Deep Unsupervised Hashing with Latent Semantic Components

Qinghong Lin¹*, Xiaojun Chen¹†, Qin Zhang¹, Shaotian Cai¹, Wenzhe Zhao², Hongfa Wang²

¹Shenzhen University, Shenzhen, China
²Tencent 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

1 This work is done when Qinghong Lin is an intern at Tencent.
2 Corresponding authors

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.
ages and their co-occurrence relations, we notice that the homology relationships between them are another useful in-
formation, 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 algo-

rithm to yield discriminative binary codes. Firstly, DSCH constructs a semantic component structure with fully ex-
ploring 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 hier-
archy. Secondly, DSCH makes the images close to their se-
matic component centers on both fine-grained and coarse-

rarchy. Secondly, DSCH makes the images close to their se-
mal component centers on both fine-grained and coarse-
erarchy. Secondly, DSCH makes the images close to their se-
mal component centers on both fine-grained and coarse-
to coarse-grained semantic components. To speed up the con-
struction of the graph, Anchor Graph Hashing (AGH) (Liu
et al. 2011) proposes a sparse low-rank graph by introduc-
ing a set of anchors. Although the aforementioned methods
have made progress in this area, they are all shallow archi-
tectures that rely heavily on hand-crafted features. To tackle
this problem, amount of deep unsupervised hashing meth-
ods (Deng et al. 2019) (Tu et al. 2020) (Tu et al. 2021a) (Tu
et al. 2021b) have been proposed, in which Deep binary des-
criptors (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 struc-
ture. Twin-Bottleneck Hashing (TBH) (Shen et al. 2020) in-
troduces 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 Vin-
yals 2018) to meet the requirement of hashing learning as:

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

where

\[
l_i^a = - \log \frac{e^{\cos(b_i^a, b_i^b) / \tau}}{e^{\cos(b_i^a, b_i^b) / \tau} + \sum_{v \neq i} e^{\cos(b_v^a, b_v^b) / \tau}} \tag{2}
\]

in which \(b_i^a = f(x_i^a)\) denotes the relaxed binary codes gen-
erated from a hash encoder \(f(\cdot)\) by giving a transformed view \(a\) of image \(x_i\) as input. \(v \in \{a, b\}\) denotes which view

is selected and \(\tau\) 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.

Related Work

Unsupervised Hashing

There are many well-known traditional hashing methods
been established in decades. Among them, Iterative Quan-
tization (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 de-
termine the pairwise code distances. To speed up the con-
struction of the graph, Anchor Graph Hashing (AGH) (Liu
et al. 2011) proposes a sparse low-rank graph by introduc-
ing a set of anchors. Although the aforementioned methods
have made progress in this area, they are all shallow archi-
tectures that rely heavily on hand-crafted features. To tackle
this problem, amount of deep unsupervised hashing meth-
ods (Deng et al. 2019) (Tu et al. 2020) (Tu et al. 2021a) (Tu
et al. 2021b) have been proposed, in which Deep binary des-
criptors (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 struc-
ture. Twin-Bottleneck Hashing (TBH) (Shen et al. 2020) in-
troduces 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 Vin-
yals 2018) to meet the requirement of hashing learning as:

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

where

\[
l_i^a = - \log \frac{e^{\cos(b_i^a, b_i^b) / \tau}}{e^{\cos(b_i^a, b_i^b) / \tau} + \sum_{v \neq i} e^{\cos(b_v^a, b_v^b) / \tau}} \tag{2}
\]

in which \(b_i^a = f(x_i^a)\) denotes the relaxed binary codes gen-
erated from a hash encoder \(f(\cdot)\) by giving a transformed view \(a\) of image \(x_i\) as input. \(v \in \{a, b\}\) denotes which view

is selected and \(\tau\) 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 $\Theta$ with maximum likelihood based on observed data $X$ and unobserved latent variables $Z = \{z_j\}_{j=1}^m$:

$$\sum_i^n \log p(x_i|\Theta) = \sum_i^n \sum_j^m p(x_i, z_j|\Theta)$$

To tackle this objective, EM contains two iterative steps: E step. Expecting the posterior distribution of latent variables $Z$ based on $X$ and the last model parameters $\Theta^t$, which is derived as:

$$Q_i(z_j) = p(z_j|x_i, \Theta^t)$$

**M step.** Based on the posterior distribution $Q_i(z_j)$ of E step, we define the log likelihood function of $\Theta$ as:

$$\mathcal{L}(\Theta) = \sum_i^n \sum_j^m Q_i(z_j) \log p(x_i, z_j|\Theta)$$

Then, updating the model parameters $\Theta$ by maximizing the expectation of Eq.5.

$$\Theta^{t+1} = \arg\max_{\Theta} \mathcal{L}(\Theta)$$

The EM algorithm iterates the E step and the M step until convergence. By introducing the latent variables $Z$, the EM algorithm has been proved to optimize the initial objective $\sum_i^n \log p(x_i|\Theta)$ (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 $\Theta$. To make this architecture meets the requirements of hash learning, we replace its last layers with a 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 sgn$(\cdot)$, and get the continuous approximation of binary code $b_i$ as:

$$b_i = \text{tanh} ( f(x_i|\Theta)) \in [-1, +1]^r$$

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

$$b_i = \text{sgn} ( f(x_i|\Theta)) \in \{-1, +1\}^r$$

**Semantic Component Structure**

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

![Diagram](image)

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 $H = \{b_i\}_{i=1}^n$, therefore the distribution of a sample $h_i$ can be modeled by these fine-grained components:

$$p(h_i) = \sum_{j=1}^{m_1} p(C_j^{(1)}) p(h_i|C_j^{(1)}) = \sum_{j=1}^{m_1} \pi_j^{(1)} N(h_i|\mu_j^{(1)}, \Sigma_j^{(1)})$$

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 $C_j^{(1)}$. $p(h_i|C_j^{(1)})$ is measured by a multivariate gaussian distribution $N(h_i|\mu_j^{(1)}, \Sigma_j^{(1)})$ 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:

$$\{\pi_j^{(1)}, \mu_j^{(1)}, \Sigma_j^{(1)}\}_{j=1}^{m_1} \leftarrow \text{GMM} (H, m_1)$$

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 $p_j = [p_{j1}, p_{j2}, \ldots, p_{mj}[^r$ to rep-
represent the assignments of sample $h_i$ belong to each fine-grained component, which elements $p^{(1)}_{ji}$ is estimated as:

$$p^{(1)}_{ji} = \frac{n^{(1)}_{ji} N(h_i) \mu_j^{(1)} \Sigma_j^{(1)}}{\sum_{k=1}^{m_1} n^{(1)}_{ki} N(h_i) \mu_k^{(1)} \Sigma_k^{(1)}} \quad (11)$$

**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 $\{C_j^{(2)}\}_{j=1}^{m_2}$, 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:

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

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 $h_i \rightarrow C_j^{(1)}$ of Eq.11 and $C_j^{(1)} \rightarrow C_k^{(2)}$ of Eq.12, we establish the assignment $p^{(2)}_{ki}$ of sample $h_i$ to the $k$-th coarse-grained semantics $C_k^{(2)}$ by a chain rule $h_i \rightarrow C_j^{(1)} \rightarrow C_k^{(2)}$, which formulated as:

$$p^{(2)}_{ki} = p(C_k^{(2)}|h_i) = \sum_{j=1}^{m_1} p(C_k^{(2)}|C_j^{(1)}) p^{(1)}_{ji} \quad (14)$$

**Complexity analysis.** The complexity of constructing a semantic structure is $O(n m_1 r^3 + m_1 m_2 + n m_1 m_2)$ 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 $p_i$ of Eq.11 as:

$$s_{ij} = \cos(p_i, p_j) = \frac{p_i^T p_j}{\|p_i\|_2 \|p_j\|_2} \quad (15)$$

where $s_{ij} \in [0, 1]$, denotes how similar $h_i$ and $h_j$ are in their distribution of assigned fine-grained semantic components. Especially, when $s_{ij}$ equals to 1, we have $p_i = p_j$, which means $h_i$ and $h_j$ are almost the same semantically. When $s_{ij}$ is equal to 0, it means $h_i$ and $h_j$ 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:

$$\mathcal{L}_1 = \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_{ij} (\tilde{t}_{ij} + \tilde{t}_{ab}^o + \tilde{t}_{ij} + \tilde{t}_{ab}^b) \quad (16)$$

where $\alpha_{ij}$ equals to $\sum_{s_{ij}}$ with normalization and $\tilde{t}_{ij}$ as:

$$\tilde{t}_{ij} = -\log \frac{e^{\cos(h_i^a, h_j^a)/\tau}}{\sum_{v_1 \in \{a,b\}} \sum_{v_2 \in \{a,b\}} \sum_{v_3 \in \{a,b\}} e^{\cos(h_i^v, h_j^v)/\tau}} \quad (17)$$

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 $s_{ij} = 1$.

**Component Correlation** The goal of hash encoder $f(\cdot|\Theta)$ can be formulated to find the parameters $\Theta$ with the maximum likelihood of $\sum_{i=1}^{n} \log p(x_i|\Theta)$. 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:

$$\sum_{i=1}^{n} \log p(x_i|\Theta) = \sum_{i=1}^{n} \log \sum_{j=1}^{m_1} \sum_{j=1}^{m_2} p(x_i, C_j^{(1)}|\Theta) \quad (18)$$

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

**E step.** We estimate the posterior distribution $Q_i(C_j^{(1)})$ of latent semantic components based on $X$ and parameters $\Theta'$, which with solutions Eq.11 and Eq.14.

$$Q_i(C_j^{(1)}) = p(C_j^{(1)}|x_i, \Theta')$$

$$\Rightarrow p(C_j^{(1)}|h_i) = \begin{cases} p^{(1)}_{ji}, & l = 1 \\ p^{(1)}_{jl}, & l = 2 \end{cases} \quad (19)$$

7491
M step. Based on $Q_j(C^{(l)}_j)$, we estimate the log-likelihood of $\Theta$ as:

$$L_2(\Theta) = \sum_{i=1}^{n} \sum_{l=1}^{2} \sum_{j=1}^{m_j} Q_j(C^{(l)}_j) \log p(x_i, C^{(l)}_j | \Theta)$$

(20)

in which $p(x_i, C^{(l)}_j | \Theta)$ equivalent to:

$$p(x_i, C^{(l)}_j | \Theta) = p(x_i | C^{(l)}_j, \Theta) p(C^{(l)}_j | \Theta)$$

$$\Rightarrow p(h_i | C^{(l)}_j) p(C^{(l)}_j)$$

(21)

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

$$p(h_i | C^{(l)}_j) = \frac{e^{\cos(h_i, \mu^{(l)}_j)/\tau}}{\sum_{g=1}^{m_j} e^{\cos(h_i, \mu^{(l)}_g)/\tau}}$$

(22)

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

$$\Theta^{t+1} = \arg \max_\Theta L_2(\Theta) = \arg \min_\Theta -L_2(\Theta)$$

$$= \min \left\{ - \sum_{l=1}^{2} \sum_{i=1}^{n} \sum_{j=1}^{m_j} m_j \beta^{(l)}_{ij} \log \frac{e^{\cos(h_i, \mu^{(l)}_j)/\tau}}{\sum_{g=1}^{m_j} e^{\cos(h_i, \mu^{(l)}_g)/\tau}} \right\}$$

(23)

where $\beta^{(l)}_{ij}$ equal to $\pi_j^{(l)} P_{ij}^{(l)}$.

The principle behind Eq.23 is that we direct every data sample $h_i$ close to their associated components center $\mu^{(l)}_j$ with its assignment weights $\beta^{(l)}_{ij}$, 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 $L_2$ and integrate $L_1$ into the EM framework M steps. Besides, we discretize the component center $\mu^{(l)}_j$ in Eq.23 as $\text{sgn}(\mu^{(l)}_j)$ to minimize the quantization error. Therefore, the total loss is formed as:

$$\min_\Theta L = L_1 + \lambda L_2$$

$$= - \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} \left( \tilde{r}_{ij}^{aa} + \tilde{r}_{ij}^{ab} + \tilde{r}_{ij}^{ba} + \tilde{r}_{ij}^{bb} \right)$$

$$- \lambda \sum_{v} \sum_{l=1}^{2} \sum_{i=1}^{n} \sum_{j=1}^{m_j} \beta^{(l)}_{ij} \log \frac{e^{\cos(h_i, \text{sgn}(\mu^{(l)}_j))/\tau}}{\sum_{g=1}^{m_j} e^{\cos(h_i, \text{sgn}(\mu^{(l)}_g))/\tau}}$$

(24)

where $\lambda$ is a weight coefficient of loss $L_2$ and $\tilde{r}_{ij}^{ab}$ 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 $X$, hash code length $r$, temperature factor $\tau$, semantic components number $m_1$ and $m_2$, weight coefficient $\lambda$, epoch $E$, learning rate $\eta$.

1. Initialize parameters $\Theta^0$.
2. for $t = 1$ to $E$ do
   3. $\triangleright$ E step.
   4. Sampling $\{C^{(1)}_j\}_{j=1}^{m_1}$ via GMM.
   5. Sampling $\{C^{(2)}_j\}_{j=1}^{m_2}$ via k-means.
   6. $\triangleright$ M step.
   7. Update model parameters $\Theta^{t+1}$ via Eq.25.

   end for

9. Obtain the $B$ via Eq.8.

**Ensure:** Hash model $f(\cdot | \Theta)$, hash codes $B$.

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

$$\Theta \leftarrow \Theta - \eta \nabla_\Theta (L)$$

(25)

where $\eta$ is the learning rate and $\nabla_\Theta$ represents a derivative of $\Theta$.

**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 $32 \times 32$. 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.
**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\textsuperscript{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 $224 \times 224$. For data augmentation, we use the same strategy as CIB. The epoch number was set to 100 with batch size 128, and the factor $\tau$ we set as 1. We adopted the Adam optimization with learning rate $\eta$ 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 $\lambda$ 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%, 2.1% 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) component correlation with $\mathcal{L}_2$ 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.

<table>
<thead>
<tr>
<th>Method</th>
<th>Reference</th>
<th>CIFAR-10</th>
<th>FLICKR25K</th>
<th>NUS-WIDE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>16 bits</td>
<td>32 bits</td>
<td>64 bits</td>
</tr>
<tr>
<td></td>
<td></td>
<td>16 bits</td>
<td>32 bits</td>
<td>64 bits</td>
</tr>
<tr>
<td></td>
<td></td>
<td>16 bits</td>
<td>32 bits</td>
<td>64 bits</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Loss Components</th>
<th>MAP@5000</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base</td>
<td>IC</td>
</tr>
<tr>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

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.
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 0.9\(m_1\) for CIFAR-10 and 0.1\(m_1\) for FLICKR25K. In Fig.4(c)(d), we study the effect of \(\lambda\), which denotes the weights of component correlation. We find that performance gain is brought when \(\lambda\) smaller than 0.1 and \(\lambda = 0.1\) are a good choice in two datasets.

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.

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


