Deep Unsupervised Hashing with Latent Semantic Components

Qinghong Lin[1]*, Xiaojun Chen[1]†, Qin Zhang[1], Shaotian Cai[1], Wenzhe Zhao[2], Hongfa Wang[2]

[1]Shenzhen University, Shenzhen, China
[2]Tencent Data Platform
linqinghong@email.szu.edu.cn, {xjchen, qinzhang}@szu.edu.cn, cai.st@foxmail.com, {carsonzhao, hongfawang}@tencent.com

Abstract

Deep unsupervised hashing has been appreciated in the regime of image retrieval. However, most prior arts failed to detect the semantic components and their relationships behind the images, which makes them lack discriminative power. To make up the defect, we propose a novel **Deep Semantic Components Hashing (DSCH)**, which involves a common sense that an image normally contains a bunch of semantic components with homology and co-occurrence relationships. Based on this prior, DSCH regards the semantic components as latent variables under the Expectation-Maximization framework and designs a two-step iterative algorithm with the objective of maximum likelihood of training data. Firstly, DSCH constructs a semantic component structure by uncovering the fine-grained semantics components of images with a Gaussian Mixture Modal (GMM), where an image is represented as a mixture of multiple components, and the semantics co-occurrence are exploited. Besides, coarse-grained semantics components, are discovered by considering the homology relationships between fine-grained components, and the hierarchy organization is then constructed. Secondly, DSCH makes the images close to their semantic component centers at both fine-grained and coarse-grained levels, and also makes the images share similar semantic components close to each other. Extensive experiments on three benchmark datasets demonstrate that the proposed hierarchical semantic components indeed facilitate the hashing model to achieve superior performance.

Introduction

With the explosive growth of social data such as images, how to conduct rapid similarity searches has become one of the basic requirements of large-scale information retrieval. Hashing (Wang et al. 2015) (Liu and Zhang 2016) has become a widely studied solution to this problem. The goal of hashing is to convert high-dimensional feature and similarity information into compact binary codes, which can greatly speed up computation with efficient xor operations and save storage space.

Hashing techniques can be generally divided into supervised and unsupervised categories. Supervised hashing methods (Liu et al. 2012) (Shen et al. 2015) (Yuan et al. 2020) use label information to train hashing models, which achieve reliable performances. However, obtaining adequate labeled data is normally expensive and impractical, which hinders the application of these methods. In this scenario, an increasing number of researchers turn their attention to unsupervised hashing methods (Liu et al. 2011) (Liu et al. 2014) (Shen et al. 2018) (Lin et al. 2021). With the lack of labels, the key to unsupervised hashing methods is the manual design of semantic similarity and how they guide the learning of hashing models. However, existed methods (Yang et al. 2018) (Song et al. 2018) (Deng et al. 2019) mainly develop the similarity information derived from the entire image, which fails to recognize the semantic composition of each image. Generally, a real-world picture is composed of different types of objects, which makes it contain rich semantic connotations. We call a type of object with specific semantic meaning as a **semantic component**. For instance, a picture of `walking a dog` can be comprised of the following components: `people`, `dog` and `grasses`, in which `dog` are usually correlated with `people` as human pets. We call this phenomenon semantic **co-occurrence**.

Besides the detection of the semantic components of im-

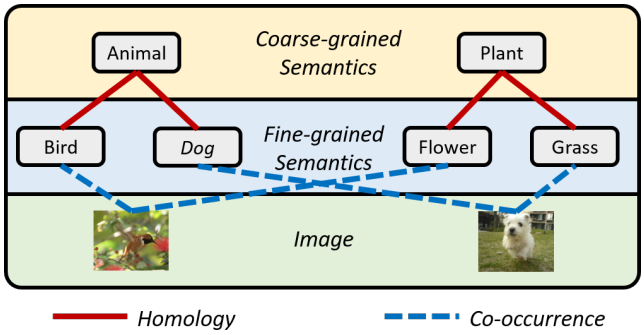


Figure 1: An illustration of semantic component structure, where a picture is normally composed of multiples interrelated semantic components. e.g, `dog` and `grass` are associated by an image, we call this relations as *co-occurrence*. Besides, these semantic components are hierarchically organized from fine to coarse scale, e.g, `dog` belongs to `animal`, this relationship known as *homology*.

*This work is done when Qinghong Lin is an intern at Tencent.
†Corresponding authors

ages and their co-occurrence relations, we notice that the **homology** relationships between them are another useful information, which is organized in a hierarchical way, such as `chihuahua` and `husky` both belong to category `dog`. This allows people to learn new concepts easily by inference its connection with learned concepts. For example, `husky` is another breed of `dog`, so it should have some similar biological qualities with chihuahua. The mentioned facts inspired us to introduce the homology of semantic components into the visual retrieval task. Several supervised hashing methods (Sun et al. 2019) (Wang et al. 2018) proved that the semantic hierarchy can benefit hashing models. But how to model it in an unsupervised way is still an open question.

Based on the above two observations, in this paper, we propose a novel unsupervised hashing framework, called **Deep Semantic Components Hashing (DSCH)**, which assumes that each image is composed of multiple latent semantic components and propose a two-step iterative algorithm to yield discriminative binary codes. Firstly, DSCH constructs a semantic component structure with fully exploring the homology and co-occurrence relations of semantic knowledge, and an example is shown in Figure. 1. It formulates each image as the mixture of multiple fine-grained semantic components with GMM, and clusters them to coarse-grained semantic components to construct the hierarchy. Secondly, DSCH makes the images close to their semantic component centers on both fine-grained and coarse-grained levels, and also makes the images share similar semantic components close to each other. These two steps can be unified into an Expectation-Maximization framework that iteratively optimizes model parameters with the maximum likelihood of the data. The main contributions of this paper can be summarized as follows:

- We propose a novel deep unsupervised hashing framework DSCH, which treats the semantic components of images as latent variables and learns hashing models based on a two-step iterative framework.
- We propose a semantic component structure, which represents images as the mixture of multiple semantic components considering their co-occurrence and homology relationships.
- We propose a hashing learning strategy by extending the contrastive loss to adapt semantic components, where images are pulled to their semantic component centroid and close to other images with similar semantic components.
- The extensive experiments on CIFAR-10, FLICKR25K, and NUS-WIDE datasets show that our DSCH is effective and achieves superior performance.

Notation and Problem Definition

Let us introduce some notations for this paper, we use boldface uppercase letters like $\mathbf{A}$ to represent the matrix, $\mathbf{a}_i$ represents the i-th row of $\mathbf{A}$, a_{ij} represents the i-th row and the j-th column element of $\mathbf{A}$ and $\mathbf{A}^T$ denotes the transpose of $\mathbf{A}$. $\|\cdot\|_2$ represents the l_2-norm of a vector. $\|\cdot\|_F$ represents the Frobenius norm of a matrix. $\mathrm{sgn}(\cdot)$ represents the sign function, which outputs 1 for positive numbers, or -1 otherwise. $\tanh(\cdot)$ represents the hyperbolic tangent function. $\cos(\mathbf{x},\mathbf{y}) \triangleq \frac{\mathbf{x}^T\mathbf{y}}{\|\mathbf{x}\|_2\|\mathbf{y}\|_2}$ represents the cosine similarity between vector $\mathbf{x}$ and vector $\mathbf{y}$.

Suppose we have n database images $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^n$ that contains n images without labels. The purpose of our method is to learn a hash function $\mathcal{H} : \mathbf{x}_i \rightarrow \mathbf{b}_i$ that mapping $\mathbf{X}$ into compact binary hash codes $\mathbf{B} = \{\mathbf{b}_i\}_{i=1}^n \in \{-1,+1\}^{n\times r}$, where r represents the length of hash codes.

Related Work

Unsupervised Hashing

There are many well-known traditional hashing methods been established in decades. Among them, Iterative Quantization (ITQ) (Gong et al. 2012) minimizes the quantization error of mapping by finding the optimal rotation of the data. Spectral Hashing (SH) (Weiss, Torralba, and Fergus 2009) construct a Laplacians graph with Euclidean distance to determine the pairwise code distances. To speed up the construction of the graph, Anchor Graph Hashing (AGH) (Liu et al. 2011) proposes a sparse low-rank graph by introducing a set of anchors. Although the aforementioned methods have made progress in this area, they are all shallow architectures that rely heavily on hand-crafted features. To tackle this problem, amount of deep unsupervised hashing methods (Deng et al. 2019) (Tu et al. 2020) (Tu et al. 2021a) (Tu et al. 2021b) have been proposed, in which Deep binary descriptors (DeepBit) (Lin et al. 2016) treats the image and its rotation as similar pairs in hash code learning. Semantic Structure-based unsupervised Deep Hashing (SSDH) (Yang et al. 2018) finds the cosine distance distribution of pairs based on Gaussian estimation to construct a semantic structure. Twin-Bottleneck Hashing (TBH) (Shen et al. 2020) introduces two bottlenecks that can collaboratively exchange meaningful information. Recently, inspired by the success in the unsupervised representation domain (He et al. 2020) (Chen et al. 2020), contrastive learning have been introduced to reinforce the discrimination power of binary codes (Luo et al. 2020), Contrastive Information Bottleneck (CIB) (Qiu et al. 2021) modifies the contrastive loss (Oord, Li, and Vinyals 2018) to meet the requirement of hashing learning as:

$$\mathcal{L}_0 = \frac{1}{n}\sum_{i=1}^{n}\left(l_i^a + l_i^b\right) \tag{1}$$

where

$$l_i^a = -\log\frac{e^{\cos(\mathbf{b}_i^a,\mathbf{b}_i^b)/\tau}}{e^{\cos(\mathbf{b}_i^a,\mathbf{b}_i^b)/\tau} + \sum\limits_{v,j\neq i} e^{\cos(\mathbf{b}_i^a,\mathbf{b}_j^v)/\tau}} \tag{2}$$

in which $\mathbf{b}_i^a = f(\mathbf{x}_i^a)$ denotes the relaxed binary codes generated from a hash encoder $f(\cdot)$ by giving a transformed view a of image $\mathbf{x}_i$ as input. $v \in \{a,b\}$ denotes which view is selected and τ represents the temperature parameters.

In the above setting, each image and its augmentation are treated as similar pairs, while all other combinations are treated as negative pairs, which will bury the similar pairs with cross samples. Therefore, in our DSCH, we expand the above objective to cover more potentially similar signals.

Expectation-Maximum

The Expectation-Maximum (EM) algorithm (Dempster, Laird, and Rubin 1977) has been proposed to estimate the model parameters Θ with maximum likelihood based on observed data $\mathbf{X}$ and unobserved latent variables $\mathbf{Z} = \{\mathbf{z}_j\}_{j=1}^m$:

$$\sum_i^n \log p(\mathbf{x}_i|\Theta) = \sum_i^n \log \sum_j^m p(\mathbf{x}_i, \mathbf{z}_j|\Theta) \tag{3}$$

To tackle this objective, EM contains two iterative steps:

E step. Expecting the posterior distribution of latent variables $\mathbf{Z}$ based on $\mathbf{X}$ and the last model parameters Θ^t, which is derived as:

$$Q_i(\mathbf{z}_j) = p(\mathbf{z}_j|\mathbf{x}_i, \Theta^t) \tag{4}$$

M step. Based on the posterior distribution $Q_i(\mathbf{z}_j)$ of E step, we define the log likelihood function of Θ as

$$\mathfrak{L}(\Theta) = \sum_i^n \sum_j^m Q_i(\mathbf{z}_j) \log p(\mathbf{x}_i, \mathbf{z}_j|\Theta) \tag{5}$$

Then, updating the model parameters Θ by maximizing the expectation of Eq.5.

$$\Theta^{t+1} = \arg\max_{\Theta} \mathfrak{L}(\Theta) \tag{6}$$

The EM algorithm iterates the E step and the M step until convergence. By introducing the latent variables $\mathbf{Z}$, the EM algorithm has been proved to optimize the initial objective $\sum_i^n \log p(\mathbf{x}_i|\Theta)$ (Neal and Hinton 1998).

Methodology

In this section, we illustrate DSCH from three folds. Firstly, how we define the hash encoder. Secondly, how to model the semantic components. Lastly, how we learn binary codes from built semantic components. An overall pipeline is shown in Fig.2, and we will discuss each regard in detail.

Hash Encoder

We employ the VGG-19 (Simonyan and Zisserman 2014) as hash encoder and denote it as $f(\cdot|\Theta)$ with network parameters Θ. To make this architecture meets the requirements of hash learning, we replace its last layers with a fc layer with 1000 hidden units and followed by another fc layer, where the node number is equal to hash codes length r. In training process, we adopt the $\tanh(\cdot)$ as the final activation function to tackle the ill-posed gradient of $\mathrm{sgn}(\cdot)$, and get the continuous approximation of binary code $\mathbf{b}_i$ as:

$$\mathbf{h}_i = \tanh\left(f(\mathbf{x}_i|\Theta)\right) \in [-1, +1]^r \tag{7}$$

Once finished training, we adopt $\mathrm{sgn}(\cdot)$ to yield the discrete binary code for Out-of-Sample extension:

$$\mathbf{b}_i = \mathrm{sgn}\left(f(\mathbf{x}_i|\Theta)\right) \in \{-1, +1\}^r \tag{8}$$

Semantic Component Structure

In this subsection, we illustrate that how we build a semantic component structure with co-occurence and homology relationships.

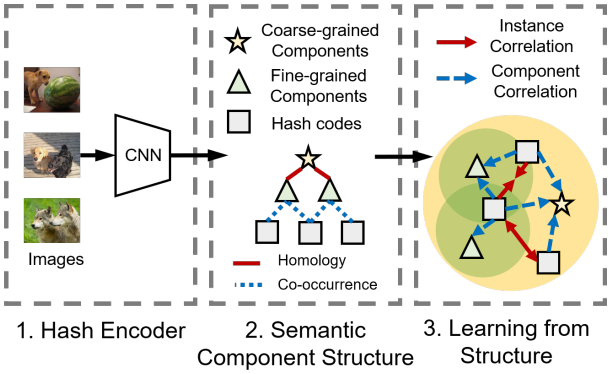


Figure 2: The pipeline of DSCH, which including three parts: (1) A hash encoder extracts the code representations for each image. (2) Constructing a semantic component structure based on obtained code representation. (3) Optimizing the encoder by modeling two kinds of correlations.

Semantic Co-occurrence Each image in the real world is usually composed of different types of objects, and each type of object is associated with a specific category. But in the unsupervised domain, since the label information is not available, it is a challenge to identify such categories meaning. Considering that any combination of objects may appear in an image, we assume that an image is generated from a mixture of a finite number of semantic components, where the distribution of each semantic component represents a kind of categories information. Based on the discussion above, it is natural to adopt GMM to model the relations between data points and semantic components. We set the component number as a large number m_1 to cover any possible semantics categories in fine-grained, and we denotes these fine-grained components as $\{C_j^{(1)}\}_{j=1}^{m_1}$. Next, we fit the parameters of components based on code representation $\mathbf{H} = \{\mathbf{h}_i\}_{i=1}^n$, therefore the distribution of a sample $\mathbf{h}_i$ can be modeled by these fine-grained components:

$$p(\mathbf{h}_i) = \sum_{j=1}^{m_1} p(C_j^{(1)}) p(\mathbf{h}_i|C_j^{(1)}) = \sum_{j=1}^{m_1} \pi_j^{(1)} \mathcal{N}(\mathbf{h}_i|\mu_j^{(1)}, \Sigma_j^{(1)}) \tag{9}$$

where $\pi_j^{(1)}$ is equal to $p(C_j^{(1)})$ and satisfies $\sum_{j=1}^{m_1} \pi_j^{(1)} = 1$ and $0 \leq \pi_j^{(1)} \leq 1$, which denotes the prior probability of component $C_j^{(1)}$. $p(\mathbf{h}_i|C_j^{(1)})$ is measured by a multivariate gaussian distribution $\mathcal{N}(\mathbf{h}_i|\mu_j^{(1)}, \Sigma_j^{(1)})$ with j-th component parameter mean vector $\mu_j^{(1)} \in \mathbb{R}^r$ and covariance $\Sigma_j^{(1)} \in \mathbb{R}^{r \times r}$. These components parameters $\{\pi_j^{(1)}, \mu_j^{(1)}, \Sigma_j^{(1)}\}_{j=1}^{m_1}$ could be calculated iteratively via GMM algorithm:

$$\{\pi_j^{(1)}, \mu_j^{(1)}, \Sigma_j^{(1)}\}_{j=1}^{m_1} \leftarrow \mathrm{GMM}\left(\mathbf{H}, m_1\right) \tag{10}$$

When GMM converges, we use mean vector $\mu_j^{(1)}$ to denote the representation of j-th fine-grained semantic component $C_j^{(1)}$ and define $\mathbf{p}_i = [p_{1i}^{(1)}, p_{2i}^{(1)}, \cdots, p_{m_1 i}^{(1)}]^T$ to rep-

resent the assignments of sample $\mathbf{h}_i$ belong to each fine-grained component, which elements $p_{ji}^{(1)}$ is estimated as:

$$p_{ji}^{(1)} = p(C_j^{(1)}|\mathbf{h}_i) = \frac{\pi_j^{(1)}\mathcal{N}(\mathbf{h}_i|\mu_j^{(1)}, \Sigma_j^{(1)})}{\sum_{k=1}^{m_1}\pi_k^{(1)}\mathcal{N}(\mathbf{h}_i|\mu_k^{(1)}, \Sigma_k^{(1)})} \quad (11)$$

Semantic Homology Next, we explore another correlation among semantic components, name semantic homology, which means that components with similar meanings should be from the same source. For example, `chihuahua` and `husky` are both breeds of `dogs`. Intuitively, these relationships can be organized in a hierarchical way by clustering.

We are inspired and perform k-means with a less cluster number $m_2 < m_1$ on mean vectors $\{\mu_j^{(1)}\}_{j=1}^{m_1}$ of fine-grained components $\{C_j^{(1)}\}_{j=1}^{m_1}$ to form coarse-grained components $\{C_j^{(2)}\}_{k=1}^{m_2}$, where the assignment of fine-grained $C_j^{(1)}$ belongs to coarse-grained $C_k^{(2)}$ is defined as:

$$p(C_k^{(2)}|C_j^{(1)}) = \begin{cases} 1, & \text{if } C_j^{(1)} \in C_k^{(2)} \\ 0, & \text{otherwise} \end{cases} \quad (12)$$

and we represent the representation of $C_k^{(2)}$ as:

$$\mu_k^{(2)} = \frac{\sum_j^{m_1} p(C_k^{(2)}|C_j^{(1)})\mu_j^{(1)}}{\sum_j^{m_1} p(C_k^{(2)}|C_j^{(1)})} \quad (13)$$

Also, we define the prior probability of coarse-grained component $p(C_k^2)$ as $\pi_k^{(2)} = \frac{1}{m_2}$, since the centroid obtained with k-means should be treated fairly.

Lastly, based on the assignment $\mathbf{h}_i \to C_j^{(1)}$ of Eq.11 and $C_j^{(1)} \to C_k^{(2)}$ of Eq.12, we establish the assignment $p_k^{(2)}(i)$ of sample $\mathbf{h}_i$ to the k-th coarse-grained semantics $C_k^{(2)}$ by a chain rule $\mathbf{h}_i \to C_j^{(1)} \to C_k^{(2)}$, which formulated as:

$$p_{ki}^{(2)} = p(C_k^{(2)}|\mathbf{h}_i) = \sum_{j=1}^{m_1} p(C_k^{(2)}|C_j^{(1)})p_j^{(1)}(i) \quad (14)$$

Complexity analysis. The complexity of constructing a semantic structure is $O(nm_1r^3 + m_1m_2r + nm_1m_2)$ with n training samples, code length r, m_1 fine-grained components and m_2 coarse-grained components. In the above, the first term is GMM algorithm, the second term denotes k-means clustering, and the third term is brought by the mapping of Eq.14. Notably, this complexity is linear to data samples number $O(n)$.

Learning from Structure

In this section, we illustrate how we learn a discriminative hash model from the built semantic component structure. We decouple this structure as multiple connections with two kinds: instance correlation and semantic correlation.

Instance Correlation The pairwise similarity between images is an essential cue to direct the hash model. Generally, if two images are similar semantically, then their semantic composition should also be similar. For example, two pictures both contain `sky`, `dog` and `airplane` are very likely to describe a similar scene. Thus, the similarity of images can be expressed by how similar their semantic components are. Based on this tuition, we develop a metric s_{ij} based on sample assignments to fine-grained components $\mathbf{p}_i$ of Eq.11 as:

$$s_{ij} = \cos(\mathbf{p}_i, \mathbf{p}_j) = \frac{\mathbf{p}_i^T\mathbf{p}_j}{\|\mathbf{p}_i\|_2\|\mathbf{p}_j\|_2} \quad (15)$$

where $s_{ij} \in [0, 1]$, denotes how similar $\mathbf{h}_i$ and $\mathbf{h}_j$ are in their distribution of assigned fine-grained semantic components. Especially, when s_{ij} equals to 1, we have $\mathbf{p}_i = \mathbf{p}_j$, which means $\mathbf{h}_i$ and $\mathbf{h}_j$ are almost the same semantically. When s_{ij} is equal to 0, it means $\mathbf{h}_i$ and $\mathbf{h}_j$ are totally dissimilar.

In the loss of Eq.2, it normally treats two transformed views of an image as a positive pair, while ignoring the similarity information with cross samples. Thus, a straightforward idea is expanding the scope of similar pairs. We adopt the s_{ij} to identify which pairs are similar and extent the contrastive loss of Eq.1 as follows:

$$\mathcal{L}_1 = \sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_{ij}\left(\widetilde{l}_{ij}^{aa} + \widetilde{l}_{ij}^{ab} + \widetilde{l}_{ij}^{ba} + \widetilde{l}_{ij}^{bb}\right) \quad (16)$$

where α_{ij} equals to $\frac{s_{ij}}{4\sum_i\sum_j s_{ij}}$ with normalization and $\widetilde{l}_{ij}^{ab}$ as:

$$\widetilde{l}_{ij}^{ab} = -\log\frac{e^{\cos(\mathbf{h}_i^a, \mathbf{h}_j^b)/\tau}}{\sum_{v_1}\sum_{v_2}\sum_{i'}\sum_{j'}\left(e^{\cos(\mathbf{h}_{i'}^{v_1}, \mathbf{h}_{j'}^{v_2})/\tau}\right)} \quad (17)$$

in which $v_1 \in \{a, b\}$ and $v_2 \in \{a, b\}$. In the objective of Eq.16, these data pair with a higher similar semantic components s_{ij} will be encouraged to be closer more with their different transformed views. Notably, our similar information is not only cross-views but also cross-samples, and the Eq.1 can be formulated as a special case when only $s_{ii} = 1$.

Component Correlation The goal of hash encoder $f(\cdot|\Theta)$ can be formulated to find the parameters Θ with the maximum likelihood of $\sum_i^n \log p(\mathbf{x}_i|\Theta)$. By regarding these semantic components $\{C_j^{(1)}\}_{j=1}^{m_1}$ and $\{C_j^{(2)}\}_{j=1}^{m_2}$ as latent variables. We can rewrite the initial goal as:

$$\sum_{i=1}^{n}\log p(\mathbf{x}_i|\Theta) = \sum_{i=1}^{n}\log\sum_{l=1}^{2}\sum_{j=1}^{m_l}p(\mathbf{x}_i, C_j^{(l)}|\Theta) \quad (18)$$

and this objective can be solved iteratively via the EM algorithm, which includes the following two steps:

E step. We estimate the posterior distribution $Q_i(C_j^{(l)})$ of latent semantic components based on $\mathbf{X}$ and parameters Θ^t, which with solutions Eq.11 and Eq.14.

$$\begin{aligned} Q_i(C_j^{(l)}) &= p(C_j^{(l)}|\mathbf{x}_i, \Theta^t) \\ &\Rightarrow p(C_j^{(l)}|\mathbf{h}_i) = \begin{cases} p_{ji}^{(1)}, & l = 1 \\ p_{ji}^{(2)}, & l = 2 \end{cases} \end{aligned} \quad (19)$$

M step. Based on $Q_i(C_j^{(l)})$, we estimate the log-likelihood of Θ as:

$$\mathcal{L}_2(\Theta) = \sum_{i=1}^{n}\sum_{l=1}^{2}\sum_{j=1}^{m_l} Q_i(C_j^{(l)}) \log p(\mathbf{x}_i, C_j^{(l)}|\Theta) \quad (20)$$

in which $p(\mathbf{x}_i, C_j^{(l)}|\Theta)$ equivalent to:

$$\begin{aligned} p(\mathbf{x}_i, C_j^{(l)}|\Theta) &= p(\mathbf{x}_i|C_j^{(l)}, \Theta)p(C_j^{(l)}|\Theta) \\ &\Rightarrow p(\mathbf{h}_i|C_j^{(l)})p(C_j^{(l)}) \end{aligned} \quad (21)$$

where $p(C_j^{(l)})$ equal to $\pi_j^{(1)}$ from Eq.10 when $l = 1$, and equal to $\pi_j^{(2)}$ when $l = 2$. The $p(\mathbf{h}_i|C_j^{(l)})$ could be expressed as the posterior probability of sample $\mathbf{h}_i$ affiliates with $C_j^{(l)}$, which normally relates to the cosine similarity between $\mathbf{h}_i$ and components representation $\mu_j^{(l)}$. Also, consider that probability $p(\mathbf{h}_i|C_j^{(l)})$ is non-negative and should be normalized, we formulate it as:

$$p(\mathbf{h}_i|C_j^{(l)}) = \frac{e^{\cos(\mathbf{h}_i, \mu_j^{(l)})/\tau}}{\sum_{g=1}^{m_l} e^{\cos(\mathbf{h}_i, \mu_g^{(l)})/\tau}} \quad (22)$$

Combining the Eq. 19, Eq. 20 and Eq. 22 together, the solution of model parameters is defined as:

$$\begin{aligned} \Theta^{t+1} &= \arg\max_{\Theta} \mathcal{L}_2(\Theta) = \arg\min_{\Theta} -\mathcal{L}_2(\Theta) \\ &= \min - \sum_{l=1}^{2}\sum_{i=1}^{n}\sum_{j=1}^{m_l} \beta_{ij}^{l} \log \frac{e^{\cos(\mathbf{h}_i, \mu_j^{(l)})/\tau}}{\sum_{g=1}^{m_l} e^{\cos(\mathbf{h}_i, \mu_g^{(l)})/\tau}} \end{aligned} \quad (23)$$

where β_{ij}^{l} equal to $\pi_j^{(l)} p_{ji}^{(l)}$.

The principle behind Eq.23 is that we direct every data sample $\mathbf{h}_i$ close to their associated components center $\mu_j^{(l)}$ with its assignment weights β_{ij}^{k} , with the purpose for align the instance representation with a weighted combination of components in different semantic granularities.

Objective Function and Optimization

We introduce data augmentation into $\mathcal{L}_2$ and integrate $\mathcal{L}_1$ into the EM framework M steps. Besides, we discretize the component center $\mu_j^{(l)}$ in Eq.23 as $\mathrm{sgn}(\mu_j^{(l)})$ to minimize the quantization error. Therefore, the total loss is formed as:

$$\begin{aligned} \min_{\Theta} \mathcal{L} &= \mathcal{L}_1 + \lambda\widetilde{\mathcal{L}_2} \\ &= -\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_{ij} \left(\widetilde{l}_{ij}^{aa} + \widetilde{l}_{ij}^{ab} + \widetilde{l}_{ij}^{ba} + \widetilde{l}_{ij}^{bb}\right) \\ &\quad - \lambda \sum_{v}^{a,b}\sum_{l=1}^{2}\sum_{i=1}^{n}\sum_{j=1}^{m_l} \beta_{ij}^{l} \log \frac{e^{\cos(\mathbf{h}_i^v, \mathrm{sgn}(\mu_j^{(l)}))/\tau}}{\sum_{g=1}^{m_l} e^{\cos(\mathbf{h}_i^v, \mathrm{sgn}(\mu_g^{(l)}))/\tau}} \end{aligned} \quad (24)$$

where λ is a weight coefficient of loss $\mathcal{L}_2$ and $\widetilde{l}_{ij}^{ab}$ is defined in Eq.17.

The algorithm of DSCH is described in Algorithm.1, which is based on the EM algorithm, and its optimization contains two alternate steps:

Algorithm 1: **Deep Semantic Component Hashing (DSCH)**

Require: Hash model $f(\cdot|\Theta)$, image set $\mathbf{X}$, hash code length r, temperature factor τ, semantic components number m_1 and m_2, weight coefficient λ, epoch E, learning rate η.

```
Initialize parameters Θ^0.
for t = 1 to E do
    ▷ E STEP.
    Sampling {C_j^(1)}_{j=1}^{m_1} via GMM.
    Sampling {C_j^(2)}_{j=1}^{m_2} via k-means.
    ▷ M STEP.
    Update model parameters Θ^{t+1} via Eq.25.
end for
Obtain the B via Eq.8.
```

Ensure: Hash model $f(\cdot|\Theta)$, hash codes $\mathbf{B}$

- **E step.** Sampling the semantic components $\{C_j^{(1)}\}_{j=1}^{m_1}$ with GMM and $\{C_j^{(2)}\}_{j=1}^{m_2}$ with k-means.
- **M step.** Optimizing the network parameter Θ via back propagation (BP) with a mini-batch sampling.

$$\Theta \leftarrow \Theta - \eta\nabla_{\Theta}(\mathcal{L}) \quad (25)$$

where η is the learning rate and ∇_{Θ} represents a derivative of Θ.

Experiments

In this section, we conduct experiments on various public benchmark datasets to evaluate our DSCH method.

Datasets

We evaluate our methods on three public benchmark datasets, i.e. CIFAR-10, FLICKR25K, and NUS-WIDE.

CIFAR-10 dataset contains 60,000 images in 10 categories, where each class contains 6,000 images with size 32×32. We randomly selected 100 images for each class as the query set, 1,000 in total. Then we used the remaining images as the retrieval set, among them, we randomly selected 1,000 images per class as the training set.

FLICKR25K is a dataset with multi-label, which contains 25,000 images and each image is labeled with at least one of 24 classes labels. We randomly selected 1,000 images per class as the query set and the remaining images are left for retrieval sets. In the retrieval sets, we randomly choose 10,000 images as the training set.

NUS-WIDE is a multi-label dataset that includes 269,648 images in 81 classes, and each image is also tagged with multiple labels more than classes. We selected 21 most frequent classes from the dataset and each class contains at least 5,000 related images. Among them, 2100 images are selected as a query set randomly while the remaining images were treated as a retrieval set, where 10,500 images were for training. For multi-label datasets, if the retrieved image shares at least one label, it is regarded as being associated with the query image.

Method	Reference	CIFAR-10			FLICKR25K			NUS-WIDE		
		16 bits	32 bits	64 bits	16 bits	32 bits	64 bits	16 bits	32 bits	64 bits
LSH+VGG	STOC-02	0.177	0.192	0.261	0.596	0.619	0.650	0.385	0.455	0.446
SH+VGG	NeurIPS-09	0.254	0.248	0.229	0.661	0.608	0.606	0.508	0.449	0.441
ITQ+VGG	PAMI-13	0.269	0.295	0.316	0.709	0.696	0.684	0.519	0.576	0.598
AGH+VGG	ICML-11	0.397	0.428	0.441	0.744	0.735	0.771	0.563	0.698	0.725
SP+VGG	CVPR-15	0.280	0.343	0.365	0.726	0.705	0.713	0.581	0.603	0.673
SGH+VGG	ICML-17	0.286	0.320	0.347	0.608	0.657	0.693	0.463	0.588	0.638
GH	NeurIPS-18	0.355	0.424	0.419	0.702	0.732	0.753	0.599	0.657	0.695
SSDH	IJCAI-18	0.241	0.239	0.256	0.710	0.696	0.737	0.542	0.629	0.635
BGAN	AAAI-18	0.535	0.575	0.587	0.766	0.770	0.795	0.719	0.745	0.761
MLS^{3}RDUH	IJCAI-20	0.562	0.588	0.595	0.797	0.809	0.809	0.730	0.754	0.764
TBH	CVPR-20	0.432	0.459	0.455	0.779	0.794	0.797	0.678	0.717	0.729
CIB	IJCAI-21	0.547	0.583	0.602	0.773	0.781	0.798	0.756	0.777	0.781
DSCH	**Proposed**	**0.624**	**0.644**	**0.670**	**0.817**	**0.827**	**0.828**	**0.762**	**0.780**	**0.786**

Table 1: MAP@5000 results on CIFAR10, FLICKR25K and NUS-WIDE. The best result is shown in boldface.

Experiment Settings

Metrics. We evaluate model performance by three widely used metrics: Mean Average Precision (MAP) to measure the hamming ranking quality, Precision/Precision-recall curves to display model overall performance.

Baseline Methods. We compared DSCH with twelve unsupervised hashing methods, including six shallow hashing models: **LSH** (Andoni and Indyk 2006), **SH** (Weiss, Torralba, and Fergus 2009), **ITQ** (Gong et al. 2012), **AGH** (Liu et al. 2011), **SP** (Xia et al. 2015), **SGH** (Dai et al. 2017) and six deep hashing models: **GH** (Su et al. 2018), **SSDH** (Yang et al. 2018), **BGAN** (Song et al. 2018), **MLS^{3}RDUH** (Tu, Mao, and Wei 2020) and **TBH** (Shen et al. 2020) and **CIB** (Qiu et al. 2021). The parameters of the above methods referred to the setting provided by papers, and all shallow hashing methods use the fc7 layer 4096-dimensional feature of VGG19 pre-trained on ImageNet.

Implementation Details. We conducted experiments on a workstation equipped with Intel Xeon Platinum 8255C CPU and Nvidia V100 GPU. During the training, each input image was resized to 224×224. For data augmentation, we use the same strategy as CIB. The epoch number was set to 100 with batch size 128, and the factor τ we set as 1. We adopted the Adam optimization with learning rate η to 5e-4. The parameters m_1 and m_2 were set to $\{1000, 1000, 2000\}$ and $\{900, 100, 1500\}$ for CIFAR, FLICKR and NUS-WIDE respectively, and λ was fixed to 0.1 by default.

Comparison Results and Discussions

The MAP@5000 results on three benchmarks are shown in Tab. 1, with code length varying from 16 to 64. It is clear that DSCH consistently achieves the best results among three datasets, with an averaged increase of 11.9%, 5.0%, 0.6% on CIFAR-10, FLICKR25K, NUS-WIDE compared with the CIB. To sufficiently reveal the overall performance of DSCH, we report the PR curve and Precision@5000 curve of 64 bits in Fig.3. It can be found that the PR curve of DSCH covers more areas in three benchmarks and normally has a higher precision with the same returned images number. This means DSCH yields a stable superior performance.

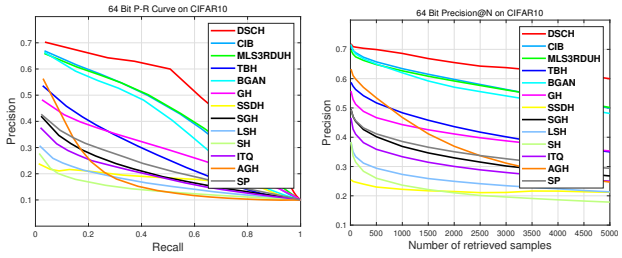
Ablation Study

In this section, we conduct an ablation analysis to understand the effect of DSCH components, which consists of two major losses: (a) instance correlation with $\mathcal{L}_1$ by expanding the $\mathcal{L}_0$ to cover cross-sample similar pairs. (b) components correlation with $\widetilde{\mathcal{L}_2}$ to keep semantic alignment at both fine and coarse levels. Therefore, we first design a variant with loss $\mathcal{L}_0$ as a baseline (`Base`), and then replace it by $\mathcal{L}_1$ to evaluate the instance correlation (`IC`). Further, equip this variant with the fine-grained (`CC-F`) and coarse-grained components correlation (`CC-C`) progressively. We list the results on the FLICKR25K in Table.2, which are evaluated with MAP@5000 and the code length varies from 16 to 64.

Components				MAP@5000		
`Base`	`IC`	`CC-F`	`CC-C`	16 bits	32 bits	64 bits
✓				0.735	0.744	0.769
✓	✓			0.768	0.781	0.797
✓	✓	✓		0.809	0.819	0.822
✓	✓	✓	✓	**0.817**	**0.827**	**0.828**

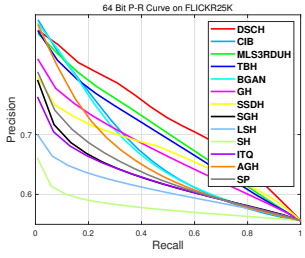
Table 2: Ablation study on FLICKR25K by validating the DSCH different components.

As shown in the second row, introducing instance correlation is able to obtain 4.4% (16 bits), 5.0% (32 bits), 3.6% (64 bits) improvements, which means that semantic components provide good clues to seek similar pairs. Further, the performance can be gradually improved by involving the fine-grained and coarse-grained semantic correlation, resulting in 11.1% (16 bits), 11.1% (32 bits), 7.6% (64 bits) gains over baseline, which indicates that the hierarchical semantic components enhance the discrimination power of model.

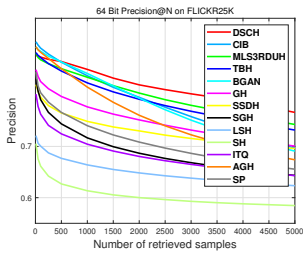


(a) CIFAR-10: 64 bits

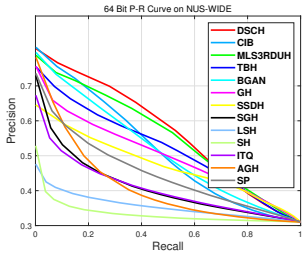
(b) CIFAR-10: 64 bits



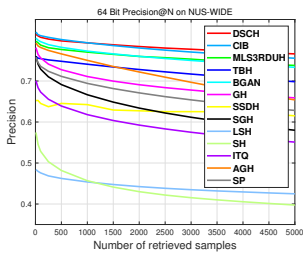
(c) FLICK25K: 64 bits



(d) FLICK25K: 64 bits



(e) NUS-WIDE: 64 bits

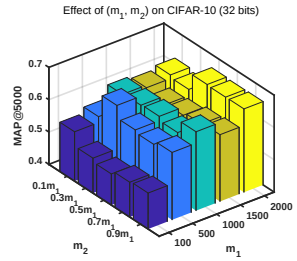


(f) NUS-WIDE: 64 bits

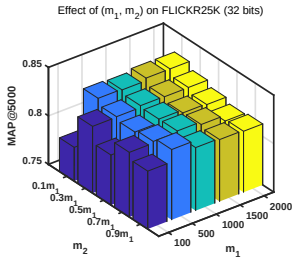
Figure 3: Precision-recall (PR) curves and Precision@5000 curves on the CIFAR-10, FLICK25K and NUS-WIDE datasets with 64 bits length.

Parameter Sensitivity

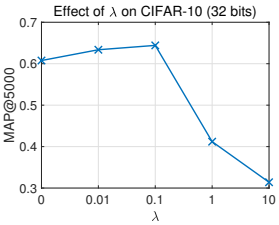
In Fig.4 (a)(b), we evaluate the influence of different settings of components number (m_1, m_2) on CIFAR-10 and FLICKR25K with 32 bits, where $m_1 \in [100, 500, 1000, 1500, 2000]$ and $m_2 = \gamma m_1, \gamma \in [0.1, 0.3, 0.5, 0.7, 0.9]$. The pattern shows that retrieval performance is jointly affected by m_1 and m_2 under different semantic granularity. Generally, MAP@5000 increases with m_1, this is due to the larger fine-grained components number bringing a finer division of semantic, which helps the model achieve better discrimination capabilities. Besides, a proper m_2 brings a certain performance improvement. We noticed that in CIFAR-10, a larger m_2 is preferred but in FLICKR25K, a very large m_2 will bring relative performance degradation. The recommended value is $0.9m_1$ for CIFAR-10 and $0.1m_1$ for FLICKR25K. In Fig.4(c)(d), we study the effect of λ, which denotes the weights of component correlation. We find that performance gain is brought when λ smaller than 0.1 and $\lambda = 0.1$ are a good choice in two datasets.



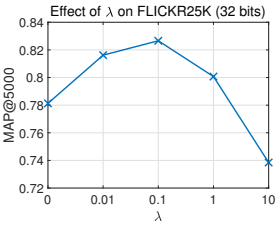
(a) (m_1, m_2) in CIFAR-10



(b) (m_1, m_2) in FLICKR25K



(c) λ in CIFAR-10

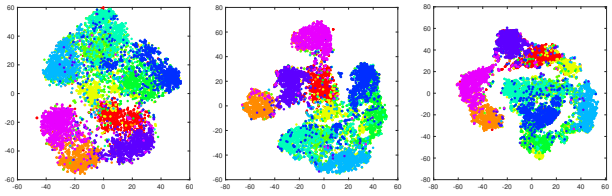


(d) λ in FLICKR25K

Figure 4: Parameter analysis on CIFAR-10 and FLICKR25K

Visualization Analysis

In Fig.5, we display the t-SNE visualization of hash codes on CIFAR-10 with 32 bits from different variants enumerated in Tab.2. The color of points indicates the category samples belong to. The figure shows that introducing instance correlation obviously refine the manifold structure, where the images within the same class shared smaller distances to each other. By further equipping components correlation, data points are encouraged to be closer to their associated components, so the cluster structure is more compact.



(a) `Base` (b) `Base+IC` (c) `Base+IC+CC`

Figure 5: t-SNE visualization of hash codes on CIFAR-10

Conclusion

We proposed a novel DSCH to yield binary hash codes by regarding every image is composed of latent semantic components and optimizing the model based on an EM framework, which includes the iterative two steps: E-step, DSCH constructs a semantic structure to recognize the semantic component of images and explores their homology and co-occurrence relationships. M-step, DSCH maximizes the similarities among samples with shared semantic components and pulls data points to their associated components centers at different granularities.. Extensive experiments on three datasets demonstrate the effectiveness of DSCH.

Acknowledgments

This work is jointly supported by the 2021 Tencent Rhino-Bird Research Elite Training Program; in part by Major Project of the New Generation of Artificial Intelligence (No. 2018AAA0102900); in part by NSFC under Grant no. 61773268; and in part by the Shenzhen Research Foundation for Basic Research, China, under Grant JCYJ20210324093000002.

References

Andoni, A.; and Indyk, P. 2006. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. In *2006 47th annual IEEE symposium on foundations of computer science (FOCS'06)*, 459–468. IEEE.

Chen, T.; Kornblith, S.; Norouzi, M.; and Hinton, G. 2020. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*.

Dai, B.; Guo, R.; Kumar, S.; He, N.; and Song, L. 2017. Stochastic generative hashing. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, 913–922.

Dempster, A. P.; Laird, N. M.; and Rubin, D. B. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1): 1–22.

Deng, C.; Yang, E.; Liu, T.; Li, J.; Liu, W.; and Tao, D. 2019. Unsupervised semantic-preserving adversarial hashing for image search. *IEEE Transactions on Image Processing*, 28(8): 4032–4044.

Gong, Y.; Lazebnik, S.; Gordo, A.; and Perronnin, F. 2012. Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval. *IEEE transactions on pattern analysis and machine intelligence*, 35(12): 2916–2929.

He, K.; Fan, H.; Wu, Y.; Xie, S.; and Girshick, R. 2020. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 9729–9738.

Lin, K.; Lu, J.; Chen, C.-S.; and Zhou, J. 2016. Learning compact binary descriptors with unsupervised deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1183–1192.

Lin, Q.; Chen, X.; Zhang, Q.; Tian, S.; and Chen, Y. 2021. Deep Self-Adaptive Hashing for Image Retrieval. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, 1028–1037.

Liu, W.; Mu, C.; Kumar, S.; and Chang, S.-F. 2014. Discrete graph hashing.

Liu, W.; Wang, J.; Ji, R.; Jiang, Y.-G.; and Chang, S.-F. 2012. Supervised hashing with kernels. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, 2074–2081. IEEE.

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

Liu, W.; and Zhang, T. 2016. Multimedia hashing and networking. *IEEE MultiMedia*, 23(3): 75–79.

Luo, X.; Wu, D.; Ma, Z.; Chen, C.; Zhong, H.; Deng, M.; Huang, J.; and Hua, X.-s. 2020. Cimon: Towards high-quality hash codes. *arXiv preprint arXiv:2010.07804*.

Neal, R. M.; and Hinton, G. E. 1998. A view of the EM algorithm that justifies incremental, sparse, and other variants. In *Learning in graphical models*, 355–368. Springer.

Oord, A. v. d.; Li, Y.; and Vinyals, O. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*.

Qiu, Z.; Su, Q.; Ou, Z.; Yu, J.; and Chen, C. 2021. Unsupervised Hashing with Contrastive Information Bottleneck. *arXiv preprint arXiv:2105.06138*.

Shen, F.; Shen, C.; Liu, W.; and Tao Shen, H. 2015. Supervised discrete hashing. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 37–45.

Shen, F.; Xu, Y.; Liu, L.; Yang, Y.; Huang, Z.; and Shen, H. T. 2018. Unsupervised deep hashing with similarity-adaptive and discrete optimization. *IEEE transactions on pattern analysis and machine intelligence*, 40(12): 3034–3044.

Shen, Y.; Qin, J.; Chen, J.; Yu, M.; Liu, L.; Zhu, F.; Shen, F.; and Shao, L. 2020. Auto-Encoding Twin-Bottleneck Hashing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2818–2827.

Simonyan, K.; and Zisserman, A. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Song, J.; He, T.; Gao, L.; Xu, X.; Hanjalic, A.; and Shen, H. T. 2018. Binary generative adversarial networks for image retrieval. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Su, S.; Zhang, C.; Han, K.; and Tian, Y. 2018. Greedy hash: Towards fast optimization for accurate hash coding in cnn. In *Advances in neural information processing systems*, 798–807.

Sun, C.; Song, X.; Feng, F.; Zhao, W. X.; Zhang, H.; and Nie, L. 2019. Supervised hierarchical cross-modal hashing. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 725–734.

Tu, R.-C.; Mao, X.-L.; Guo, J.-N.; Wei, W.; and Huang, H. 2021a. Partial-Softmax Loss based Deep Hashing. In *Proceedings of the Web Conference 2021*, 2869–2878.

Tu, R.-C.; Mao, X.-L.; Kong, C.; Shao, Z.; Li, Z.-L.; Wei, W.; and Huang, H. 2021b. Weighted Gaussian Loss based Hamming Hashing. In *Proceedings of the 29th ACM International Conference on Multimedia*, 3409–3417.

Tu, R.-C.; Mao, X.-L.; Ma, B.; Hu, Y.; Yan, T.; Wei, W.; and Huang, H. 2020. Deep cross-modal hashing with hashing functions and unified hash codes jointly learning. *IEEE Transactions on Knowledge and Data Engineering*.

Tu, R.-C.; Mao, X.-L.; and Wei, W. 2020. MLS3RDUH: Deep Unsupervised Hashing via Manifold based Local Semantic Similarity Structure Reconstructing. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, 3466–3472.

Wang, D.; Huang, H.; Lu, C.; Feng, B.-S.; Wen, G.; Nie, L.; and Mao, X.-L. 2018. Supervised deep hashing for hierarchical labeled data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Wang, J.; Liu, W.; Kumar, S.; and Chang, S.-F. 2015. Learning to hash for indexing big data—A survey. *Proceedings of the IEEE*, 104(1): 34–57.

Weiss, Y.; Torralba, A.; and Fergus, R. 2009. Spectral hashing. In *Advances in neural information processing systems*, 1753–1760.

Xia, Y.; He, K.; Kohli, P.; and Sun, J. 2015. Sparse Projections for High-Dimensional Binary Codes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Yang, E.; Deng, C.; Liu, T.; Liu, W.; and Tao, D. 2018. Semantic structure-based unsupervised deep hashing. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, 1064–1070.

Yuan, L.; Wang, T.; Zhang, X.; Tay, F. E.; Jie, Z.; Liu, W.; and Feng, J. 2020. Central similarity quantization for efficient image and video retrieval. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 3083–3092.