-the-art image retrieval performance on three datasets, NUS-WIDE, CIFAR-10, and MS-COCO.

2. Related Work

Existing hashing methods [2, 16, 12, 26, 22, 30, 25, 11, 34, 35, 24] consist of unsupervised and supervised hashing. Please refer to [31] for a comprehensive survey.

Unsupervised hashing methods learn hash functions that encode data to binary codes by training from unlabeled data. Typical methods include reconstruction error minimization [27, 12, 14] and graph embedding [32, 23]. Supervised hashing further explores supervised information (e.g. pairwise similarity or relevance feedback) to generate discriminative and compact hash codes [16, 26, 22, 28]. Supervised Hashing with Kernels (KSH) [22] and Supervised Discrete Hashing (SDH) [28] generate nonlinear or discrete binary hash codes by minimizing (maximizing) the Hamming distances across similar (dissimilar) pairs of data points.

Recently, deep learning to hash methods [33, 17, 28, 8, 36, 1, 21, 4] yield breakthrough results on image retrieval datasets by blending the power of deep learning [15, 13]. In particular, DHN [36] is the first end-to-end framework that jointly preserves pairwise similarity and controls the quantization error. HashNet [4] improves DHN by balancing the positive and negative pairs in training data, and by continuation technique for lower quantization error, which obtains state-of-the-art performance on several benchmark datasets.

However, previous deep hashing methods perform unsatisfactorily for Hamming space retrieval [9], with early pruning to discard irrelevant data points out of Hamming Radius 2. The reason for inefficient Hamming space retrieval is that their loss functions penalize little when two similar data points have large Hamming distance. Thus they cannot concentrate relevant data points to be within Hamming radius 2. We propose a novel cross-entropy loss for similarity-preserving learning and a novel quantization loss for controlling hashing quality, both based on the Cauchy distribution. To our knowledge, this work is the first endeavor towards deep hashing for Hamming space retrieval.

3. Deep Cauchy Hashing

In similarity retrieval systems, we are given a training set of N points $\{\boldsymbol{x}_i\}_{i=1}^N$, each represented by a D-dimensional feature vector $\boldsymbol{x}_i \in \mathbb{R}^D$. Some pairs of points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are provided with similarity labels s_{ij}, where $s_{ij} = 1$ if $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are similar while $s_{ij} = 0$ if $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are dissimilar. Deep hashing learns a nonlinear hash function $f : \boldsymbol{x} \mapsto \boldsymbol{h} \in \{-1,1\}^K$ from input space $\mathbb{R}^D$ to Hamming space $\{-1,1\}^K$ using deep neural networks, which encodes each point $\boldsymbol{x}$ into compact K-bit hash code $\boldsymbol{h} = f(\boldsymbol{x})$ such that the similarity information conveyed in given pairs $\mathcal{S}$ can be preserved in the compact hash codes. In supervised hashing, the similarity information $\mathcal{S} = \{s_{ij}\}$ can be collected from the semantic labels of data points or relevance feedback from click-through data in online search engines.

Definition 1 (Hamming Space Retrieval). *For binary codes of K bits, the number of distinct hash buckets to examine is $N(K,r) = \sum_{k=0}^{r} \binom{K}{k}$, where r is the Hamming radius. $N(K,r)$ grows rapidly with r and when $r \leq 2$, it only requires $O(1)$ time for each query to find all r-neighbors. Hamming space retrieval refers to the retrieval scenario that directly returns data points within Hamming radius r to each query, by hash table lookups instead of linear scan.*

This paper presents a new Deep Cauchy Hashing (**DCH**) to enable efficient Hamming space retrieval in an end-to-end framework, as shown in Figure 1. The proposed deep architecture accepts pairwise input images $\{(\boldsymbol{x}_i, \boldsymbol{x}_j, s_{ij})\}$ and processes them through an end-to-end pipeline of deep representation learning and binary hash coding: **(1)** a convolutional network (CNN) for learning deep representation of each image $\boldsymbol{x}_i$, **(2)** a fully-connected hash layer (*fch*) for transforming the deep representation into K-bit hash code $\boldsymbol{h}_i \in \{1,-1\}^K$, **(3)** a novel Cauchy cross-entropy loss for similarity-preserving learning in Hamming space, and **(4)** a novel Cauchy quantization loss for controlling both the binarization error and the hashing quality in Hamming space.

3.1. Deep Architecture

The architecture for Deep Cauchy Hashing is shown in Figure 1. We extend from AlexNet [15] with five convolutional layers *conv1–conv5* and three fully connected layers *fc6–fc8*. We replace the classifier layer *fc8* with a new hash layer *fch* of K hidden units, which transforms the representation of the *fc7* layer into K-dimensional continuous code $\boldsymbol{z}_i \in \mathbb{R}^K$ for each image $\boldsymbol{x}_i$. We obtain hash code $\boldsymbol{h}_i$ through the sign thresholding $\boldsymbol{h}_i = \operatorname{sgn}(\boldsymbol{z}_i)$. However, since it is hard to optimize the sign function due to ill-posed gradient, we adopt the hyperbolic tangent (tanh) function to squash the continuous code $\boldsymbol{z}_i$ to be within $[-1,1]$, which reduces the gap between the continuous code $\boldsymbol{z}_i$ and the binary hash code $\boldsymbol{h}_i$. To further guarantee the quality of hash codes for efficient Hamming space retrieval, we preserve the similarity between the training pairs $\{(\boldsymbol{x}_i, \boldsymbol{x}_j, s_{ij}) : s_{ij} \in \mathcal{S}\}$ and control the quantization error, both performed in the Hamming space. Towards this goal, this paper proposes two novel loss functions based on the long-tailed Cauchy distribution: a pairwise Cauchy cross-entropy loss

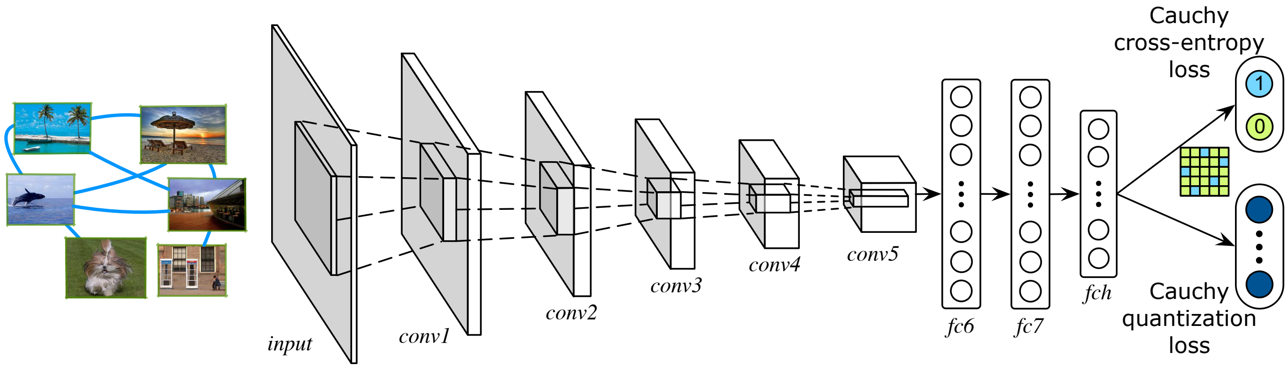


Figure 1. The architecture of the proposed Deep Cauchy Hashing (DCH), which is comprised of four key components: **(1)** a convolutional network (CNN) for learning deep representation of each image $\boldsymbol{x}_i$, **(2)** a fully-connected hash layer (*fch*) for transforming the deep representation into K-bit hash code $\boldsymbol{h}_i \in \{1, -1\}^K$, **(3)** a novel Cauchy cross-entropy loss for similarity-preserving learning in the Hamming space, and **(4)** a novel Cauchy quantization loss for controlling both the binarization error and the hash code quality. *Best viewed in color.*

and a pointwise Cauchy quantization loss, both derived in the Maximum a Posteriori (MAP) estimation framework.

3.2. Bayesian Learning Framework

In this paper, we propose a Bayesian learning framework to perform deep hashing from similarity data by jointly preserving similarity of pairwise images and controlling the quantization error. Given training images with pairwise similarity labels as $\{(\boldsymbol{x}_i, \boldsymbol{x}_j, s_{ij}) : s_{ij} \in \mathcal{S}\}$, the logarithm Maximum a Posteriori (MAP) estimation of the hash codes $\boldsymbol{H} = [\boldsymbol{h}_1, \ldots, \boldsymbol{h}_N]$ for N training images can be defined as

$$\begin{aligned}\log P(\boldsymbol{H}|\mathcal{S}) &\propto \log P(\mathcal{S}|\boldsymbol{H}) P(\boldsymbol{H}) \\ &= \sum_{s_{ij}\in\mathcal{S}} w_{ij} \log P(s_{ij}|\boldsymbol{h}_i, \boldsymbol{h}_j) + \sum_{i=1}^{N} \log P(\boldsymbol{h}_i)\end{aligned} \tag{1}$$

where $P(\mathcal{S}|\boldsymbol{H}) = \prod_{s_{ij}\in\mathcal{S}} [P(s_{ij}|\boldsymbol{h}_i, \boldsymbol{h}_j)]^{w_{ij}}$ is the weighted likelihood function [6], and w_{ij} is the weight for each training pair $(\boldsymbol{x}_i, \boldsymbol{x}_j, s_{ij})$, which tackles the data imbalance problem by weighting training pairs according to the importance of misclassifying that pair. Since similarity label can only be $s_{ij} = 1$ or $s_{ij} = 0$, to account for the data imbalance between similar and dissimilar pairs, we propose

$$w_{ij} = \begin{cases} |\mathcal{S}| / |\mathcal{S}_1|, & s_{ij} = 1 \\ |\mathcal{S}| / |\mathcal{S}_0|, & s_{ij} = 0 \end{cases} \tag{2}$$

where $\mathcal{S}_1 = \{s_{ij} \in \mathcal{S} : s_{ij} = 1\}$ is the set of similar pairs and $\mathcal{S}_0 = \{s_{ij} \in \mathcal{S} : s_{ij} = 0\}$ is the set of dissimilar pairs. For each pair, $P(s_{ij}|\boldsymbol{h}_i, \boldsymbol{h}_j)$ is the conditional probability of similarity label s_{ij} given a pair of hash codes $\boldsymbol{h}_i$ and $\boldsymbol{h}_j$, which can be naturally defined by the Bernoulli distribution,

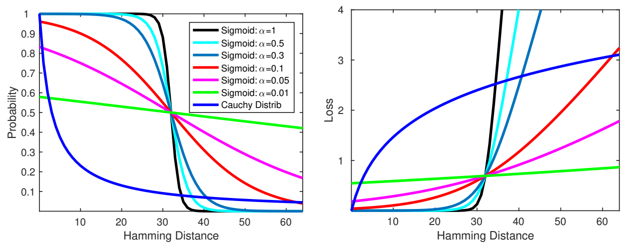
$$\begin{aligned}P(s_{ij}|\boldsymbol{h}_i, \boldsymbol{h}_j) &= \begin{cases} \sigma(\mathrm{d}(\boldsymbol{h}_i, \boldsymbol{h}_j)), & s_{ij} = 1 \\ 1 - \sigma(\mathrm{d}(\boldsymbol{h}_i, \boldsymbol{h}_j)), & s_{ij} = 0 \end{cases} \\ &= \sigma(\mathrm{d}(\boldsymbol{h}_i, \boldsymbol{h}_j))^{s_{ij}} (1 - \sigma(\mathrm{d}(\boldsymbol{h}_i, \boldsymbol{h}_j)))^{1-s_{ij}}\end{aligned} \tag{3}$$

where $\mathrm{d}(\boldsymbol{h}_i, \boldsymbol{h}_j)$ denotes the Hamming distance between hash codes $\boldsymbol{h}_i$ and $\boldsymbol{h}_j$, and σ is a well-defined probability function to be elaborated in the next subsection.

Similar to binary-class logistic regression for pointwise data, we see in Equation (3) that the smaller the Hamming distance $\mathrm{d}(\boldsymbol{h}_i, \boldsymbol{h}_j)$ is, the larger the conditional probability $P(1|\boldsymbol{h}_i, \boldsymbol{h}_j)$ will be, implying that the image pair $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ should be classified as similar; otherwise, the larger the conditional probability $P(0|\boldsymbol{h}_i, \boldsymbol{h}_j)$ will be, implying that the image pair should be classified as dissimilar. Hence, Equation (3) is a reasonable extension of the binary-class logistic regression to the pairwise classification scenario, which is a natural solution to the binary similarity labels $s_{ij} \in \{0, 1\}$.

3.3. Cauchy Hash Learning

With the Bayesian learning framework, any valid probability function σ and distance function d can be used to instantiate a specific hashing model. Previous state-of-the-art deep hashing methods, such as DHN [36] and HashNet [4], usually adopt generalized sigmoid function $\sigma(x) = 1/(1 + e^{-\alpha x})$ as the probability function. However, we discover a key misspecification problem of the generalized sigmoid function as illustrated in Figure 2. We can observe that the probability of generalized sigmoid function (by varying α) stays high when the Hamming distance between hash codes is much larger than 2 and only starts to decrease obviously when the Hamming distance becomes close to $K/2$. This implies that previous deep hashing methods cannot pull the Hamming distance between the hash codes of similar data points to be smaller than 2, because the probabilities for different Hamming distances smaller than $K/2$ are not discriminative enough. This is a severe disadvantage of the existing hashing methods, which makes efficient Hamming space retrieval impossible. Note that, a well-specified loss function for Hamming space retrieval should penalize significantly for similar points with Hamming distance greater than 2, while prior methods are misspecified for this goal.



(a) Probability (b) Loss

Figure 2. The values of Probability (a) and Loss (b) with respect to Hamming Distance between the hash codes of similar data points ($s_{ij} = 1$). Probability (Loss) based on generalized sigmoid function is very large (small) even for Hamming distance much larger than 2, which is ill-specified for Hamming ball retrieval. As a desired property, Probability based on Cauchy distribution is large only for smaller Hamming distance; loss based on Cauchy distribution significantly penalizes for Hamming distance larger than 2.

To tackle the above misspecification problem of sigmoid function, we propose a novel probability function based on the Cauchy distribution with many desired properties as

$$\sigma\left(\mathrm{d}\left(\boldsymbol{h}_i, \boldsymbol{h}_j\right)\right) = \frac{\gamma}{\gamma + \mathrm{d}\left(\boldsymbol{h}_i, \boldsymbol{h}_j\right)}, \tag{4}$$

where γ is the scale parameter of the Cauchy distribution, and the normalization constant $\frac{1}{\pi\sqrt{\gamma}}$ is omitted for clarity, since it will not influence the final model. In Figure 2, we can observe that the probability of the proposed Cauchy distribution decreases very fast when the Hamming distance is small, resulting in that the similar points will be pulled to be within small Hamming radius. The decaying speed of the probability will be even faster by using a smaller γ, which imposes more force to concentrate similar points to be within small Hamming radius. Hence, the scale parameter γ is very important to control the tradeoff levels between precision and recall. By simply varying γ, we can support diverse Hamming space retrieval scenarios with different Hamming radiuses to give different pruning rates.

Since discrete optimization of Equation (1) with binary constraints $\boldsymbol{h}_i \in \{-1, 1\}^K$ is very challenging, continuous relaxation is applied to the binary constraints for ease of optimization, as adopted by most previous hashing methods [31, 36]. To control the quantization error $\|\boldsymbol{h}_i - \mathrm{sgn}(\boldsymbol{h}_i)\|$ caused by continuous relaxation and to learn high-quality hash codes, we propose a novel prior for each hash code $\boldsymbol{h}_i$ based on a symmetric variant of the Cauchy distribution as

$$P\left(\boldsymbol{h}_i\right) = \frac{\gamma}{\gamma + \mathrm{d}\left(\left|\boldsymbol{h}_i\right|, \mathbf{1}\right)}, \tag{5}$$

where γ is the scale parameter of the symmetric Cauchy distribution, and $\mathbf{1} \in \mathbb{R}^K$ is the vector of ones.

By using the continuous relaxation, we need to replace the Hamming distance with its best approximation on continuous codes. For a pair of binary hash codes $\boldsymbol{h}_i$ and $\boldsymbol{h}_j$, there exists a nice relationship between their Hamming distance $\mathrm{d}(\boldsymbol{h}_i, \boldsymbol{h}_j)$ and the normalized Euclidean distance as

$$\begin{aligned} \mathrm{d}\left(\boldsymbol{h}_i, \boldsymbol{h}_j\right) &= \frac{K}{4}\left\|\frac{\boldsymbol{h}_i}{\|\boldsymbol{h}_i\|} - \frac{\boldsymbol{h}_j}{\|\boldsymbol{h}_j\|}\right\|_2^2 \\ &= \frac{K}{2}\left(1 - \cos\left(\boldsymbol{h}_i, \boldsymbol{h}_j\right)\right). \end{aligned} \tag{6}$$

Hence this paper adopts $\mathrm{d}\left(\boldsymbol{h}_i, \boldsymbol{h}_j\right) = \frac{K}{2}\left(1 - \cos\left(\boldsymbol{h}_i, \boldsymbol{h}_j\right)\right)$. After using the continuous relaxation to ease the optimization, the normalized Euclidean distance compares each pair of continuous codes on a unit sphere, while the Hamming distance compares each pair of hash codes on a unit hypercube. More intuitively, the unit sphere is always a circumscribed sphere of the hypercube. As a desirable result, by mitigating the diversity of code lengths, the normalized Euclidean distance on continuous codes is always the upper bound of the Hamming distance on hash codes.

By taking Equations (3) and (5) into the MAP estimation framework in Equation (1), we obtain the optimization problem of the proposed Deep Cauchy Hashing (**DCH**) as

$$\min_{\Theta} L + \lambda Q, \tag{7}$$

where λ is a hyper-parameter to trade-off the Cauchy cross-entropy loss L and the Cauchy quantization loss Q, and Θ denotes the set of network parameters to be optimized. Specifically, the Cauchy cross-entropy loss L is derived as

$$L = \sum_{s_{ij} \in \mathcal{S}} w_{ij}\left(s_{ij}\log\frac{\mathrm{d}\left(\boldsymbol{h}_i, \boldsymbol{h}_j\right)}{\gamma} + \log\left(1 + \frac{\gamma}{\mathrm{d}\left(\boldsymbol{h}_i, \boldsymbol{h}_j\right)}\right)\right), \tag{8}$$

and similarly, the Cauchy quantization loss is derived as

$$Q = \sum_{i=1}^{N}\log\left(1 + \frac{\mathrm{d}\left(\left|\boldsymbol{h}_i\right|, \mathbf{1}\right)}{\gamma}\right), \tag{9}$$

where $\mathrm{d}(\cdot, \cdot)$ is either the Hamming distance between the hash codes or the normalized Euclidean distance between the continuous codes. Since the quantization error will be controlled by the proposed Cauchy quantization loss in the joint optimization problem (7), for ease of optimization, we can use continuous relaxation for the hash codes $\boldsymbol{h}_i$ during training (only for training, not for testing).

Based on the MAP estimation in Equation (7), we can enable statistically optimal learning of compact hash codes by jointly preserving the pairwise similarity and controlling the quantization error. Finally, we can obtain K-bit binary codes by simple sign thresholding $\boldsymbol{h} \leftarrow \mathrm{sgn}(\boldsymbol{h})$, where $\mathrm{sgn}(\boldsymbol{h})$ is the sign function on vectors that for $i = 1, \ldots, K$, $\mathrm{sgn}(h_i) = 1$ if $h_i > 0$, otherwise $\mathrm{sgn}(h_i) = -1$. It is worth noting that, since we have minimized the quantization error in (7) during training, this final binarization step will incur very small loss of retrieval quality as validated empirically.

4. Experiments

We evaluate the efficacy of the proposed DCH approach with several state-of-the-art shallow and deep hashing methods on three benchmark datasets. Codes and configurations will be available at: https://github.com/thuml.

4.1. Setup

NUS-WIDE [5] is a benchmark dataset that contains 269,648 images from Flickr.com. Each image is manually annotated by some of the 81 ground truth concepts (categories) for evaluating retrieval models. We follow similar experimental protocols in [36, 4], and randomly sample 5,000 images as the query points, with the remaining images used as the database and randomly sample 10,000 images from the database as the training points.

MS-COCO [20] is a popular dataset for image recognition, segmentation and captioning. The current release contains 82,783 training images and 40,504 validation images, where each image is labeled by some of the 80 semantic concepts. We randomly sample 5,000 images as query points, with the rest used as the database, and randomly sample 10,000 images from the database for training.

CIFAR-10 is a standard dataset with 60,000 images in 10 classes. We follow protocol in [36, 3] to randomly select 100 images per class as query set, 500 images per class as training set, and the rest images are used as the database.

Following standard protocol as in [33, 17, 36, 4], the similarity information for hash learning and for ground-truth evaluation is constructed from image labels: if two images i and j share at least one label, they are similar and $s_{ij} = 1$; otherwise, they are dissimilar and $s_{ij} = 0$. Note that, although we use the image labels to construct the similarity information, the proposed approach DCH can learn hash codes when only the similarity information is available. By constructing the training data in this way, the ratio between the number of dissimilar pairs and the number of similar pairs is roughly 10, 5, and 1 for CIFAR-10, NUS-WIDE, and MS-COCO, respectively. These datasets exhibit the data imbalance phenomenon and can be used to evaluate different hashing methods under data imbalance scenario.

We compare the retrieval performance of **DCH** with eight classical or state-of-the-art hashing methods: supervised shallow methods **ITQ-CCA** [12], **BRE** [16], **KSH** [22], and **SDH** [28], and supervised deep methods **CNNH** [33], **DNNH** [17], **DHN** [36], and **HashNet** [4].

We follow standard evaluation methods for Hamming space retrieval [9], which consists of two consecutive steps: **(1)** Pruning, to return data points within Hamming radius 2 for each query using hash table lookups; **(2)** Scanning, to re-rank the returned data points in ascending order of their distances to each query using continuous codes. To evaluate the effectiveness of Hamming space retrieval, we report three standard evaluation metrics to measure the quality of the data points within Hamming radius 2: Mean Average Precision within Hamming Radius 2 (**MAP@H≤2**), Precision curves within Hamming Radius 2 (**P@H≤2**), and Recall curves within Hamming Radius 2 (**R@H≤2**).

Due to the potentially best efficiency, the search based on pruning followed by re-ranking is widely-deployed in large-scale online retrieval systems such as search engines. And the continuous representations before the sign function are always adopted for the re-ranking step. In the MAP@H≤2 evaluation metric, we compute the Mean Average Precision for the re-ranked list of the data points pruned by Hamming radius 2, which can measure the search quality. More specifically, given a set of queries, we first compute the Average Precision (AP) of each query as

$$\mathrm{AP}@T = \frac{\sum_{t=1}^{T} P(t)\,\delta(t)}{\sum_{t'=1}^{T} \delta(t')}, \tag{10}$$

where T is the number of top-returned data points within Hamming radius 2, $P(t)$ denotes the precision of top t retrieved results, and $\delta(t) = 1$ if the t-th retrieved result is a true neighbor of the query, otherwise $\delta(t) = 0$. Then MAP@H≤2 is computed as the mean of average precisions for all queries. The larger the MAP@H≤2, the better the search quality for the data points within Hamming radius 2.

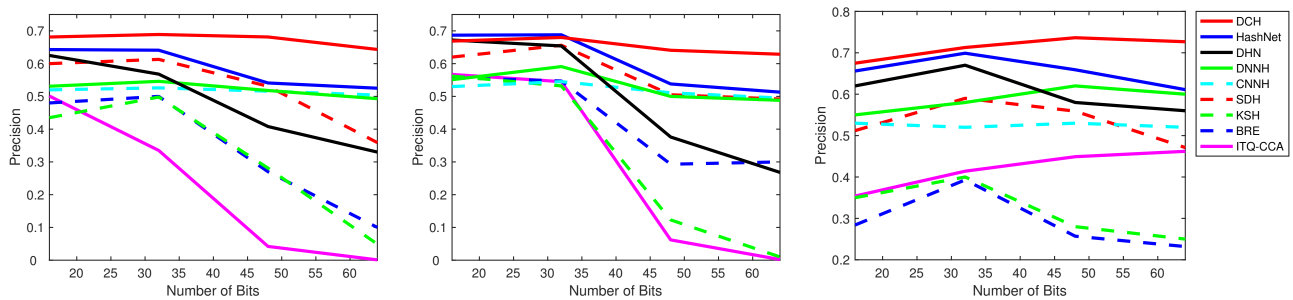
For shallow hashing methods, we use as image features the 4096-dimensional DeCAF_7 features [7]. For deep hashing methods, we use raw images as the input. We adopt the AlexNet architecture [15] for all deep hashing methods, and implement DCH based on the **TensorFlow** framework. We fine-tune convolutional layers *conv1–conv5* and fully-connected layers *fc6–fc7* copied from the AlexNet model pre-trained on ImageNet and train the last layer, all through back-propagation. As the last layer is trained from scratch, we set its learning rate to be 10 times that of the lower layers. We use mini-batch stochastic gradient descent (SGD) with 0.9 momentum and cross-validate the learning rate from 10^{-5} to 10^{-2} with a multiplicative step-size $10^{\frac{1}{2}}$. We fix the mini-batch size of images as 256 and the weight decay parameter as 0.0005. We select the model parameters of DCH, λ and γ, by cross-validation. We also select the parameters of each comparison method by cross-validation.

4.2. Results

The Mean Average Precision of Re-ranking within Hamming Radius 2 (MAP@H≤2) results of all comparison methods are listed in Table 1. Results show that DCH substantially outperforms all comparison methods, in that DCH can perform high-quality pruning within Hamming radius 2 in the first step, enabling efficient re-ranking in the second step. Specifically, compared to SDH, the best shallow hashing method with deep features as input, DCH achieves absolute increases of **14.4%**, **13.3%** and **21.9%** in average

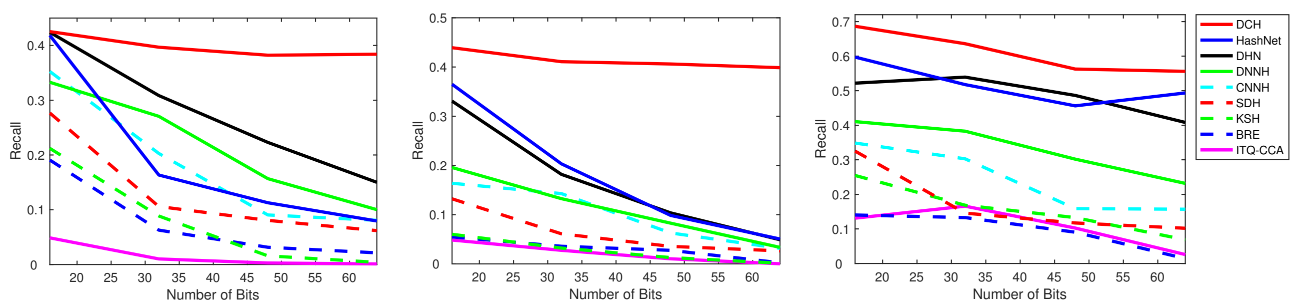
Table 1. Mean Average Precision of Re-ranking within Hamming Radius 2 (MAP@H≤2) for Different Bits on Three Benchmark Datasets

Method	NUS-WIDE				MS-COCO				CIFAR-10			
	16 bits	32 bits	48 bits	64 bits	16 bits	32 bits	48 bits	64 bits	16 bits	32 bits	48 bits	64 bits
ITQ-CCA [12]	0.5706	0.4397	0.0825	0.0051	0.5949	0.5612	0.0585	0.0105	0.4258	0.4652	0.4774	0.4932
BRE [16]	0.5502	0.5422	0.4128	0.2202	0.5625	0.5498	0.4214	0.4014	0.4216	0.4519	0.4002	0.3438
KSH [22]	0.5185	0.5659	0.4102	0.0608	0.5797	0.5532	0.2338	0.0216	0.4368	0.4585	0.4012	0.3819
SDH [28]	0.6681	0.6824	0.5979	0.4679	0.6449	0.6766	0.5226	0.5108	0.5620	0.6428	0.6069	0.5012
CNNH [33]	0.5843	0.5989	0.5734	0.5729	0.5602	0.5685	0.5376	0.5058	0.5512	0.5468	0.5454	0.5364
DNNH [17]	0.6191	0.6216	0.5902	0.5626	0.5771	0.6023	0.5235	0.5013	0.5703	0.5985	0.6421	0.6118
DHN [36]	0.6901	0.7021	0.6685	0.5664	0.6749	0.6680	0.5151	0.4186	0.6929	0.6445	0.5835	0.5883
HashNet [4]	0.6944	0.7147	0.6736	0.6190	0.6851	0.6900	0.5589	0.5344	0.7476	0.7776	0.6399	0.6259
DCH	**0.7401**	**0.7720**	**0.7685**	**0.7124**	**0.7010**	**0.7576**	**0.7251**	**0.7013**	**0.7901**	**0.7979**	**0.8071**	**0.7936**



(a) NUS-WIDE (b) MS-COCO (c) CIFAR-10

Figure 3. The Precision curves within Hamming Radius 2 (P@H≤2) of DCH and comparison methods on the three benchmark datasets.



(a) NUS-WIDE (b) MS-COCO (c) CIFAR-10

Figure 4. The Recall curves within Hamming Radius 2 (R@H≤2) of DCH and comparison methods on the three benchmark datasets.

MAP@H≤2 with different code lengths on NUS-WIDE, MS-COCO and CIFAR-10, respectively. DCH outperforms HashNet, the state-of-the-art deep hashing method, by large margins of **7.3%**, **10.4%** and **9.9%** in average MAP@H≤2 with different code lengths on three benchmark datasets.

The MAP@H≤2 results reveal some interesting insights. **(1)** Shallow hashing methods cannot learn discriminative deep features and compact hash codes through end-to-end framework, which explains the fact that they are surpassed by deep hashing methods. **(2)** Deep hashing methods DHN and HashNet learn less lossy hash codes by preserving the similarity information and controlling the quantization error, which also significantly outperform deep methods CNNH and DNNH without reducing the quantization error.

The proposed DCH model improves substantially from the state-of-the-art HashNet by two important perspectives: **(1)** DCH preserves similarity relationships based on Cauchy distribution, which can achieve better pruning performance within Hamming radius 2; **(2)** DCH learns the novel Cauchy cross-entropy loss and Cauchy quantization loss based on normalized distance, which can better approximate the Hamming distance to learn nearly lossless hash codes.

The performance of Precision within Hamming Radius 2 (P@H≤2) is very important for Hamming space retrieval, since it only requires $O(1)$ time for each query and enables really efficient pruning. As shown in Figure 3, DCH often achieves the highest P@H≤2 results on all three benchmark datasets with regard to different code lengths. This validates that DCH can learn compacter hash codes than all comparison methods and can enable more efficient and accurate Hamming space retrieval. With longer hash codes, the Hamming space will become more sparse and fewer

Table 2. Mean Average Precision of Re-ranking within Hamming Radius 2 (MAP@H≤2) of DCH and Its Variants on Three Datasets

Method	NUS-WIDE				MS-COCO				CIFAR-10			
	16 bits	32 bits	48 bits	64 bits	16 bits	32 bits	48 bits	64 bits	16 bits	32 bits	48 bits	64 bits
DCH	**0.7401**	**0.7720**	**0.7685**	**0.7124**	**0.7010**	**0.7576**	**0.7251**	**0.7013**	**0.7901**	**0.7979**	**0.8071**	**0.7936**
DCH-Q	0.7086	0.7458	<u>0.7432</u>	<u>0.7028</u>	0.6785	<u>0.7238</u>	<u>0.6982</u>	<u>0.6913</u>	<u>0.7645</u>	<u>0.7789</u>	<u>0.7858</u>	<u>0.7832</u>
DCH-C	0.6997	0.7205	0.6874	0.6328	0.6767	0.6972	0.5801	0.5536	0.7513	0.7628	0.6819	0.6627
DCH-E	<u>0.7178</u>	<u>0.7511</u>	0.7302	0.6982	<u>0.6826</u>	0.7128	0.6803	0.6735	0.7598	0.7739	0.7495	0.7287

data points will fall in the Hamming ball within radius 2. This is why most previous hashing methods achieve worse retrieval performance with longer code lengths. It is worth noting that DCH achieves a relatively mild decrease or even an increase in accuracy by longer code lengths, validating that DCH can effectively concentrate hash codes of similar data points together to be within the Hamming radius 2, which significantly benefit Hamming space retrieval.

The performance of Recall within Hamming Radius 2 (R@H≤2) is crucial for Hamming space retrieval, since it is likely that all data points will be pruned out due to the highly sparse Hamming space. As shown in Figure 4, DCH achieves the highest R@H≤2 results on all three datasets w.r.t different code lengths, which is very encouraging. This validates that DCH can concentrate more relevant points to be within the Hamming ball of radius 2 than all the comparison methods. As the Hamming space will become more sparse when using longer hash codes, most hashing baselines incur serious performance drop on R@H≤2. Since the relationships between data pairs on multi-label datasets are more complex than that on single-label datasets, the performance drop becomes more serious on multi-label datasets such as NUS-WIDE and MS-COCO. By introducing the novel Cauchy cross-entropy loss and Cauchy quantization loss, even on multi-label datasets, the proposed DCH incurs very small performance drop on R@H≤2 as the hash codes become longer, showing that DCH can concentrate more relevant points to be within Hamming radius 2 even using longer code lengths. The ability to use longer codes gives a flexible to tradeoff between accuracy and efficiency, a flexibility that is often impossible for previous hashing methods.

4.3. Discussion

4.3.1 Ablation Study

We investigate three DCH variants: (1) **DCH-Q** is a DCH variant without using the new Cauchy quantization loss (9), in other words λ=0; (2) **DCH-C** is a DCH variant replacing the Cauchy cross-entropy loss with the popular sigmoid cross-entropy loss [36, 4]; (3) **DCH-E** is a DCH variant replacing the normalized Euclidean distance (6) with the Euclidean distance as $\mathrm{d}(\boldsymbol{h}_i, \boldsymbol{h}_j) = \frac{1}{4}\|\boldsymbol{h}_i - \boldsymbol{h}_j\|_2^2$. The MAP@H≤2 results w.r.t. different code lengths on all three benchmark datasets are reported in Table 2.

Cauchy Cross-Entropy Loss. DCH outperforms DCH-C by very large margins of 6.3%, 9.4% and 8.3% in average MAP@H≤2 with different code lengths on NUS-WIDE, MS-COCO and CIFAR-10, respectively. The Cauchy cross-entropy loss (8) is based on the Cauchy distribution, which can keep more relevant points to be within small Hamming radius to enable effective Hamming space retrieval, whereas the sigmoid cross-entropy loss cannot achieve this desired property. Also, DCH outperforms DCH-E by large margins of 2.4%, 3.4% and 4.4% in average MAP@H≤2 with different code lengths on three datasets. In real search engines, normalized distance is widely used to mitigate the diversity of vector lengths and improve the retrieval quality, which has not been integrated with the cross-entropy loss [31].

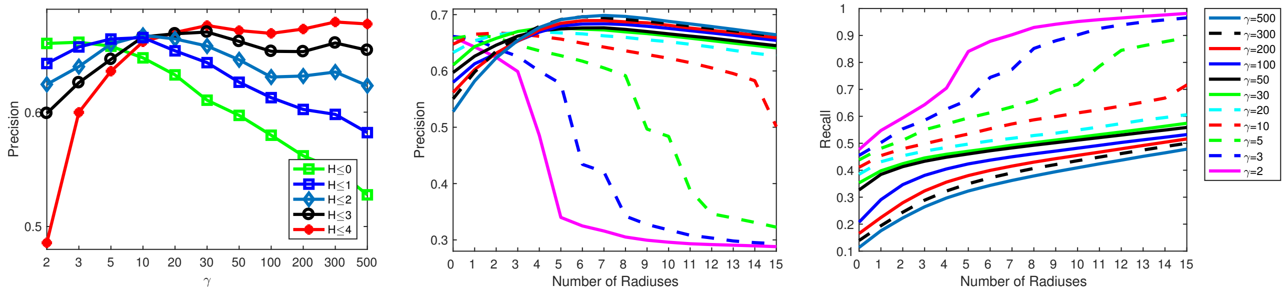
Cauchy Quantization Loss. DCH outperforms DCH-Q by 2.3%, 2.3%, and 1.9% in average MAP@H≤2 with different code lengths on three benchmarks, respectively. These results validate that the novel Cauchy quantization loss (9) can enhance the pruning efficiency and improve the pruning and re-ranking results within Hamming radius 2.

4.3.2 Sensitivity Study

In Hamming space retrieval, a key aspect is to control the tradeoff between precision and recall as well as efficiency w.r.t. different Hamming radiuses. As aforementioned, the time cost for Hamming pruning through hash table lookups is $N(K,r) = \sum_{k=0}^{r}\binom{K}{k}$, where r is the Hamming radius. Hence $N(K,r) \propto K^r$, and we can only tolerate smaller Hamming radius, typically $r \leq 10$. In other words, we cannot use large Hamming radius for higher recall. In this paper, we introduce the Cauchy distribution parameter γ to fully tradeoff precision and recall, with sensitivity study of γ in terms of precision and recall shown in Figure 5.

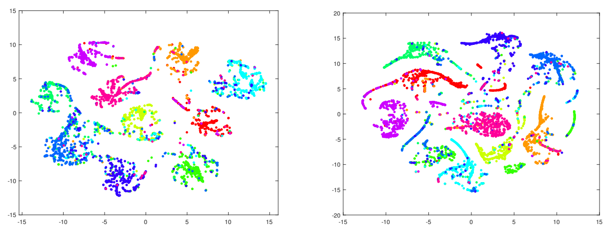
Figure 5(a) shows the precision curves w.r.t. different values of γ for practical Hamming radiuses $r = [0, \dots, 4]$. Figure 5(b) shows the precision curves w.r.t. different Hamming radiuses for typical values of $\gamma = [2, \dots, 500]$, which is an alternative view of Figure 5(a). As can be seen, larger (smaller) Hamming radius requires larger (smaller) value of γ to guarantee the highest precision, which is consistent with the theoretical analysis. Although larger Hamming radius leads to higher precision when larger γ is used, the pruning cost will be exponentially enlarged as $O(K^r)$.

Figure 5(c) shows the recall curves w.r.t. different Hamming radiuses for typical values of $\gamma = [2, \dots, 500]$. As can be seen, larger (smaller) Hamming radius leads to higher (lower) recall, which is consistent with the theoretical anal-



(a) Precision w.r.t different γ (b) Precision w.r.t different radiuses (c) Recall w.r.t different radiuses

Figure 5. Sensitivity study for DCH using 64-bit hash codes on NUS-WIDE dataset: (a) Precision curves w.r.t. γ for different Hamming radiuses, (b) Precision curves w.r.t. Hamming radiuses for different γ, and (c) Recall curves w.r.t. Hamming radiuses for different γ.



(a) DCH (b) HashNet

Figure 6. The t-SNE visualization of hash codes on CIFAR-10.

ysis. A very interesting observation is that, smaller (larger) γ leads to larger (smaller) recall. As analyzed before, γ controls the power of concentrating relevant data points within smaller Hamming balls, and smaller γ leads to larger concentration power. Again, although larger Hamming radius leads to higher recall when smaller γ is used, the pruning cost will be exponentially enlarged as $O(K^r)$.

Therefore, DCH provides with the powerful flexibility to tradeoff precision and recall as well the pruning efficiency. In practice, we usually decide the Hamming radius first based on the efficiency requirement, and typically $r = 2$. Then we tradeoff the precision and recall solely based on γ. As seen from Figure 5, $\gamma = 5$ is the best choice for Hamming radius $r = 2$. Besides the optimal value, we can vary $\gamma \in [2, 50]$ to achieve both satisfactory precision and recall.

4.3.3 Visualization Study

Visualization of Hash Codes by t-SNE. Figure 6 shows the t-SNE visualization [29] of the hash codes learned by DCH and the best deep hashing baseline HashNet [4] on CIFAR-10 dataset. We can observe that the hash codes generated by DCH show clear discriminative structures where the hash codes in different categories are well separated, while the hash codes generated by HashNet do not show such clear structures. This verifies that by introducing the Cauchy distribution for hashing, the hash codes generated through DCH are more discriminative than that generated by HashNet, enabling more effective image retrieval.

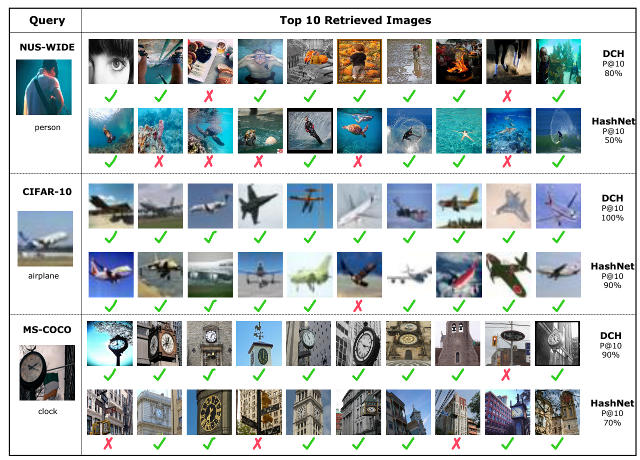


Figure 7. The top 10 images returned by DCH and HashNet.

Top 10 Results. In Figure 7, we visualize the top 10 returned images of DCH and the best deep hashing baseline HashNet [4] for three query images on NUS-WIDE, MS-COCO and CIFAR-10, respectively. It shows that DCH can yield much more relevant and user-desired retrieval results.

5. Conclusion

This paper establishes efficient and effective Hamming space retrieval with constant-time search complexity. The proposed Deep Cauchy Hashing (DCH) approach generates compact and concentrated hash codes by jointly optimizing a novel Cauchy cross-entropy loss and a Cauchy quantization loss in a single Bayesian learning framework. The overall model can be trained end-to-end with well-specified loss functions. Extensive experiments show that DCH can yield state-of-the-art Hamming space retrieval performance on three datasets, NUS-WIDE, CIFAR-10, and MS-COCO.

6. Acknowledgements

This work was supported by the National Key R&D Program of China (No. 2017YFC1502003), the National Natural Science Foundation of China (No. 61772299, 71690231, 61502265), and Tsinghua TNList Laboratory Key Project.

References

[1] Y. Cao, M. Long, J. Wang, and S. Liu. Deep visual-semantic quantization for efficient image retrieval. In *CVPR*, 2017. 2

[2] Y. Cao, M. Long, J. Wang, Q. Yang, and P. S. Yu. Deep visual-semantic hashing for cross-modal retrieval. In *SIGKDD*, pages 1445–1454, 2016. 2

[3] Y. Cao, M. Long, J. Wang, H. Zhu, and Q. Wen. Deep quantization network for efficient image retrieval. In *AAAI*. AAAI, 2016. 5

[4] Z. Cao, M. Long, J. Wang, and P. S. Yu. Hashnet: Deep learning to hash by continuation. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 5609–5618, 2017. 1, 2, 3, 5, 6, 7, 8

[5] T.-S. Chua, J. Tang, R. Hong, H. Li, Z. Luo, and Y.-T. Zheng. Nus-wide: A real-world web image database from national university of singapore. In *ICMR*. ACM, 2009. 5

[6] J. P. Dmochowski, P. Sajda, and L. C. Parra. Maximum likelihood in cost-sensitive learning: Model specification, approximations, and upper bounds. *Journal of Machine Learning Research (JMLR)*, 11(Dec):3313–3332, 2010. 3

[7] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *ICML*, 2014. 5

[8] V. Erin Liong, J. Lu, G. Wang, P. Moulin, and J. Zhou. Deep hashing for compact binary codes learning. In *CVPR*, pages 2475–2483. IEEE, 2015. 1, 2

[9] D. J. Fleet, A. Punjani, and M. Norouzi. Fast search in hamming space with multi-index hashing. In *CVPR*. IEEE, 2012. 1, 2, 5

[10] A. Gionis, P. Indyk, R. Motwani, et al. Similarity search in high dimensions via hashing. In *VLDB*, volume 99, pages 518–529. ACM, 1999. 1

[11] Y. Gong, S. Kumar, H. Rowley, S. Lazebnik, et al. Learning binary codes for high-dimensional data using bilinear projections. In *CVPR*, pages 484–491. IEEE, 2013. 2

[12] Y. Gong and S. Lazebnik. Iterative quantization: A procrustean approach to learning binary codes. In *CVPR*, pages 817–824, 2011. 1, 2, 5, 6

[13] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *CVPR*, 2016. 2

[14] H. Jegou, M. Douze, and C. Schmid. Product quantization for nearest neighbor search. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 33(1):117–128, Jan 2011. 2

[15] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012. 2, 5

[16] B. Kulis and T. Darrell. Learning to hash with binary reconstructive embeddings. In *NIPS*, pages 1042–1050, 2009. 1, 2, 5, 6

[17] H. Lai, Y. Pan, Y. Liu, and S. Yan. Simultaneous feature learning and hash coding with deep neural networks. In *CVPR*. IEEE, 2015. 1, 2, 5, 6

[18] M. S. Lew, N. Sebe, C. Djeraba, and R. Jain. Content-based multimedia information retrieval: State of the art and challenges. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 2(1):1–19, Feb. 2006. 1

[19] W.-J. Li, S. Wang, and W.-C. Kang. Feature learning based deep supervised hashing with pairwise labels. In *IJCAI*, 2016. 1

[20] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *ECCV*, pages 740–755. Springer, 2014. 5

[21] H. Liu, R. Wang, S. Shan, and X. Chen. Deep supervised hashing for fast image retrieval. In *CVPR*, pages 2064–2072, 2016. 1, 2

[22] W. Liu, J. Wang, R. Ji, Y.-G. Jiang, and S.-F. Chang. Supervised hashing with kernels. In *CVPR*. IEEE, 2012. 1, 2, 5, 6

[23] W. Liu, J. Wang, S. Kumar, and S.-F. Chang. Hashing with graphs. In *ICML*. ACM, 2011. 2

[24] X. Liu, J. He, C. Deng, and B. Lang. Collaborative hashing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2139–2146, 2014. 1, 2

[25] X. Liu, J. He, B. Lang, and S.-F. Chang. Hash bit selection: a unified solution for selection problems in hashing. In *CVPR*, pages 1570–1577. IEEE, 2013. 1, 2

[26] M. Norouzi and D. M. Blei. Minimal loss hashing for compact binary codes. In *ICML*, pages 353–360. ACM, 2011. 1, 2

[27] R. Salakhutdinov and G. E. Hinton. Learning a nonlinear embedding by preserving class neighbourhood structure. In *AISTATS*, pages 412–419, 2007. 2

[28] F. Shen, C. Shen, W. Liu, and H. Tao Shen. Supervised discrete hashing. In *CVPR*. IEEE, June 2015. 1, 2, 5, 6

[29] L. van der Maaten and G. Hinton. Visualizing high-dimensional data using t-sne. *Journal of Machine Learning Research*, 9: 2579–2605, Nov 2008. 8

[30] J. Wang, S. Kumar, and S.-F. Chang. Semi-supervised hashing for large-scale search. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 34(12):2393–2406, 2012. 1, 2

[31] J. Wang, T. Zhang, J. Song, N. Sebe, and H. T. Shen. A survey on learning to hash. *IEEE Trans. Pattern Anal. Mach. Intell.*, 40(4):769–790, 2018. 1, 2, 4, 7

[32] Y. Weiss, A. Torralba, and R. Fergus. Spectral hashing. In *NIPS*, 2009. 2

[33] R. Xia, Y. Pan, H. Lai, C. Liu, and S. Yan. Supervised hashing for image retrieval via image representation learning. In *AAAI*, pages 2156–2162. AAAI, 2014. 1, 2, 5, 6

[34] F. X. Yu, S. Kumar, Y. Gong, and S.-F. Chang. Circulant binary embedding. In *ICML*, pages 353–360. ACM, 2014. 2

[35] P. Zhang, W. Zhang, W.-J. Li, and M. Guo. Supervised hashing with latent factor models. In *SIGIR*, pages 173–182. ACM, 2014. 1, 2

[36] H. Zhu, M. Long, J. Wang, and Y. Cao. Deep hashing network for efficient similarity retrieval. In *AAAI*. AAAI, 2016. 1, 2, 3, 4, 5, 6, 7