HashGAN: Deep Learning to Hash with Pair Conditional Wasserstein GAN

Yue Cao, Bin Liu, Mingsheng Long,* 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

Deep learning to hash improves image retrieval performance by end-to-end representation learning and hash coding from training data with pairwise similarity information. Subject to the scarcity of similarity information that is often expensive to collect for many application domains, existing deep learning to hash methods may overfit the training data and result in substantial loss of retrieval quality. This paper presents HashGAN, a novel architecture for deep learning to hash, which learns compact binary hash codes from both real images and diverse images synthesized by generative models. The main idea is to augment the training data with nearly real images synthesized from a new Pair Conditional Wasserstein GAN (PC-WGAN) conditioned on the pairwise similarity information. Extensive experiments demonstrate that HashGAN can generate high-quality binary hash codes and yield state-of-the-art image retrieval performance on three benchmarks, 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 neighbors (ANN) search has attracted increasing attention. Parallel to the traditional indexing methods [21], another advantageous solution is hashing methods [37], 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 [37] that build data-dependent hash encoding schemes for efficient image retrieval, which have shown better performance than data-independent hashing methods, e.g. Locality-Sensitive Hashing (LSH) [12].

Many learning to hash methods have been proposed to enable efficient ANN search by Hamming ranking of compact binary hash codes [19, 13, 28, 11, 25, 36, 41]. Recently, deep learning to hash methods [40, 20, 34, 10, 42, 22, 24, 6] 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 results on many datasets. In particular, it proves crucial to jointly learn similarity-preserving representations and control quantization error of converting continuous representations to binary codes [42, 22, 24, 6].

However, the encouraging performance comes only by large-scale image data where sufficient supervised information is available in the forms of pointwise labels or pairwise similarity. In many image retrieval applications, the supervised information available may be insufficient, especially for new domains. And it is usually costly or even prohibitive to annotate sufficient training data for deep learning to hash. Subject to the scarcity of similarity information, existing deep learning to hash methods [42, 6] may overfit the training images and result in substantial loss of retrieval quality.

This paper presents HashGAN, a novel deep architecture for deep learning to hash, which learns compact binary hash codes from both real images and large-scale synthesized images. We propose a novel Pair Conditional Wasserstein GAN (PC-WGAN), which synthesizes discriminative and diverse images by conditioning on the pairwise similarity information. To the best of our knowledge, PC-WGAN is the first GAN that enables image synthesis by incorporating pairwise similarity information. Well-specified loss functions including cosine cross-entropy loss and cosine quantization loss are proposed for similarity-preserving learning and quantization error control. The proposed HashGAN can be trained end-to-end by back-propagation in a minimax optimization mechanism. Extensive experiments demonstrate that HashGAN can generate high-quality binary hash codes and yield state-of-the-art multimedia retrieval performance on three datasets, NUS-WIDE, CIFAR-10, and MS-COCO.

2. Related Work

Hashing Methods. Existing hashing methods [2, 4, 19, 13, 28, 11, 25, 41] consist of unsupervised and supervised

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

hashing. Please refer to [37] for a comprehensive survey.

Unsupervised hashing methods learn hash functions that encode data to binary codes by training from unlabeled data [33, 13, 17, 39, 26]. Supervised hashing further explores supervised information (e.g. pairwise similarity or relevance feedback) to generate compact hash codes [19, 28, 25, 34]. Recently, deep learning to hash methods yield breakthrough results on image retrieval datasets by blending the power of deep learning [40, 20]. In particular, DHN [42] is the first end-to-end framework that jointly preserves pairwise similarity and controls the quantization error. HashNet [6] improves DHN by balancing the positive and negative pairs in training data to trade of precision vs. recall, and by continuation technique for lower quantization error, which obtains state-of-the-art performance on several benchmark datasets.

Generative Models. Generative Adversarial Networks (GANs) [14] are powerful models for generating images in a minimax game mechanism without requiring supervised information. State-of-the-art unsupervised generative models for image synthesis include Deep Convolutional GANs (DCGANs) [31] and Wasserstein GANs (WGANs) [1, 15]. Recently, a more powerful family of generative models synthesize images with GANs by further conditioning on supervised information (e.g., class labels or text descriptions) [27, 32]. Auxiliary Classifier GAN (AC-GAN) [29] is the state-of-the-art solution to integrate supervised information by feeding it into the generator and adding a loss function to account for the supervised information in the discriminator.

Existing supervised generative models only incorporate pointwise supervised information, e.g. class labels or text descriptions. However, in many real retrieval applications, we only have pairwise similarity information for training hashing models [40, 20, 42, 5, 6]. Deep Semantic Hashing (DSH) [30] is the first hashing method that explores GANs for image synthesis, but it can only incorporate pointwise side information (class labels) which is often unavailable in online image retrieval applications. Different from previous methods, we propose a new HashGAN architecture, which consists of a specifically-designed Pair Conditional Wasserstein GAN (PC-WGAN) that enables incorporation of pairwise similarity information for generating diverse synthetic images, and a deep hashing network trained with both real and synthetic images to generate nearly lossless hash codes.

3. HashGAN

In similarity retrieval systems, we are given N training points $\mathcal{X} = \{\boldsymbol{x}_i\}_{i=1}^N$, where some pairs of points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are given with *pairwise* 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. The goal of deep learning to hash is to learn nonlinear hash function $F : \boldsymbol{x} \mapsto \boldsymbol{h} \in \{-1, 1\}^K$ from input space 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 $\mathcal{S}$ between the given pairs can be preserved in the compact hash codes. In supervised hashing, the similarity pairs $\mathcal{S} = \{s_{ij}\}$ can be constructed from semantic labels of data points or relevance feedback from click-through data in online search systems.

This paper presents **HashGAN**, a deep learning to hash architecture with a novel Pair Conditional Wasserstein GAN (PC-WGAN) specifically designed for generative learning from images with pairwise similarity information. Figure 1 shows the architecture of HashGAN, which consists of two main components. **(1)** A pair conditional Wasserstein GAN (PC-WGAN), which takes as inputs the training images and pairwise similarity and jointly learns a generator G and a discriminator D: the generator G accepts as input the concatenation of a random noise $\boldsymbol{u}$ and an embedding vector $\boldsymbol{v}$ that encodes the similarity information to synthesize nearly real images; the discriminator D tries to distinguish the real and synthetic images using the adversarial loss. **(2)** A hash encoder F, which generates compact binary hash codes $\boldsymbol{h}$ for all images in a Bayesian learning framework: the framework jointly preserves the similarity information of both the real and synthetic images by a cosine cross-entropy loss and controls the quantization error by a cosine quantization loss.

3.1. Pair Conditional WGAN

The training strategy of generative adversarial networks (GANs) [14] defines a minimax game between two competing networks: a generator network G that captures underlying data distribution of real images for synthesizing images, and a discriminator network D that distinguishes the real images from synthetic images. Specifically, the generator G accepts as input a random noise $\boldsymbol{u}$, which is sampled from some simple noise distribution such as uniform distribution or spherical Gaussian distribution, and synthesizes a fake image $\tilde{\boldsymbol{x}} = G(\boldsymbol{u})$; the discriminator D takes as inputs either a real image $\boldsymbol{x}$ or a synthetic image $\tilde{\boldsymbol{x}}$ and must distinguish them by minimizing the classification error of probabilities $D(\boldsymbol{x})$ and $D(\tilde{\boldsymbol{x}})$. To tackle the training difficulty of GANs, [15] proposes an improved training strategy of Wasserstein GAN (WGAN) [1], which trains the discriminator through the Wasserstein distance that is continuous everywhere and differentiable almost everywhere, and proposes to enforce a differentiable Lipschitz constraint with gradient penalty as

$$\begin{aligned} \min_D L_D = & \mathop{\mathbb{E}}_{\tilde{\boldsymbol{x}} \sim \mathbb{P}_g} \left[D\left(\tilde{\boldsymbol{x}}\right) \right] - \mathop{\mathbb{E}}_{\boldsymbol{x} \sim \mathbb{P}_r} \left[D\left(\boldsymbol{x}\right) \right] \\ & + \gamma \mathop{\mathbb{E}}_{\hat{\boldsymbol{x}} \sim \mathbb{P}_{\hat{\boldsymbol{x}}}} \left[\left(\left\| \nabla_{\hat{\boldsymbol{x}}} D(\hat{\boldsymbol{x}}) \right\|_2 - 1 \right)^2 \right], \end{aligned} \tag{1}$$

where γ is the penalty coefficient typically set as $\gamma = 10$, $\mathbb{P}_r$ is the real data distribution, $\mathbb{P}_g$ is the generator distribution implicitly defined by $\tilde{\boldsymbol{x}} = G(\boldsymbol{u})$, and $\mathbb{P}_{\hat{\boldsymbol{x}}}$ is implicitly defined as sampling uniformly along straight lines between pairs of points sampled from the real data distribution $\mathbb{P}_r$

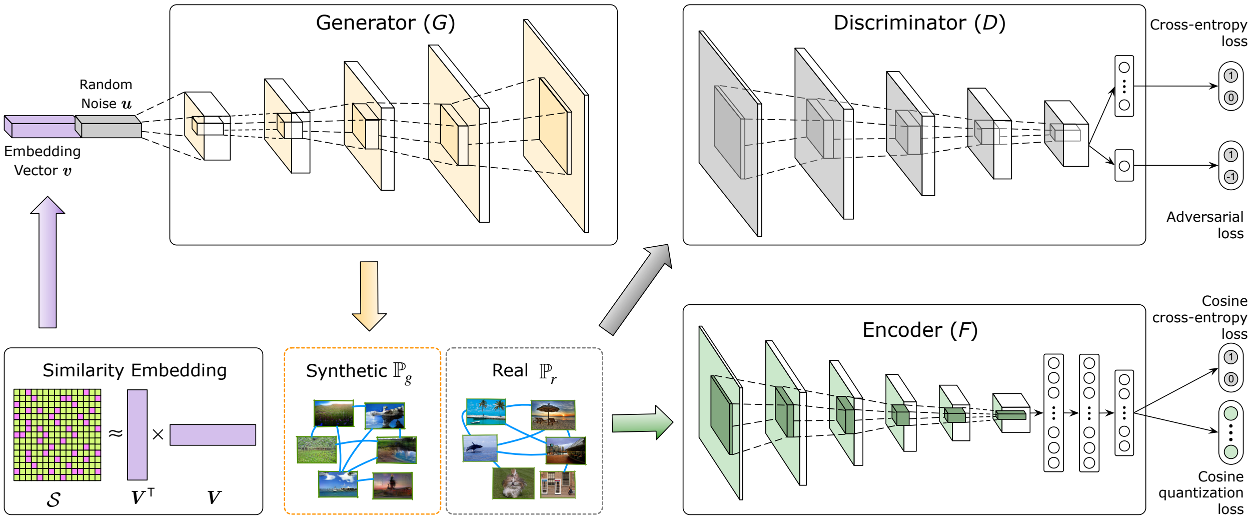


Figure 1. HashGAN for deep learning to hash with a new Pair Conditional Wasserstein GAN (PC-WGAN). The architecture of the proposed HashGAN consists of two main components. **(1)** A pair conditional Wasserstein GAN (PC-WGAN), which takes as inputs the training images and pairwise similarity and jointly learns a generator G and a discriminator D: the generator G accepts as input the concatenation of a random noise $\boldsymbol{u}$ and an embedding vector $\boldsymbol{v}$ that encodes the similarity information to synthesize nearly real images; the discriminator D tries to distinguish the real and synthetic images using the adversarial loss. **(2)** A hash encoder F, which generates compact binary hash codes $\boldsymbol{h}$ for all images in a Bayesian learning framework: the framework jointly preserves the similarity information of both the real and synthetic images by a cosine cross-entropy loss and minimizes the quantization error by a cosine quantization loss. *Best viewed in color.*

and the generator distribution $\mathbb{P}_g$. In the minimax game, the generator is trained to maximize probability of classifying synthetic images as real, which is equivalent to minimizing

$$\min_G L_G = \mathop{\mathbb{E}}_{\boldsymbol{u}\sim\mathbb{P}_{\boldsymbol{u}}} \left[-D\left(G(\boldsymbol{u})\right)\right], \tag{2}$$

where $\boldsymbol{u}$ is a random noise sampled from some simple noise distribution $\mathbb{P}_{\boldsymbol{u}}$. The goal of the generator is to maximally fool the discriminator with nearly real synthesized images. This improved WGAN [15] performs better than standard WGAN and enables stable and efficient training of various GAN architectures with almost no hyper-parameter tuning.

While WGAN with stabilized training strategy [15] can synthesize good images from random noises, it cannot take the advantage of useful side information. Class conditional synthesis can significantly improve the quality of generated samples [29], by supplying both the generator and discriminator with class labels to produce class conditional samples [27]. The auxiliary classifier GAN (AC-GAN) [29] is the state-of-the-art solution to integrate side information, which enables conditioning by feeding the supervised information into the generator and adding a new loss function with the supervised information in the discriminator. The generator synthesizes images from the inputs combined of supervised information and random noise, and the discriminator jointly distinguishes different classes as well as real from synthetic.

A major disadvantage of AC-GAN is that it can only be conditioned on point-wise supervised information, such as class labels or feature vectors from counterpart modalities. However, in deep learning to hash, we only have data $\mathcal{X} = \{\boldsymbol{x}_i\}_{i=1}^N$ with pairwise similarity information $\mathcal{S} = \{s_{ij}\}$. A naive solution to applying AC-GAN on data with pairwise information is that for each image $\boldsymbol{x}_i$, use each row $\boldsymbol{s}_{i\cdot} \in \mathbb{R}^N$ of the similarity matrix $\mathcal{S}$ as the supervised information. Unfortunately, this solution is infeasible since the number N of training points is usually larger than several thousands in deep learning to hash, whereas AC-GAN with such high-dimensional inputs cannot be trained successfully. Hence, it still remains an open problem how to enable GANs and WGANs conditioned on pairwise supervised information.

In this paper, we propose Pair Conditional WGAN (PC-WGAN), a new extension of WGAN to learn from data with pairwise supervised information $\{\mathcal{X}, \mathcal{S}\}$. At first, we reduce the high-dimension of pointwise supervised information $\boldsymbol{s}_{i\cdot}$ by a similarity embedding approach, which embeds the similarity information $\boldsymbol{s}_{i\cdot}$ associated with each image to a low-dimensional vector $\boldsymbol{v}_i \in \mathbb{R}^V$. The similarity embedding can be attained by minimizing the following reconstruction loss

$$\min_{\boldsymbol{v}_i \geqslant \boldsymbol{0}|_{i=1}^N} L_V = \sum_{s_{ij}\in\mathcal{S}} \left(s_{ij} - \boldsymbol{v}_i^{\mathsf{T}}\boldsymbol{v}_j\right)^2, \tag{3}$$

where L_V is the similarity embedding loss, and the nonnegative constraints are imposed to make the latent embeddings consistent with prior supervised information, which is given as nonnegative similarity labels $\{s_{ij}\}$. Since $s_{ij} \approx \boldsymbol{v}_i^{\mathsf{T}}\boldsymbol{v}_j$, each embedding vector $\boldsymbol{v}_i$ can approximately represent the similarity information of each point $\boldsymbol{x}_i$ with V-dimension, which is low-dimensional and can be fed as inputs to GANs.

In PC-WGAN, each generated point has a corresponding embedding vector $\boldsymbol{v}_i \sim \mathbb{P}_{\boldsymbol{v}}$ in addition to the random noise

$\boldsymbol{u}_i \in \mathbb{R}^V$. The generator uses both embedding vector and random noise to generate every image as $\tilde{\boldsymbol{x}}_i = G(\boldsymbol{v}_i, \boldsymbol{u}_i)$. The discriminator should give two probability distributions: one over the synthetic vs real as $D\left(\tilde{\boldsymbol{x}}\right)$ and $D\left(\boldsymbol{x}\right)$ for binary classification, and another over the similar vs dissimilar in all image pairs as $C\left(\tilde{\boldsymbol{x}}_i, \boldsymbol{x}_j\right)$ and $1 - C\left(\tilde{\boldsymbol{x}}_i, \boldsymbol{x}_j\right)$ for pairwise classification. More specifically, the discriminator network (except the last classifier layer) is shared between D and C. Denote by $\tilde{\boldsymbol{z}}$ and $\boldsymbol{z}$ the last-layer activations of network C for pairwise classification then $C\left(\tilde{\boldsymbol{x}}_i, \boldsymbol{x}_j\right) = \frac{1}{1+\exp\left(-\tilde{\boldsymbol{z}}_i^{\mathsf{T}} \boldsymbol{z}_j\right)}$. The overall loss for training discriminator of PC-WGAN is

$$
\begin{aligned}
\min_{D,C} L_{D,C} = & \mathop{\mathbb{E}}_{\tilde{\boldsymbol{x}} \sim \mathbb{P}_g} \left[D\left(\tilde{\boldsymbol{x}}\right)\right] - \mathop{\mathbb{E}}_{\boldsymbol{x} \sim \mathbb{P}_r} \left[D\left(\boldsymbol{x}\right)\right] \\
& + \gamma \mathop{\mathbb{E}}_{\hat{\boldsymbol{x}} \sim \mathbb{P}_{\hat{\boldsymbol{x}}}} \left[\left(\left\|\nabla_{\hat{\boldsymbol{x}}} D(\hat{\boldsymbol{x}})\right\|_2 - 1\right)^2\right] \\
& - \sum_{s_{ij} \in \mathcal{S}} s_{ij} \log C\left(\tilde{\boldsymbol{x}}_i, \boldsymbol{x}_j\right) \\
& - \sum_{s_{ij} \in \mathcal{S}} \left(1 - s_{ij}\right) \log\left(1 - C\left(\tilde{\boldsymbol{x}}_i, \boldsymbol{x}_j\right)\right),
\end{aligned}
\tag{4}
$$

where the third and fourth rows are the cross-entropy loss between probability $C\left(\tilde{\boldsymbol{x}}_i, \boldsymbol{x}_j\right)$ and pairwise similarity s_{ij}. In the minimax game, the generator is trained to maximize probabilities of synthetic being real as well as similar being dissimilar or vice versa, which is equivalent to minimizing

$$
\begin{aligned}
\min_{G,V} L_{G,V} = & \mathop{\mathbb{E}}_{\substack{\boldsymbol{v} \sim \mathbb{P}_{\boldsymbol{v}} \\ \boldsymbol{u} \sim \mathbb{P}_{\boldsymbol{u}}}} \left[-D\left(G\left(\boldsymbol{v}, \boldsymbol{u}\right)\right)\right] + \sum_{s_{ij} \in \mathcal{S}} \left(s_{ij} - \boldsymbol{v}_i^{\mathsf{T}} \boldsymbol{v}_j\right)^2 \\
& - \sum_{s_{ij} \in \mathcal{S}} s_{ij} \log C\left(\tilde{\boldsymbol{x}}_i, \boldsymbol{x}_j\right) \\
& - \sum_{s_{ij} \in \mathcal{S}} \left(1 - s_{ij}\right) \log\left(1 - C\left(\tilde{\boldsymbol{x}}_i, \boldsymbol{x}_j\right)\right),
\end{aligned}
\tag{5}
$$

and note that $\tilde{\boldsymbol{x}}_i = G(\boldsymbol{v}_i, \boldsymbol{u}_i)$. The goal of the generator is to maximally fool the discriminator with synthetic images generated from the similarity embedding and random noise.

In applications, the size of training data with similarity information is remarkably smaller than the size of complete unlabeled data. We enable PC-WGAN to learn from both labeled data and unlabeled data to synthesize high-quality images by further using zero embedding vector $\boldsymbol{v}_j = \boldsymbol{0}$ for each unlabeled image $\boldsymbol{x}_j \notin \mathcal{X}$. The generator distribution $\mathbb{P}_g$ changes to $G(\boldsymbol{v}_i, \boldsymbol{u}_i) \cup G(\boldsymbol{0}, \boldsymbol{u}_j)$, and $\mathbb{P}_r$ becomes distributions of both supervised and unsupervised real images. Though both $\mathbb{P}_g$ and $\mathbb{P}_r$ are changed due to unlabeled data, the PC-WGAN objectives in (4) and (5) remain unchanged.

3.2. Deep Learning to Hash

The PC-WGAN trained on images with pairwise similarity information can generate high-quality synthetic images, which can be used to boost the performance of deep learning to hash over images with insufficient similarity labels. In this paper, we propose a hash encoder network F, which generates compact hash codes for both synthetic and real images in a Bayesian framework. The hash encoder consists of three components: **(1)** a deep convolutional network (CNN) for learning deep compact codes $\boldsymbol{h}_i = F(\bar{\boldsymbol{x}}_i)$ for each input image $\bar{\boldsymbol{x}}_i$, where $\bar{\boldsymbol{x}}_i$ can be a real image $\boldsymbol{x}$ with similarity information or a synthetic image $\tilde{\boldsymbol{x}}$ generated by PC-WGAN with similarity information; **(2)** a cosine cross-entropy loss for similarity-preserving hash learning; and **(3)** a cosine quantization loss for controlling quantization error.

Given training data $\mathcal{X} = \{\boldsymbol{x}_i\}_{i=1}^N$ and synthetic images $\tilde{\mathcal{X}} = \{\tilde{\boldsymbol{x}}_j\}_{j=1}^M$, we can expand the training data to $\mathcal{X} \cup \tilde{\mathcal{X}}$ and the similarity labels to $\mathcal{S} = \{s_{ij}\}_{i,j=1}^{N+M}$ for deep hashing. The logarithm Maximum a Posteriori (MAP) estimation of hash codes $\boldsymbol{H} = [\boldsymbol{h}_1, \ldots, \boldsymbol{h}_{N+M}]$ given $\mathcal{S}$ and $\mathcal{X} \cup \tilde{\mathcal{X}}$ is

$$
\begin{aligned}
& \log P\left(\boldsymbol{H} | \mathcal{S}\right) \propto \log P\left(\mathcal{S} | \boldsymbol{H}\right) P\left(\boldsymbol{H}\right) \\
& \qquad = \sum_{s_{ij} \in \mathcal{S}} w_{ij} \log P\left(s_{ij} | \boldsymbol{h}_i, \boldsymbol{h}_j\right) + \sum_{i=1}^{N+M} \log P\left(\boldsymbol{h}_i\right),
\end{aligned}
\tag{6}
$$

where $P\left(\mathcal{S} | \boldsymbol{H}\right) = \prod_{s_{ij} \in \mathcal{S}} \left[P\left(s_{ij} | \boldsymbol{h}_i, \boldsymbol{h}_j\right)\right]^{w_{ij}}$ is weighted likelihood function, and w_{ij} is the weight for each training pair $(\bar{\boldsymbol{x}}_i, \bar{\boldsymbol{x}}_j, s_{ij})$, which tackles the data imbalance problem by weighting the training pairs according to the importance of misclassifying that pair [8]. Since each similarity label in $\mathcal{S}$ can only be $s_{ij} = 1$ or $s_{ij} = 0$, to account for the data imbalance between similar and dissimilar pairs, we propose

$$
w_{ij} = \begin{cases} \left|\mathcal{S}\right| / \left|\mathcal{S}_1\right|, & s_{ij} = 1 \\ \left|\mathcal{S}\right| / \left|\mathcal{S}_0\right|, & s_{ij} = 0 \end{cases}
\tag{7}
$$

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 as pairwise logistic function,

$$
\begin{aligned}
P\left(s_{ij} | \boldsymbol{h}_i, \boldsymbol{h}_j\right) & = \begin{cases} \sigma\left(\cos\left(\boldsymbol{h}_i, \boldsymbol{h}_j\right)\right), & s_{ij} = 1 \\ 1 - \sigma\left(\cos\left(\boldsymbol{h}_i, \boldsymbol{h}_j\right)\right), & s_{ij} = 0 \end{cases} \\
& = \sigma(\cos\left(\boldsymbol{h}_i, \boldsymbol{h}_j\right))^{s_{ij}} \left(1 - \sigma\left(\cos\left(\boldsymbol{h}_i, \boldsymbol{h}_j\right)\right)\right)^{1 - s_{ij}}
\end{aligned}
\tag{8}
$$

where $\sigma\left(h\right) = 1/(1 + e^{-\alpha h})$ is adaptive sigmoid function. Similar to logistic regression, we can see that the smaller the Hamming distance $\mathrm{dist}_H\left(\boldsymbol{h}_i, \boldsymbol{h}_j\right)$ is, the larger the cosine similarity $\cos\left(\boldsymbol{h}_i, \boldsymbol{h}_j\right)$ as well as the conditional probability $P\left(1 | \boldsymbol{h}_i, \boldsymbol{h}_j\right)$ will be, implying that the image pair $\bar{\boldsymbol{x}}_i$ and $\bar{\boldsymbol{x}}_j$ should be classified as similar; otherwise, the larger the conditional probability $P\left(0 | \boldsymbol{h}_i, \boldsymbol{h}_j\right)$ will be, implying that the image pair $\bar{\boldsymbol{x}}_i$ and $\bar{\boldsymbol{x}}_j$ should be classified as dissimilar. Hence, Equation (8) is a reasonable extension of the logistic regression classifier to the pairwise classification scenario, which is optimal for binary similarity labels $s_{ij} \in \{0, 1\}$.

Since discrete optimization of Equation (6) with binary constraints $\boldsymbol{h}_i \in \{-1,1\}^K$ is very challenging, continuous relaxation $\boldsymbol{h}_i \in \mathbb{R}^K$ is applied to the binary constraints for ease of optimization, as adopted by most hashing methods [37, 42]. To control the quantization error $\|\boldsymbol{h}_i - \operatorname{sgn}(\boldsymbol{h}_i)\|$ of continuous relaxation and close the gap between Hamming distance and cosine distance for learning high-quality hash codes, we propose a novel bimodal Gaussian prior for $\boldsymbol{h}_i$ as

$$P\left(\boldsymbol{h}_i\right) = \frac{1}{2\epsilon} \exp\left(-\frac{1}{\epsilon}\left\|\frac{|\boldsymbol{h}_i|}{\|\boldsymbol{h}_i\|} - \frac{\mathbf{1}}{\sqrt{K}}\right\|_2^2\right), \tag{9}$$

where ϵ is the diversity parameter of bimodal Gaussian distribution, and $\mathbf{1} \in \mathbb{R}^K$ is the vector of ones with norm $\sqrt{K}$.

By taking Equations (8) and (9) into the MAP estimation in Equation (6), we obtain the optimization problem of the hash encoder F for learning compact hash codes as follows

$$\begin{aligned} \min_F L_F = & \sum_{s_{ij} \in \mathcal{S}} w_{ij} \log\left(1 + \exp\left(\alpha \cos\left(\boldsymbol{h}_i, \boldsymbol{h}_j\right)\right)\right) \\ & - \sum_{s_{ij} \in \mathcal{S}} w_{ij} s_{ij} \alpha \cos\left(\boldsymbol{h}_i, \boldsymbol{h}_j\right) \\ & - \beta \sum_{i=1}^{N+M} \cos\left(\left|\boldsymbol{h}_i\right|, \mathbf{1}\right), \end{aligned} \tag{10}$$

β is the parameter to balance the weight between the cosine cross-entropy loss in the first and second rows of Eq. (10) and the cosine quantization loss in the third row of (10).

3.3. HashGAN Optimization

This paper establishes deep learning to hash for images with pairwise similarity information, which constitutes two key components: Pair Conditional Wasserstein GAN (PC-WGAN) for generating nearly real images and Deep Hash Encoder for generating compact hash codes for each image. The overall optimization problem is a unified integration of the PC-WGAN objective in Equations (4) (5) and the Deep Hash Encoder objective in Equation (10). As the proposed HashGAN architecture is a variant of GANs, the two-player minimax game mechanism is adopted for the optimization. The optimization problems for discriminator D, generator G and hash encoder F are respectively computed as follows

$$\begin{aligned} & \min_{D,C} L_F + \lambda L_{D,C}, \\ & \min_{G,V} L_F + \lambda L_{G,V}, \\ & \min_{F} L_F, \end{aligned} \tag{11}$$

where λ is a parameter to trade of the importance of deep hash encoder and PC-WGAN. The network parameters can be efficiently optimized through standard back-propagation using automatic differentiation techniques by TensorFlow.

Finally, we obtain hash code for each image by simple binarization $\boldsymbol{h} \leftarrow \operatorname{sgn}(\boldsymbol{h})$, where $\operatorname{sgn}(\cdot)$ is the sign function. Through the minimax optimization in Equation (11), we can synthesize nearly real images with pairwise information by the proposed PC-WGAN, and generate nearly lossless hash codes by similarity-preserving learning and quantization error minimization from both real and synthetic images. It is worth noting that, we can alleviate the difficulty in learning with insufficient supervised information by using both real and synthetic data for deep learning to hash, which yields higher quality hash codes for improved search performance.

4. Experiments

We evaluate the efficacy of the proposed HashGAN approach with 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 [7] is a public image dataset which contains 269,648 images in the 81 ground truth categories. We follow similar experimental protocols in [42, 6], 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.

CIFAR-10 is a public dataset with 60,000 images in 10 classes. We follow protocol in [5] to randomly select 100 images per class as the query set, 500 images per class as the training set, and the rest images are used as the database.

MS-COCO [23] is a widely-used image 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.

Following standard evaluation protocol as previous work [40, 20, 42, 6], the similarity information for hash function 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 $s_{ij} = 0$. Though we use the ground truth image labels to construct the similarity information, the proposed HashGAN can learn compact binary hash codes when only the similarity information is available, which is more general than many label-information based hashing methods [38, 3].

We compare retrieval performance of **HashGAN** with eight state-of-the-art hashing methods, including supervised shallow hashing methods **BRE** [19], **ITQ-CCA** [13], **KSH** [25], **SDH** [34], and supervised deep hashing methods **CNNH** [40], **DNNH** [20], **DHN** [42] and **HashNet** [6]. We evaluate retrieval quality based on four standard evaluation metrics: Mean Average Precision (**MAP**), Precision-Recall

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

Method	NUS-WIDE				CIFAR-10				MS-COCO			
	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 [13]	0.460	0.405	0.373	0.347	0.354	0.414	0.449	0.462	0.566	0.562	0.530	0.502
BRE [19]	0.503	0.529	0.548	0.555	0.370	0.438	0.468	0.491	0.592	0.622	0.630	0.634
KSH [25]	0.551	0.582	0.612	0.635	0.524	0.558	0.567	0.569	0.521	0.534	0.534	0.536
SDH [34]	0.588	0.611	0.638	0.667	0.461	0.520	0.553	0.568	0.555	0.564	0.572	0.580
CNNH [40]	0.570	0.583	0.593	0.600	0.476	0.472	0.489	0.501	0.564	0.574	0.571	0.567
DNNH [20]	0.598	0.616	0.635	0.639	0.559	0.558	0.581	0.583	0.593	0.603	0.605	0.610
DHN [42]	0.637	0.664	0.669	0.671	0.568	0.603	0.621	0.635	0.677	0.701	0.695	0.694
HashNet [6]	0.662	0.699	0.711	0.716	0.643	0.667	0.675	0.687	0.687	0.718	0.730	0.736
HashGAN	**0.715**	**0.737**	**0.744**	**0.748**	**0.668**	**0.731**	**0.735**	**0.749**	**0.697**	**0.725**	**0.741**	**0.744**

curves (**PR**), Precision curves within Hamming distance 2 (**P@H$\leq$2**), and Precision curves with respect to the numbers of top returned samples (**P@N**). For direct comparison to published results, all methods use identical training and test sets. We follow HashNet [6] and DHN [42] and adopt MAP@5000 for NUS-WIDE dataset, MAP@5000 for MS-COCO dataset, and MAP@54000 for CIFAR-10 dataset.

For shallow hashing methods, we use as image features the 4096-dimensional DeCAF$_7$ features [9]. For deep hashing methods, we use as input the original images, and adopt AlexNet [18] as the backbone architecture. We follow [15] and adopt a four-layer ResNet [16] architecture for the discriminator and generator in HashGAN, which is proved to generate high quality images with 64×64 pixels. We adopt AlexNet [18] as the hash encoder, fine-tune all layers but the last one copied from the pre-trained AlexNet. 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 as the solver, 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 cross-validate the dimension of the embedding vector $\boldsymbol{v}$, and observe that fixing this hyper-parameter as 32 is enough to achieve satisfiable results. Also, HashGAN is not sensitive to different dimensions given that the dimension of $\boldsymbol{v}$ is large enough, e.g. 32. We select the parameters of all comparison methods through cross-validation on training data.

4.2. Results

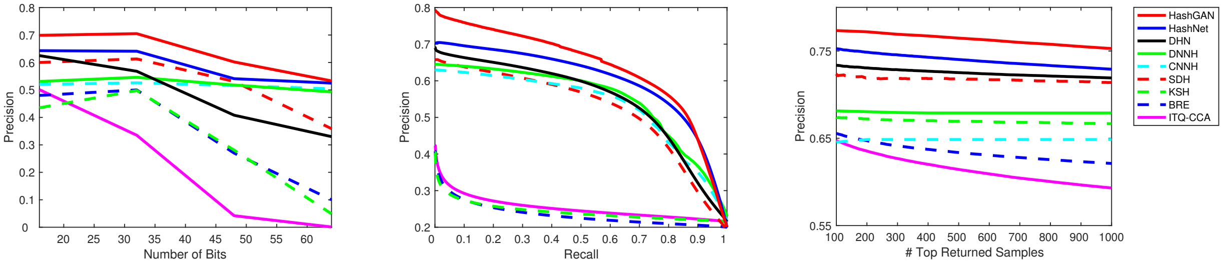
The MAP results of all methods are demonstrated in Table 1, which shows that the proposed HashGAN substantially outperforms all the comparison methods by large margins. Specifically, compared to SDH, the best shallow hashing method with deep features as input, HashGAN achieves absolute increases of **11.0%**, **19.5%** and **15.9%** in average MAP on NUS-WIDE, CIFAR-10, and MS-COCO respectively. HashGAN outperforms HashNet, the state-of-the-art deep hashing method, by large margins of **3.9%**, **5.3%** and **0.9%** in average MAP on the three datasets, respectively.

The MAP results reveal several interesting insights. **(1)** Shallow hashing methods cannot learn discriminative deep representations 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 jointly preserving similarity information and controlling the quantization error, which significantly outperform pioneering methods CNNH and DNNH without reducing the quantization error.

The proposed HashGAN improves substantially from the state-of-the-art HashNet by two important perspectives: **(1)** HashGAN integrates a novel Pair Conditional Wasserstein GAN (PC-WGAN) to synthesize nearly real images as training data, which substantially increases the diversity of training data and alleviates the technical difficulty of insufficient supervised information in many real applications. **(2)** HashGAN adopts new cosine cross-entropy loss and cosine quantization loss, which can approximate the Hamming distance more accurately to learn nearly lossless hash codes.

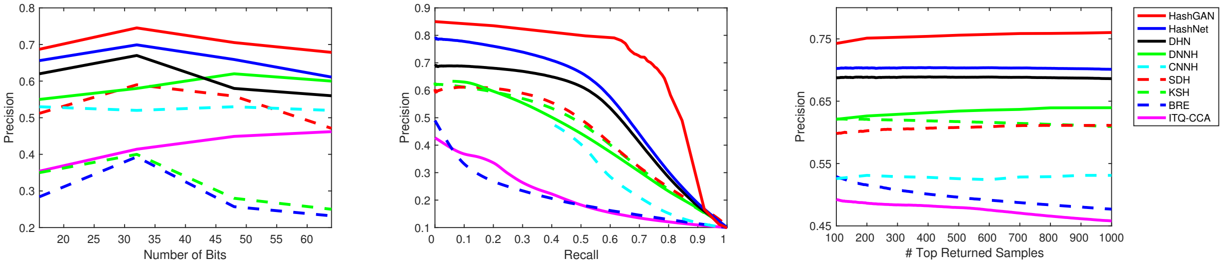
The performance in Precision within Hamming radius 2 (P@H$\leq$2) is very important for efficient image retrieval since such Hamming ranking only requires $O(1)$ time cost for each query, which enables really fast pruning. As shown in Figures 2(a), 3(a) and 4(a), HashGAN achieves the highest P@H$\leq$2 results on all three benchmark datasets using different numbers of bits. This validates that HashGAN can learn compacter hash codes than all comparison methods to establish more efficient and accurate Hamming ranking. When using longer hash codes, the Hamming space will become higher-dimensional and more sparse such that fewer data points will fall in the Hamming ball within radius 2. This explains why most existing hashing methods perform worse in terms of P@H$\leq$2 criterion with longer hash codes.

The retrieval performance in terms of Precision-Recall curves (PR) and Precision curves with respect to different numbers of top returned samples (P@N) are demonstrated in Figures 2(b), 3(b), 4(b) and 2(c), 3(c), 4(c), respectively. The proposed HashGAN significantly outperforms all comparison methods by large margins under these two evaluation metrics. In particular, HashGAN achieves much higher



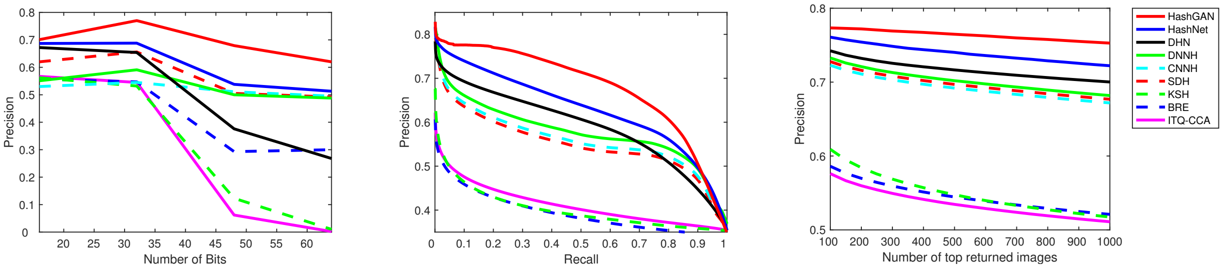
(a) Precision within Hamming radius 2 (b) Precision-recall curve @ 64 bits (c) Precision curve w.r.t. top-N @ 64 bits

Figure 2. The experimental results of HashGAN and comparison methods on the NUS-WIDE dataset under three evaluation metrics.



(a) Precision within Hamming radius 2 (b) Precision-recall curve @ 64 bits (c) Precision curve w.r.t. top-N @ 64 bits

Figure 3. The experimental results of HashGAN and comparison methods on the CIFAR-10 dataset under three evaluation metrics.



(a) Precision within Hamming radius 2 (b) Precision-recall curve @ 64 bits (c) Precision curve w.r.t. top-N @ 64 bits

Figure 4. The experimental results of HashGAN and comparison methods on the MS-COCO dataset under three evaluation metrics.

precision at lower recall levels or smaller number of top samples. This is very desirable for precision-first retrieval, which is widely implemented in practical retrieval systems.

4.3. Analysis

4.3.1 Ablation Study

We investigate four variants of HashGAN: (1) **HashGAN-B** is the HashGAN variant without binarization ($\boldsymbol{h} \leftarrow sgn(\boldsymbol{h})$ is not performed), which may serve as the upper bound of retrieval performance; (2) **HashGAN-Q** is the HashGAN variant without using the proposed quantization loss (10), in other words $\beta = 0$; (3) **HashGAN-C** is the HashGAN variant by replacing the cosine cross-entropy loss in (10) with the widely-used inner-product cross-entropy loss [42, 6]; (4) **HashGAN-G** is the HashGAN variant by removing the proposed Pair Conditional Wasserstein GAN (PC-WGAN), i.e. only the hash encoder is trained to generate hash codes and $\lambda = 0$. The MAP results with respect to different code lengths on three benchmark datasets are reported in Table 2.

Pair Conditional Wasserstein GAN. Table 2 shows that HashGAN significantly outperforms HashGAN-G by large margins of 2.5%, 3.8% and 1.8% in average MAP on three datasets, respectively. Without generating high-quality and nearly real synthetic images using PC-WGAN, the diversity of training images for deep learning to hash may be limited, which will lead to overfitting when the pairwise similarity information is insufficient and to worse search performance. The proposed PC-WGAN turns out to be the most important underpinning, which helps HashGAN achieve the state-of-the-art retrieval performance in various evaluation metrics. Besides, we feed the images generated by PC-WGAN to the best baselines DHN [42] and HashNet [6], which can also outperform the traditional methods by 2.0% on average.

Cosine Cross-Entropy Loss. Table 2 shows that HashGAN outperforms HashGAN-C by 2.0%, 2.5% and 1.9% in average MAP on the three datasets. HashGAN-C uses the widely-used inner-product cross-entropy loss [42, 6] which achieves state-of-the-art results on previous retrieval tasks. In real search engines, cosine similarity is widely used to

Table 2. Mean Average Precision (MAP) Results of HashGAN and Its Variants HashGAN-B, HashGAN-Q, HashGAN-C, and HashGAN-G

Method	NUS-WIDE				CIFAR-10				MS-COCO			
	16 bits	32 bits	48 bits	64 bits	16 bits	32 bits	48 bits	64 bits	16 bits	32 bits	48 bits	64 bits
HashGAN-B	**0.745**	**0.758**	**0.773**	**0.788**	**0.693**	**0.748**	**0.754**	**0.768**	**0.723**	**0.748**	**0.754**	**0.766**
HashGAN	0.715	0.737	0.744	0.748	0.668	0.731	0.735	0.749	0.697	0.725	0.741	0.744
HashGAN-Q	0.697	0.718	0.727	0.736	0.646	0.713	0.722	0.731	0.667	0.698	0.714	0.722
HashGAN-C	0.703	0.709	0.723	0.727	0.659	0.703	0.708	0.713	0.676	0.713	0.720	0.723
HashGAN-G	0.693	0.713	0.715	0.721	0.653	0.681	0.693	0.702	0.678	0.711	0.721	0.726

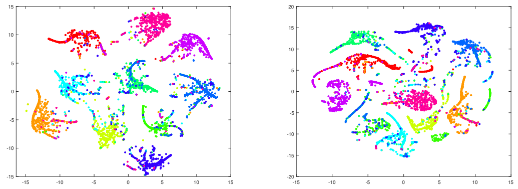

(a) HashGAN (b) HashNet

Figure 5. The t-SNE visualizations of the hash codes respectively learned by HashGAN and HashNet on the CIFAR-10 dataset.

mitigate the diversity of vector lengths and improve retrieval quality, while it has not been integrated with cross-entropy loss for supervised hash learning [37]. We propose a novel cosine cross-entropy loss (10) based on cosine similarity, which can better approximate the Hamming distance and preserve the similarity information of training image pairs.

Cosine Quantization Loss. By jointly optimizing the cosine cross-entropy loss and the cosine quantization loss over deep representations of both real and synthetic images, HashGAN incurs small average MAP decreases of 3.0%, 2.0%, and 2.6% when binarizing continuous representations of HashGAN-B. In contrast, without optimizing the cosine quantization loss (10), HashGAN-Q suffers from very large MAP decreases of 4.7%, 3.8%, and 5.3%, and substantially underperforms HashGAN. These results in Table 2 validate that the cosine quantization loss (10) can effectively reduce the binarization error and yield nearly lossless hash coding.

4.3.2 Visualization Study

Visualization of Hash Codes by t-SNE. Figure 5 shows the t-SNE visualizations [35] of the hash codes learned by the proposed HashGAN approach and the best deep hashing method HashNet [6] on the CIFAR-10 dataset. We observe that the hash codes learned by HashGAN exhibit clear discriminative structures where the hash codes in different categories are well separated, while the hash codes generated by HashNet exhibit relative vague structures. This validates that by introducing the novel pair conditional WGAN into deep hashing, the hash codes generated through the proposed HashGAN are more discriminative than that generated by HashNet, enabling more accurate image retrieval.

Visualization of Synthetic Images by HashGAN. Figure 6 illustrates the per-class image examples on CIFAR-10 dataset, which are synthetic images generated by HashGAN (left) and real images randomly selected from the dataset (right). We can observe that the synthetic images are nearly real and semantically relevant to each class but are much more diverse, which can improve the quality of hash codes.

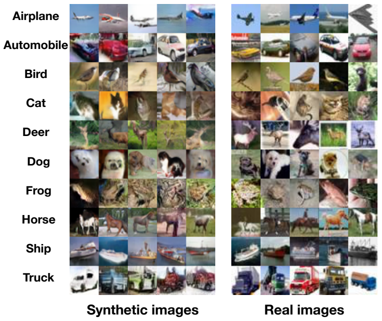


Figure 6. Visualization of image examples on CIFAR-10.

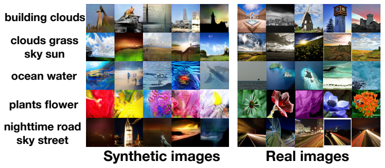


Figure 7. Visualization of image examples on NUS-WIDE.

Figure 7 illustrates synthetic (left) and real (right) image samples on NUS-WIDE, which is a multi-label (81 labels) dataset with high-resolution images and is more difficult to generate high-quality images. In both cases, HashGAN can generate plausible images to improve retrieval performance.

5. Conclusion

This paper tackles deep learning to hash with insufficient similarity information by image synthesis from generative models. The proposed HashGAN can synthesize nearly real images conditioned on the pairwise similarity information, with more diverse synthesized images to improve the quality of compact binary hash codes. Extensive empirical results demonstrate that HashGAN can yield state-of-the-art multimedia retrieval performance on standard benchmarks.

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] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein generative adversarial networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pages 214–223, 2017. 2

[2] Y. Cao, M. Long, J. Wang, and S. Liu. Collective deep quantization for efficient cross-modal retrieval. In *AAAI*, pages 3974–3980. AAAI Press, 2017. 1

[3] Y. Cao, M. Long, J. Wang, and S. Liu. Deep visual-semantic quantization for efficient image retrieval. In *CVPR*, 2017. 5

[4] 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. 1

[5] Y. Cao, M. Long, J. Wang, H. Zhu, and Q. Wen. Deep quantization network for efficient image retrieval. In *AAAI*. AAAI, 2016. 2, 5

[6] Z. Cao, M. Long, J. Wang, and P. S. Yu. Hashnet: Deep learning to hash by continuation. *ICCV*, 2017. 1, 2, 5, 6, 7, 8

[7] 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

[8] 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. 4

[9] 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. 6

[10] 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

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

[12] 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

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

[14] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014. 2

[15] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville. Improved training of wasserstein gans. *CoRR*, abs/1704.00028, 2017. 2, 3, 6

[16] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *CVPR*, 2016. 6

[17] 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

[18] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012. 6

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

[20] 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

[21] M. S. Lew, N. Sebe, C. Djeraba, and R. Jain. Content-based multimedia information retrieval: State of the art and challenges. *ACM Trans. Multimedia Comput. Commun. Appl.*, 2(1):1–19, Feb. 2006. 1

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

[23] 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

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

[25] 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

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

[27] M. Mirza and S. Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014. 2, 3

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

[29] A. Odena, C. Olah, and J. Shlens. Conditional image synthesis with auxiliary classifier gans. *ICML*, 2017. 2, 3

[30] Z. Qiu, Y. Pan, T. Yao, and T. Mei. Deep semantic hashing with generative adversarial networks. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 225–234. ACM, 2017. 2

[31] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *ICLR*, 2015. 2

[32] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee. Generative adversarial text to image synthesis. *ICML*, 2016. 2

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

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

[35] 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

[36] 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

[37] 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, 5, 8

[38] X. Wang, T. Zhang, G.-J. Qi, J. Tang, and J. Wang. Supervised quantization for similarity search. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2018–2026, 2016. 5

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

[40] 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

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

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