Self-Paced Relational Contrastive Hashing for Large-Scale Image Retrieval

Zhengyun Lu, Lu Jin, Zechao Li, *Senior Member, IEEE*, and Jinhui Tang, *Senior Member, IEEE*

***Abstract*—Supervised deep hashing aims to learn hash functions using label information. Existing methods learn hash functions by employing either pairwise/triplet loss to explore the point-to-point relation or center loss to explore the point-to-class relation. However, these methods overlook the collaboration between the above two kinds of relations and the hardness of pairs. In this work, we propose a novel Self-Paced Relational Contrastive Hashing (SPRCH) method with a single learning objective to capture valuable discriminative information from hard pairs using both the point-to-point and point-to-class relations. To exploit the above two kinds of relations, the Relational Contrastive Hash (RCH) loss is proposed, which ensures that each data anchor is closer to all similar data points and corresponding class centers in the Hamming space compared to dissimilar ones. Moreover, the proposed RCH loss reduces the drastic imbalance between point-to-point pairs and point-to-class pairs by rebalancing their weights. To prioritize hard pairs, a self-paced learning schedule is proposed, assigning higher weights to these pairs in the RCH loss. The self-paced learning schedule assigns dynamic weights to pairs according to their similarities and the training process. In this way, deep hash model can initially learn universal patterns from the entire set of pairs and then gradually acquire more valuable discriminative information from hard pairs. Experimental results on four widely-used image retrieval datasets demonstrate that our proposed SPRCH method significantly outperforms the state-of-the-art supervised deep hash methods.**

***Index Terms*—Contrastive learning, deep hashing, image retrieval, self-paced learning.**

I. Introduction

HASHING is a binary coding technique that plays a crucial role in the large-scale image retrieval task. Hashing maps the high-dimensional image data to the low-dimensional Hamming space and produces short-bit binary codes. In this way, it greatly reduces the cost of storage and computation, thereby enhancing the efficiency of image retrieval. With the development of deep learning [1], [2], numerous deep hash methods [3], [4], [5], [6], [7], [8], [9], [10], [11], [12], [13], [14], [15], [16], [17], [18], [19], [20], [21], [22], [23], [24], [25], [26], [27], [28], [29], [30], [31], [32], [33], [34], [35], [36], [37] are proposed to learn feature representations and hash codes simultaneously. Compared with shallow hash methods [38], [39], [40], [41], [42], [43], deep hash methods generate high-quality hash codes by leveraging deep neural networks to capture more discriminative information.

Existing deep hash methods [13], [17], [19], [21], [24], [25], [27], [28], [31], [33], [35], [36] learn hash functions either using pairwise loss [17], [19], [21], [24] and triplet loss [25], [27] to capture the point-to-point relation, or using the center loss [13], [28], [31], [33], [36] to capture the point-to-class relation. The main idea of these methods is to make hash codes of similar data points close, and hash codes of dissimilar data points far away. However, they only consider the point-to-point relation or point-to-class relation respectively, but ignoring their collaboration. Fig. 1(a) illustrates the example of hash codes learned using the point-to-point relation. As shown, the hash codes of similar data points are not intra-class compact, resulting in increased Hamming distances and degraded retrieval accuracy. Fig. 1(b) shows the example of hash codes learned using the point-to-class relation. By solely considering the point-to-class relation, data points far from their class centers will be misclassified, which prevents them from clustering around their class centers. Fig. 1(c) shows the example of hash codes learned by exploring the collaboration between both kinds of relations (i.e. the proposed hash loss in this work). As we can see in Fig. 1(c), the hash codes of all similar data points are compactly clustered around corresponding class centers. Thus, it is necessary for deep hash methods to explore the point-to-point and point-to-class relations simultaneously.

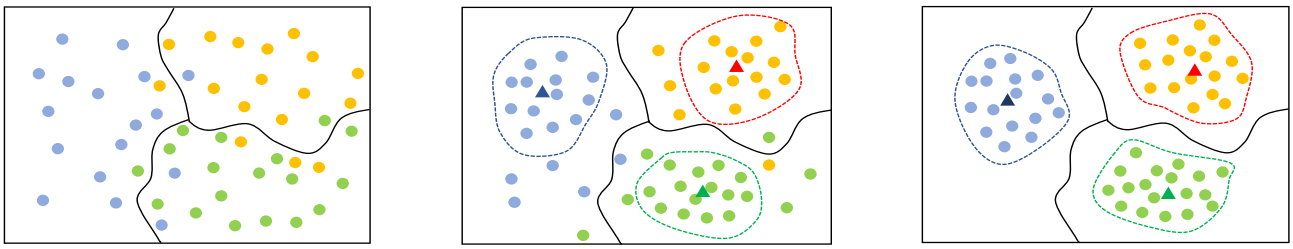
Furthermore, these deep hash methods treat all pairs equally during the learning process. However, the hardness of pairs is inconsistent. Some pairs are simple and easy to learn, while others are challenging and difficult to learn. As illustrated in Fig. 1(a) and (b), data points located near the decision boundary are frequently misclassified into irrelevant classes. These points contain more valuable discriminative knowledge and are more difficult to learn. As simple pairs occupy the majority of the training data, learning all pairs equally makes hard pairs easily ignored in the learning process, thus degrading the retrieval accuracy. Consequently, effectively leveraging hard pairs presents a challenging problem in the field of deep hashing.

Manuscript received 2 May 2023; revised 5 July 2023; accepted 16 August 2023. Date of publication 30 August 2023; date of current version 14 February 2024. This work was supported in part by the National Key Research and Development Program of China under Grant 2018AAA0102002, in part by the National Natural Science Foundation of China under Grants U20B2064 and 62102181, and in part by the Fundamental Research Funds for the Central Universities under Grant 30923010915. The Associate Editor coordinating the review of this manuscript and approving it for publication was Dr. Klaus Schoeffmann. *(Corresponding author: Lu Jin.)*

The authors are with the School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing 210094, China (e-mail: zhengyunlu@njust.edu.cn; lu.jin@njust.edu.cn; zechao.li@njust.edu.cn; jinhuitang@njust.edu.cn).

The source code of our work is available at https://github.com/IMAG-LZY/SPRCH.

Digital Object Identifier 10.1109/TMM.2023.3310333



(a) pairwise loss or triplet loss (b) center loss (c) our proposed loss

Fig. 1. Illustration of hash codes learned using different hash losses. Compared with the pairwise loss, triplet loss and center loss, our proposed loss considers both kinds of relations. Different colors represent different classes. The filled circles represent the hash codes of data points, the filled triangles represent the class centers of data points.

To address the aforementioned issues, we propose a novel deep hash method called Self-Paced Relational Contrastive Hashing (SPRCH) for large-scale image retrieval. The proposed SPRCH method aims to learn hash functions with a single learning objective, exploring hard pairs with both point-to-point and point-to-class relations simultaneously. Firstly, we propose a novel Relational Contrastive Hashing (RCH) loss to effectively explore these two types of relations. The proposed RCH loss ensures that each data anchor is close to all similar data points and corresponding class centers in the Hamming space, while being far away from dissimilar data points and irrelevant class centers. Considering the significant imbalance between point-to-point pairs and point-to-class pairs, we address this issue by adding the weighting factors to the RCH loss, specifically enlarging the weight of point-to-class pairs. Consequently, this results in the compact clustering of all similar data points around their respective class centers in the Hamming space. Secondly, we propose a self-paced learning schedule to exploit hard pairs in the proposed RCH loss. At the beginning of training process, all pairs are assigned equal weights to facilitate the deep hash model in learning universal patterns. Subsequently, higher weights are given to hard pairs according to their hardness to capture more valuable discriminative information from them. The retrieval performance is evaluated on four large-sale image retrieval datasets, and the experimental results demonstrate that our proposed method outperforms the state-of-the-art supervised deep hash methods.

In summary, the main contributions of our work are as follows:

- We propose a novel Self-Paced Relational Contrastive Hash (SPRCH) method with a single learning objective that simultaneously explores the point-to-point and point-to-class relations to mine hard pairs.
- We propose a Relational Contrastive Hash (RCH) loss that aims to maximize the agreement of all similar data points and corresponding class centers for each data anchor. Additionally, weighting factors are added to the RCH loss to address the significant imbalance between point-to-point pairs and point-to-class pairs.
- A self-paced learning schedule is introduced for the RCH loss to gradually capture valuable discriminative information from hard pairs.

II. Related Work

A. Supervised Deep Hashing

With the development of deep learning, supervised deep hashing [13], [17], [19], [21], [24], [25], [27], [28], [31], [33], [35], [36] plays a very important role in the image retrieval task. Existing hash methods use different kinds of hash losses to learn hash functions, which can be roughly divided into pairwise loss [17], [19], [21], [24], triplet loss [25], [27], and center loss [13], [28], [31], [33], [36].

Pairwise loss and triplet loss both learn hash functions using the point-to-point relation. For pairwise loss, given a data anchor, a positive/negative data point in the minibatch is selected to form a positive/negative pair with it. By minimizing pairwise loss, the Hamming distance of the positive pair will be small, and that of the negative pair will be large. DSH [17] is a classical hash method based on pairwise loss, which uses Euclidean distance to measure the Hamming distance of hash codes. HashNet [19] proposes a weighted cross-entropy loss to balance positive and negative samples and addresses the ill-posed gradient problem by smoothing the sign function. DCH [21] designs a pairwise cross-entropy loss based on Cauchy distribution, which can penalize similar image pairs if their Hamming distances are larger than a given threshold. IDHN [24] introduces a pairwise quantified similarity to reflect the fine-grained similarity between multi-label image pairs. However, pairwise loss does not consider the relative relations between positive samples and negative samples, which makes their boundary blurry. Therefore, triplet loss is proposed to solve this issue. For triplet loss, given a data anchor, a positive data point and a negative data point in the minibatch are selected to form a triplet pair with it. By minimizing triplet loss, the Hamming distance of positive pairs will be smaller than that of negative pairs. Both DSRH [25] and DRSCH [27] learns hash functions using triplet loss to generate high-quality hash codes. Whether pairwise loss or triplet loss only considers the point-to-point relation, which makes the hash codes of similar data are not intra-class compact.

Center loss explores point-to-class relation to learn hash functions. For center loss [13], [28], [31], [33], [36], given a data anchor, the corresponding class centers are selected to form positive pairs with it, irrelevant class centers are selected to form negative pairs with it. By minimizing center loss, the Hamming

distance of positive pairs will be small, and the Hamming distance of negative pairs will be large. How to generate class centers is different for different works. GreedyHash [28] uses one-hot label vectors as class centers. DPN [31], CSQ [13] and OrthoHash [33] construct class centers by leveraging the Hadamard matrix and Bernoulli distribution. Considering the fixed class centers lack the perception of data distribution, LSCSH [36] constructs self-adaptive class centers using the label semantic information and a generator module. Center loss only considers point-to-class relation, which makes data points away from corresponding class centers misclassified in the Hamming space.

The hash methods discussed above do not consider the collaboration of the point-to-point and point-to-class relations, and also ignores the hardness of pairs. To solve these issues, a Self-Paced Relational Contrastive Hashing (SPRCH) method is proposed in this work. Firstly, a novel Relational Contrastive Hash (RCH) loss is proposed to maximize the agreement of positive point-to-point and point-to-class pairs, while minimizing the agreement of negative pairs. Additionally, weighting factors are added to the proposed RCH loss to balance the above two kinds of pairs. In this way, all similar data points will be compactly clustered around corresponding class centers. Secondly, a self-paced learning schedule is proposed to gradually increase the attention of the deep hash model on hard pairs, enabling the model to capture more valuable discriminative information from hard pairs.

B. Contrastive Learning

Contrastive learning is popular in self-supervised learning recently, and performs well in various computer vision tasks [44], [45], [46], [47], [48], [49], [50], [51]. The main idea of contrastive learning is to form positive/negative data pairs with data augmentations, and then pull positive data pairs close, push negative data pairs away. SimCLR series [44], [45] and MoCo series [46], [47], [48] are the classical methods of them. For SimCLR series, they capture more negative data points by setting a large batch size. Differently, MoCo series propose a memory queue to store more negative data points, which greatly reduces the cost of storage and computation. Both SimCLR series and MoCo series focus on self-supervised learning tasks, where training data points are lake of label information. SCL [52] and ContraGan [53] extend the self-supervised contrastive learning to supervised tasks. SCL [52] proposes a supervised contrastive loss by comparing each data anchor with all positive/negative data points in the minibatch, which considers the point-to-point relation and achieves better performance in the image classification task. ContraGan [53] proposes a conditional contrastive loss for image generation task, which considers the relations between data points, as well as the relations between data and corresponding class centers.

Contrastive learning is also widely used in unsupervised deep hash methods [54], [55], [56], [57]. DUCH [54] introduces a cross-modal contrastive loss, which maximizes the agreement of intra-modal pairs and inter-modal pairs simultaneously. CMH [55] proposes a novel momentum optimizer to make binary operation practicable in contrastive learning, and designs a cross-modal ranking learning loss to alleviate the bad influence induced by false-negative pairs. To address the mismatch be- tween continuous representations and binary codes, CIBHash [56] introduces a probabilistic binary representation layer into deep hash model, which can minimize contrastive loss and reduce the mutual information between the binary codes and continuous representations at the same time. MeCoQ [57] adopts a debiased framework to avoid sampling bias and proposes a novel memory bank for quantization codes, which can provide more negative samples for contrastive loss.

Different from the above unsupervised contrastive hash methods, the proposed SPRCH method in this work introduces a novel contrastive hash loss in the supervised manner. Our proposed contrastive hash loss explores both the point-to-point and point-to-class relations and the hardness of pairs.

C. Self-Paced Learning

Self-paced learning [58], [59] is an adjustable learning schedule, which makes deep neural network learn training data points from easy to hard gradually. Self-paced learning proposes that the learning order of data points is determined by their hardness and their hardness is dynamically adjustable during the training process. In each training epoch, self-paced learning [58], [59] measures the hardness of each data point first, and then determines whether the data point participates in the training process according to an adjustable hardness threshold. How to measure the hardness of data points and design an adjustable hardness threshold are the main challenges of self-paced learning.

Self-paced learning is widely used in various deep learning works [60], [61], [62], [63], [64]. Among them, SSAH [64] designs a novel self-paced hard generation schedule for semi-supervised deep hashing. It gradually increases the difficulty of generated samples, which makes deep hash model learn generated samples from simple to hard. In this work, a different self-paced learning schedule is proposed for the proposed RCH loss. It assigns dynamic weights to pairs according to their hardness and the training process. At the beginning of training process, the deep hash model can initially learn universal patterns from the entire set of pairs and then gradually acquire more valuable discriminative information from hard pairs.

III. Method

Our method consists of three parts: a deep hash model, a relational contrastive hash loss, and a self-paced learning schedule. The overall framework is illustrated in Fig. 2. In Section III-A, we introduce the hashing problem we aim to address in this work and describe the design of our deep hash model. In Section III-B, we propose a novel relational contrastive hash loss to learn hash functions by exploring both the point-to-point and point-to-class relations. In Section III-C, we introduce a self-paced learning schedule to capture valuable discriminative information from hard positive/negative pairs for each data anchor in the proposed hash loss.

A. Deep Hash Model

We have an image dataset $\mathcal{D} = \{\boldsymbol{x}_i, \boldsymbol{y}_i\}_{i=1}^{n}$, where each $\boldsymbol{x}_i$ represents a raw image, and $\boldsymbol{y}_i = \{y_{ik}\}_{k=1}^{m} \in \{0,1\}^m$ denotes

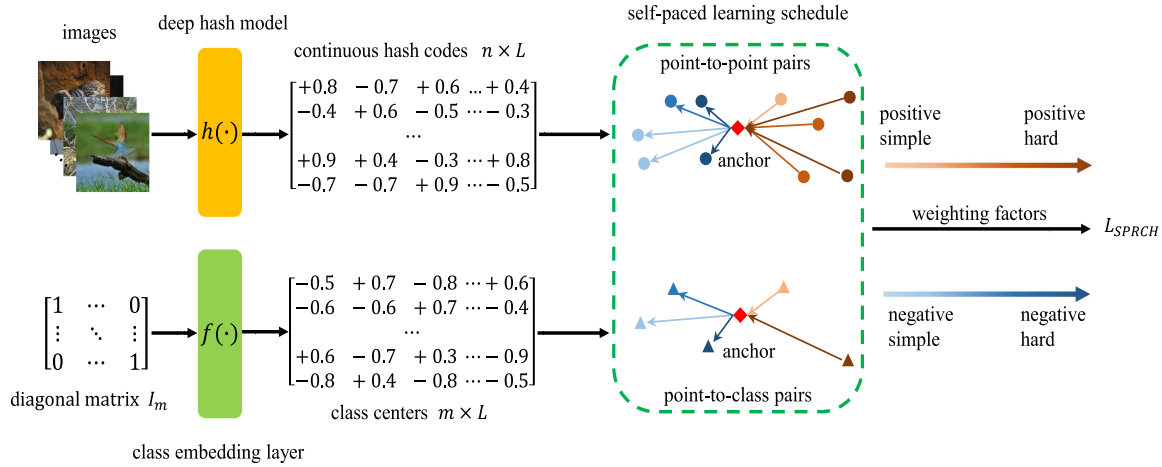


Fig. 2. Illustration of our proposed SPRCH method. Continuous hash codes and class centers are respectively generated from the deep hash model $h(\cdot)$ and the class embedding layer $f(\cdot)$. In order to explore the point-to-point and point-to-class relations simultaneously, we maximize the agreement of all similar data points and corresponding class centers for each anchor using contrastive learning. As the hardness of pairs is inconsistent, a self-paced learning schedule is introduced to make the deep hash model $h(\cdot)$ pay more attention on hard pairs. Due to the reason that the number of point-to-point pairs is much larger than that of point-to-class pairs, weighting factors are added in the self-paced relational contrastive hash loss to reduce the drastic imbalance.

the class label of $\boldsymbol{x}_i$, with m being the number of classes. If $y_{ik} = 1$, it means that the image $\boldsymbol{x}_i$ belongs to the class k; otherwise $y_{ik} = 0$. Our objective is to learn a hash function that can map a raw image $\boldsymbol{x}_i$ to an L-bit Hamming space and then generate hash codes $\boldsymbol{b}_i \in \{-1, 1\}^L$.

In this work, the hash function consists of two components: a deep hash model $h(\cdot)$ and an activation function $sign(\cdot)$. The deep hash model $h(\cdot)$ is a convolutional neural network, which maps each image $\boldsymbol{x}_i$ to the L-dimensional feature space. The activation function $sign(\cdot)$ is used to transform continuous L-dimensional feature representations into binary L-bit hash codes. Therefore, for each image $\boldsymbol{x}_i$, we define its corresponding hash codes $\boldsymbol{b}_i$ as follows:

$$\boldsymbol{b}_i = sign(h(\boldsymbol{x}_i)) \tag{1}$$

Due to the binary constraint caused by the activation function $sign(\cdot)$, deep hash model can not be trained with back propagation algorithm. To address this issue, the activation function $tanh(\cdot)$ is used to replace $sign(\cdot)$ during the training process. Thus, in the training process, we reformulate $\boldsymbol{b}_i$ as:

$$\boldsymbol{b}_i = tanh(h(\boldsymbol{x}_i)) \tag{2}$$

B. Relational Contrastive Hash Loss

Most existing supervised deep hash methods learn hash functions using the pairwise loss, triplet loss or center loss. On one hand, the pairwise loss and triplet loss only explore the point-to-point relation, resulting in the hash codes that are not compact within the same class. On the other hand, the center loss only preserves the point-to-class relation, but leading to misclassification of data points that are distant from the corresponding class centers. In order to make all similar data points compactly clustered around corresponding class centers, we propose the Relational Contrastive Hash (RCH) loss, which simultaneously explores the point-to-point and point-to-class relations.

Given a data anchor $\boldsymbol{x}_i$, sets $\Phi_i^+ = \{\boldsymbol{x}_j^+ | \boldsymbol{y}_j \boldsymbol{y}_i^T > 0\}_{j=1}^n$ and $\Phi_i^- = \{\boldsymbol{x}_j^- | \boldsymbol{y}_j \boldsymbol{y}_i^T = 0\}_{j=1}^n$ are defined as the collection of similar and dissimilar hash codes respectively. In order to explore the point-to-class relation, class centers $\mathbf{C} = tanh(f(I_m))$ are generated using a class embedding layer $f(\cdot)$ and a diagonal matrix I_m. Analogously, $\Psi_i^+ = \{\boldsymbol{c}_k^+ | y_{ik} = 1\}_{k=1}^m$ and $\Psi_i^- = \{\boldsymbol{c}_k^- | y_{ik} = 0\}_{k=1}^m$ are defined as the set of corresponding class centers and the set of irrelevant class centers respectively.

In the Hamming space, the RCH loss aims to maximize the agreement between each data anchor $\boldsymbol{x}_i$ and the positive samples in sets Φ_i^+ and Ψ_i^+, while minimizing the agreement with the negative samples in sets Φ_i^- and Ψ_i^-. The proposed RCH loss can be represented as

$$L_{RCH} = -\frac{1}{n} \sum_{\boldsymbol{x}_i \in \mathcal{D}} \log \left(\frac{P_i}{N_i + P_i} \right) \tag{3}$$

where P_i and N_i are the exponential sum of similarities for positive and negative pairs associated with the anchor $\boldsymbol{x}_i$. Specifically, P_i and N_i are defined as

$$\begin{aligned} P_i &= \sum_{\boldsymbol{x}_j^+ \in \Phi_i^+} \exp(S(\boldsymbol{b}_i, \boldsymbol{b}_j^+)/\tau) + \sum_{\boldsymbol{c}_k^+ \in \Psi_i^+} \exp(S(\boldsymbol{b}_i, \boldsymbol{c}_k^+)/\tau) \\ N_i &= \sum_{\boldsymbol{x}_j^- \in \Phi_i^-} \exp(S(\boldsymbol{b}_i, \boldsymbol{b}_j^-)/\tau) + \sum_{\boldsymbol{c}_k^- \in \Psi_i^-} \exp(S(\boldsymbol{b}_i, \boldsymbol{c}_k^-)/\tau) \end{aligned} \tag{4}$$

where $S(\boldsymbol{a}, \hat{\boldsymbol{a}}) = \frac{\boldsymbol{a}\hat{\boldsymbol{a}}^T}{\|\boldsymbol{a}\|_2 \|\hat{\boldsymbol{a}}\|_2}$ is the cosine similarity between the pair $(\boldsymbol{a}, \hat{\boldsymbol{a}})$, and τ is the temperature factor.

Compared with pairwise loss and triplet loss, our proposed RCH loss compares each data anchor $\boldsymbol{x}_i$ with all positive and negative data samples in the minibatch. Unlike center loss, our proposed RCH loss considers both class centers and data points in the minibatch. Considering these two kinds of relations, our

proposed RCH loss ensures that the generated hash codes are compactly clustered around the corresponding class centers.

For a given data anchor $\boldsymbol{x}_i$, the number of point-to-point pairs is usually much larger than that of point-to-class pairs. This significant imbalance disregards the gradient of point-to-class pairs during the training process. To address this issue and enlarge the gradient of point-to-class pairs, two weighting factors are added to point-to-class pairs in the proposed RCH loss. The weighting factors are positively correlated with the number of point-to-point pairs and can be calculated as

$$\begin{aligned} \lambda_i^+ &= |\Phi_i^+|/|\Psi_i^+| \\ \lambda_i^- &= |\Phi_i^-|/|\Psi_i^-| \end{aligned} \tag{5}$$

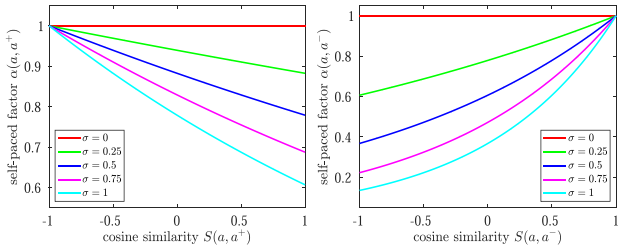
where $|A|$ calculates the number of samples in the set A. Consequently, the exponential sums of similarities in (4) can be reformulated as

$$\begin{aligned} P_i &= \sum_{\boldsymbol{x}_j^+ \in \Phi_i^+} \exp(S(\boldsymbol{b}_i, \boldsymbol{b}_j^+)/\tau) + \lambda_i^+ \sum_{\boldsymbol{c}_k^+ \in \Psi_i^+} \exp(S(\boldsymbol{b}_i, \boldsymbol{c}_k^+)/\tau) \\ N_i &= \sum_{\boldsymbol{x}_j^- \in \Phi_i^-} \exp(S(\boldsymbol{b}_i, \boldsymbol{b}_j^-)/\tau) + \lambda_i^- \sum_{\boldsymbol{c}_k^- \in \Psi_i^-} \exp(S(\boldsymbol{b}_i, \boldsymbol{c}_k^-)/\tau) \end{aligned} \tag{6}$$

As we can seen in (4), without considering weighting factors, the exponential sum of similarities for point-to-point pairs overwhelms that of point-to-class pairs. It makes the gradient of point-to-class pairs significantly smaller than that of point-to-point pairs, impeding the deep hash model's ability to exploit the point-to-class relation. A solution to address this issue is to introduce two weighting factors, λ_i^+ and λ_i^-, into our proposed RCH loss, as presented in (6). The inclusion of these weighting factors enhances the gradient of point-to-class pairs, enabling the deep hash model to effectively learn discriminative information from class centers.

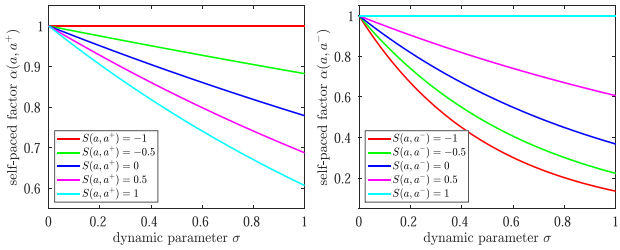
C. Self-Paced Learning Schedules

The proposed RCH loss simultaneously explores the point-to-point relation and the point-to-class relations via contrastive learning. However, the current RCH loss treats all pairs equally and fails to consider their hardness. Hard pairs contain more discriminative information than simple pairs. To better utilize this information, hard pairs should be reweighted with higher weights during the training process. Moreover, only focusing on hard pairs will cause the deep hash model to overlook universal patterns across all pairs. To solve the aforementioned issues, a self-paced learning schedule is introduced to reweight pairs gradually in the proposed RCH loss. For each pair, we design a self-paced factor that considers the hardness of the pair and the training process. At the beginning of the training process, the values of self-paced factors are close to each other, allowing the deep hash model to learn universal patterns from all the pairs. Subsequently, the self-paced factors for hard pairs will be larger than that for simple pairs, enabling the deep hash model to gradually focus more on hard pairs. Given a positive pair $(\boldsymbol{a}, \boldsymbol{a}^+)$ or a negative pair $(\boldsymbol{a}, \boldsymbol{a}^-)$, the self-paced factor can be calculated as



(a) positive pairs (b) negative pairs

Fig. 3. Curves between the self-paced factors and the cosine similarity.



(a) positive pairs (b) negative pairs

Fig. 4. Curves between the self-paced factors and the dynamic parameter σ.

$$\begin{cases} \alpha(\boldsymbol{a}, \boldsymbol{a}^+) = \exp(-1 - S(\boldsymbol{a}, \boldsymbol{a}^+))^{\sigma/4}, \\ \alpha(\boldsymbol{a}, \boldsymbol{a}^-) = \exp(-1 + S(\boldsymbol{a}, \boldsymbol{a}^-))^{\sigma}. \end{cases} \tag{7}$$

where the cosine similarity measures the hardness of the pair, and $\sigma \in [0, 1]$ is a dynamic parameter which changes according to the training process. Specifically, σ can be calculated as

$$\sigma = \begin{cases} T/T_{th}, & 0 < T \leq T_{th} \\ 1, & T_{th} < T \leq T_{\max} \end{cases} \tag{8}$$

where T is the current training epoch, $T_{\max}$ is the total number of training epochs, and $T_{th} = T_{\max}/3$.

The self-paced factor proposed in (7) is associated with the cosine similarity and the dynamic parameter σ. Fig. 3 illustrates the curves between the self-paced factor and the cosine similarity when the dynamic parameter σ is fixed. In Fig. 3(a), as the cosine similarity of the positive pair $S(\boldsymbol{a}, \boldsymbol{a}^+)$ increases, the self-paced factor $\alpha(\boldsymbol{a}, \boldsymbol{a}^+)$ decreases. When the value of $S(\boldsymbol{a}, \boldsymbol{a}^+)$ is small, this pair will be regraded as a hard pair and assigned a large self-paced factor $\alpha(\boldsymbol{a}, \boldsymbol{a}^+)$. Similarly, in Fig. 3(b), for a negative pair $(\boldsymbol{a}, \boldsymbol{a}^-)$, if its cosine similarity $S(\boldsymbol{a}, \boldsymbol{a}^-)$ is large, it will also be regraded as a hard pair and given a large self-paced factor $\alpha(\boldsymbol{a}, \boldsymbol{a}^-)$. Fig. 4 illustrates the curves between the self-paced factor and the dynamic parameter σ for a fixed value of cosine similarity. As shown in Fig. 4, there exists a negative correlation between the self-paced factor and the dynamic parameter. At the beginning of the training process, the dynamic parameter σ is close to 0, resulting in all the self-paced factors being close to 1. This enables the deep hash model learn all pairs equally. After that, the dynamic parameter σ increases gradually, which makes self-paced factors for hard pairs larger than that of simple pairs. The deep hash model can gradually pay more attention to hard pairs.

Algorithm 1: The Training Process of Our Proposed Self-Paced Contrastive Hashing Method.

Input: Training image dataset: $\mathcal{D} = \{\boldsymbol{x}_i, \boldsymbol{y}_i\}_{i=1}^{n}$, parameters τ, n_b, T_{max}.
Output: The trained deep hash model $h(\cdot)$.

Preprocess and normalize the training dataset $\mathcal{D}$;
Initialize the deep hash model $h(\cdot)$ with the pre-trained ResNet18;
Initialize the class embedding layer $f(\cdot)$;
repeat
  Randomly select a minibatch dataset $\{\boldsymbol{x}_i, \boldsymbol{y}_i\}_{i=1}^{n_b}$ from the whole dataset $\mathcal{D}$;
  Compute continuous hash codes $\{\boldsymbol{b}_i\}_{i=1}^{n_b}$ with $h(\cdot)$;
  Compute class centers $\mathbf{C}$ with $f(\cdot)$;
  Compute the exponential similarity sum P_i and N_i for each data anchor $\boldsymbol{x}_i$ with Eq. (4);
  Compute weighting factors λ_i^+ and λ_i^- for each data anchor $\boldsymbol{x}_i$ with Eq. (5);
  Update the self-paced factorss α for all pairs with Eq. (7);
  Compute the proposed hash loss with Eq. (10);
  Back propagation;
  Update $h(\cdot)$ and $f(\cdot)$.
until *reaching the maximun training epoch* T_{max};

Therefore, the self-paced weighted exponential sum of similarities for all positive/negative pairs for data anchor x_i can be reformulated as

$$\begin{aligned} P_i^{SP} = & \sum_{\boldsymbol{x}_j^+ \in \Phi_i^+} \alpha\left(\boldsymbol{b}_i, \boldsymbol{b}_j^+\right) \exp\left(S\left(\boldsymbol{b}_i, \boldsymbol{b}_j^+\right)/\tau\right) \\ & + \lambda_i^+ \sum_{\boldsymbol{c}_k^+ \in \Psi_i^+} \alpha\left(\boldsymbol{b}_i, \boldsymbol{c}_k^+\right) \exp\left(\left(S\left(\boldsymbol{b}_i, \boldsymbol{c}_k^+\right)/\tau\right)\right. \\ N_i^{SP} = & \sum_{\boldsymbol{x}_j^- \in \Phi_i^-} \alpha(\boldsymbol{b}_i, \boldsymbol{b}_j^-) \exp(S(\boldsymbol{b}_i, \boldsymbol{b}_j^-)/\tau) \\ & + \lambda_i^- \sum_{\boldsymbol{c}_k^- \in \Psi_i^-} \alpha(\boldsymbol{b}_i, \boldsymbol{c}_k^-) \exp((S(\boldsymbol{b}_i, \boldsymbol{c}_k^-)/\tau) \end{aligned} \tag{9}$$

The self-paced relational contrastive hash loss can be represented as

$$L_{SPRCH} = -\frac{1}{n} \sum_{x_i \in \mathcal{D}} \log\left(\frac{P_i^{SP}}{N_i^{SP} + P_i^{SP}}\right) \tag{10}$$

According to the above analysis, we summarize the training process of our proposed self-paced contrastive hashing method in Algorithm 1.

IV. Experiment

A. Dataset

Following the previous works [13], [17], [19], [31], [33], in our experiments, we choose four widely-used large-scale image retrieval datasets, including CIFAR-100 [65], ImageNet-100 [66], COCO [67] and NUS-WIDE [68].

TABLE I
Experiment Settings for Four Large-Scale Image Datasets

Dataset	Class	Training set	Query set	Database set
CIFAR-100	100	10,000	10,000	50,000
ImageNet-100	100	13,000	5,000	128,503
COCO	80	10,000	5,000	117,218
NUS-WIDE	21	10,500	2,100	193,734

- *CIFAR-100 [65]:* CIFAR-100 is a single-label image dataset that belongs to 100 classes. It consists of 60,000 colored tiny images, which are officially split into 50,000 training images and 10,000 test images.
- *ImageNet-100 [66]:* ImageNet-100 is a subset of ImageNet dataset and contains 100 classes. It is widely used in classification, location and retrieval tasks. In the official split, its training set and validation set respectively contain 128,503 images and 5,000 images.
- *COCO [67]:* COCO is a popular large-scale multi-label image dataset including 80 classes and has more diverse and complex scenes than other darasets. Its official training and validation sets respectively contain 82,081 tagged images and 40,137 tagged images.
- *NUS-WIDE [68]:* NUS-WIDE is a large-scale web image dataset with 81 real concepts, which contains 269,648 multi-label images. Following the previous works [13], [17], [19], [31], [33], we choose the subset of 195,834 images from top-21 frequent concepts in the entire dataset.

In this work, we divide each dataset into three sets: the training set, the query set and the database set, where the training set is used to learn hash functions, the query set and the database set are used to evaluate the retrieval performance of hash functions. The experiment settings of different datasets can be seen in Table I. For CIFAR-100, the official training and test sets respectively serve as the database set and the query set. 10,000 images from the database set are selected as the training set, with 100 images per class. For ImageNet-100, the official training set is used as the database set, from which 13,000 images are selected to form the training set, with 130 images per class. The official validation set is used as our query set. For COCO, from both official training set and validation set, we randomly select 5,000 images as the query set and select the remaining 117,218 images as the database set. From the database set, 10,000 images are randomly selected for training. For NUS-WIDE, 100 images from each class are selected as the query set and the remaining 193,734 images are used as the database set, from which 500 images per class are selected to form the training set.

B. Compared Methods

We compare the retrieval performance of our method with eleven hash methods, including three shallow hash methods (SH [38], ITQ [41] and SDH [43]), and eight deep hash methods (DSH [17], HashNet [17], DCH [21], IDHN [24], GreedyHash [28], DPN [31], CSQ [13] and OrthoHash [33]).

- *SH [38]:* It learns hash functions by exploring the neighbor structures of the training data.

- *ITQ [41]:* It learns hash functions by minimizing the quantization error between hash codes and PCA-projected data features.
- *SDH [43]:* It proposes a discrete cyclic coordinate descent (DCC) algorithm to solve the problem of discrete constraints for binary hash codes.
- *DSH [17]:* It learns hash functions with the relations of pairwise data points, and uses the Euclidean distance to measure the similarity between hash codes.
- *HashNet [19]:* It proposes a weighted cross-entropy loss to balance positive/negative samples and addresses the ill-posed gradient problem by smoothing the sign function.
- *DCH [21]:* It proposes a cross-entropy loss based on Cauchy distribution, which can penalize similar image pairs if their Hamming distance larger than the given threshold.
- *IDHN [24]:* It introduces a soft pairwise similarity metric to reflect the fine-grained similarities between multi-label images.
- *GreedyHash [28]:* It generates optimal discrete hash codes by iteratively updating the network using the greedy principle.
- *DPN [31]:* It proposes a differentiable hinge-like loss to learn hash functions, which makes hash codes of dataset close to generated class centers.
- *CSQ [13]:* It generates class centers via Hadamard matrix and Bernoulli distribution in advance and then learns hash functions with the relations between class centers and hash codes.
- *OrthoHash [33]:* It proposes a novel deep hashing model with only a single learning objective, which can learn hash codes without the quantization loss.

C. Evaluation Metrics

To evaluate image retrieval performance of hash methods, we choose four standard evaluation metrics: Precision-Recall curves (PR), Mean Average Precision (mAP), Precision curves w.r.t. different numbers of returned samples (P@N) and Precision curves within Hamming radius 2 (P@H = 2).

- *PR:* Precision-Recall curves are the curves of precision and recall, which evaluate the overall retrieval performance of different hash methods.
- *mAP:* mAP is the mean of the Average Precision (AP) scores for all query images. mAP@1 K is adopted for CIFAR-100 and ImageNet-100, and mAP@5 K is adopted for COCO and NUS-WIDE.
- *P@N:* P@N curves are the precision curves of the top N returned results, where N is set within $\{100, 200, \ldots, 1,000\}$ for all datasets.
- *P@H = 2:* P@H = 2 curves are the precision curves of the returned results that fall within the Hamming radius 2.

D. Implementation Details

Our deep hash model $h(\cdot)$ chooses the pre-trained network ResNet18 as the backbone, and replaces the last classification layer with a hash layer, which transforms the 512-dimensional feature representations to L-bit continuous hash codes. Besides, we use a fully-connected layer as class embedding layer $f(\cdot)$ to generate class centers.

Our proposed hash method is implemented with the PyTorch framework. The training images are first resized to 256 × 256, and then randomly cropped to 224 × 224. The query images and database images are first resized to 256 × 256, and then center cropped to 224 × 224. The deep hash model $h(\cdot)$ and the class embedding layer $f(\cdot)$ are both trained with Adam optimizer. For deep hash model $h(\cdot)$, the learning rate of backbone is set to 1e-5, the learning rate of hash layer is set to 1e-4. For class embedding layer $f(\cdot)$, the learning rate is set to 1e-3. The number of training epoch $T_{\max}$ is set to 100, the batch size n_b is set to 128, and the temperature factor τ is set to 0.3.

To make a fair comparison, for compared shallow hash methods, we use the 512-dimensional output features of the pre-trained ResNet18 to learn hash functions. For compared deep hash methods, the backbone, image transformation and training epoch are the same as our proposed hash method. The experimental results of all compared methods are run by us using the released source codes in the GitHub.

E. Ablation Studies

In this section, we evaluate the effectiveness of our proposed deep hash method on three parts: the proposed RCH loss, the weighting factors and the self-paced learning schedule. The ablation studies are conducted on the single-label dataset CIFAR-100 and the multi-label dataset COCO, and their implementation details are the same as that in Section IV-D.

1) Ablation Studies of the Proposed RCH Loss: Our proposed RCH loss in (1) explores the similarity relations of the point-to-point and point-to-class simultaneously. To evaluate the effectiveness of the proposed RCH loss, we compare it with two variant contrastive hash losses: the point-to-point contrastive hash loss L_{p2p} and the point-to-class contrastive hash loss L_{p2c}. The point-to-point contrastive hash loss L_{p2p} only maximizes the agreement of each data anchor and all similar data points, while minimizes the agreement of each data anchor and all dissimilar data points, which can be represented as:

$$-\frac{1}{n}\sum_{\boldsymbol{x}_i\in\mathcal{D}}\log\left(\frac{\sum_{\boldsymbol{x}_j^+\in\Phi_i^+}\exp\left(\frac{S(\boldsymbol{b}_i,\boldsymbol{b}_j^+)}{\tau}\right)}{\sum_{\boldsymbol{x}_j^-\in\Phi_i^-}\exp\left(\frac{S(\boldsymbol{b}_i,\boldsymbol{b}_j^-)}{\tau}\right)+\sum_{\boldsymbol{x}_j^+\in\Phi_i^+}\exp\left(\frac{S(\boldsymbol{b}_i,\boldsymbol{b}_j^+)}{\tau}\right)}\right) \tag{11}$$

The point-to-class contrastive hash loss L_{p2c} only makes each data anchor close to all corresponding class centers, and away from all irrelevant class centers in the Hamming space, which can be represented as:

$$-\frac{1}{n}\sum_{\boldsymbol{x}_i\in\mathcal{D}}\log\left(\frac{\sum_{\boldsymbol{c}_k^+\in\Psi_i^+}\exp\left(\frac{S(\boldsymbol{b}_i,\boldsymbol{c}_k^+)}{\tau}\right)}{\sum_{\boldsymbol{c}_k^-\in\Psi_i^-}\exp\left(\frac{S(\boldsymbol{b}_i,\boldsymbol{c}_k^-)}{\tau}\right)+\sum_{\boldsymbol{c}_k^+\in\Psi_i^+}\exp\left(\frac{S(\boldsymbol{b}_i,\boldsymbol{c}_k^+)}{\tau}\right)}\right) \tag{12}$$

TABLE II
COMPARED RESULTS IN TERMS OF MAP (%) OF THE POINT-TO-POINT CONTRASTIVE HASH LOSS, THE POINT-TO-CLASS CONTRASTIVE HASH LOSS AND OUR PROPOSED RCH LOSS ON THE CIFAR-100 AND COCO DATASETS

Loss	Relation		CIFAR-100 mAP@1K				COCO mAP@5K			
	p2p	p2c	16 bits	32 bits	48 bits	64 bits	16 bits	32 bits	48 bits	64 bits
L_{p2p}	✓		48.1	57.0	60.4	61.0	81.1	83.0	83.7	84.1
L_{p2c}		✓	49.4	58.3	60.6	60.6	80.0	82.3	82.8	83.3
L_{RCH}	✓	✓	**50.9**	**60.6**	**62.6**	**62.9**	**82.8**	**84.9**	**85.6**	**86.0**

The best results are highlighted in bold.

TABLE III
COMPARED RESULTS IN TERMS OF MAP (%) OF THE PROPOSED RCH LOSS WITH AND WITHOUT WEIGHTING FACTORS ON THE CIFAR-100 AND COCO DATASETS

Loss	Weighting factors in Eq. (5)	CIFAR-100 mAP@1K				COCO mAP@5K			
		16 bits	32 bits	48 bits	64 bits	16 bits	32 bits	48 bits	64 bits
L_{RCH}		50.5	60.1	62.4	**62.9**	80.6	82.5	82.9	83.5
L_{RCH}	✓	**50.9**	**60.6**	**62.6**	**62.9**	**82.8**	**84.9**	**85.6**	**86.0**

The best results are highlighted in bold.

We show the experimental results in terms of mAP (%) across different hash bits and different image datasets in Table II. Compared with L_{p2p} and L_{p2c}, the proposed RCH loss achieves better performance. For example, on the CIFAR-100 dataset, our RCH loss respectively outperforms L_{p2p} and L_{p2c} by average margin of 2.5% and 2.0% in mAP. It confirms that if we only consider the point-to-point relation or the point-to-class relation, many valuable information will be ignored and the learned hash codes will be less discriminative. The proposed RCH loss focuses on the collaboration of the above two kinds of relations, which makes sure all similar data points will be compactly clustered around corresponding class centers in the Hamming space. Therefore, it is necessary to learn hash functions using both kinds of relations.

2) Ablation Studies of the Weighting Factors: In order to reduce the imbalance between the point-to-point pairs and point-to-class pairs, two weighting factors λ_i^+ and λ_i^- are added in the RCH loss by enlarging the weight of the point-to-class pairs. To evaluate the effectiveness of weighting factors, we compare the proposed RCH loss with and without weighting factors.

Table III illustrates the experimental results of the above two hash losses in terms of mAP (%) across different hash bits and different image datasets. The mAP results of the RCH loss with weighting factors achieves the average improvement of 2.5% over that without weighting factors on the COCO dataset, while the mAP results are close to each other on the CIFAR-100 dataset. In order to explore the reasons for the situation that the improvements on CIFAR-100 are less significant than COCO, we calculate the imbalance ratios between two kinds of positive/negative pairs in the minibatch for all data anchors. The imbalance ratios can be formulated as

$$ratio^+ = \frac{1}{n}\sum_{\boldsymbol{x}_i \in \mathcal{D}} \frac{|\Phi_i^+|}{|\Psi_i^+|}$$
$$ratio^- = \frac{1}{n}\sum_{\boldsymbol{x}_i \in \mathcal{D}} \frac{|\Phi_i^-|}{|\Psi_i^-|} \tag{13}$$

where $ratio^+$ is the imbalance ratios between two kinds of positive pairs, while $ratio^-$ between two kinds of negative pairs respectively. $|\Phi_i^+|$ and $|\Phi_i^-|$ are sets of data points sampling from the minibatch. The larger the value of the imbalance ratio, the more serious the imbalance between two kinds of pairs becomes.

According to (13), $ratio^+ = 2.26$ and $ratio^- = 1.27$ on the CIFAR-100 dataset, while $ratio^+ = 15.72$ and $ratio^- = 1.07$ on the COCO dataset. It indicates that the imbalance between two kinds of pairs is minor on the CIFAR-100 dataset, but drastic on the COCO dataset. Thus, the weighting factors do not perform well on the CIFAR-100 dataset but play an important role on the COCO dataset. This confirms that the bad influence of a significant imbalance between two kinds of pairs can be mitigated by employing the proposed weighting factors, resulting in an improvement of the RCH loss. Therefore, the weighting factors are indispensable for the RCH loss to balance two kinds of pairs.

3) Ablation Studies of the Self-Paced Learning Schedule: Compared with simple pairs, hard pairs contain more valuable information. In order to capture these valuable information gradually, for the proposed RCH loss, we propose a self-paced learning schedule, which reweights hard pairs with higher weights. To evaluate the effectiveness of the self-paced learning schedule, we compare the retrieval performance of the relational contrastive hash (RCH) loss with that of the self-paced relational contrastive hash (SPRCH) loss.

The experimental results can be seen in Table IV. Compared with the RCH loss, the SPRCH loss achieves better performance, especially on long-bit hash codes. For example, on the CIFAR-100 dataset, the mAP of the SPRCH loss outperforms that of the RCH loss by the margin of 1.3% on 64-bit hash codes. It confirms that the information contained in hard pairs is important. Using the self-paced learning schedule, the deep hash model will learn universal patterns from whole pairs at the beginning of the training process and then learn more valuable information from hard pairs gradually. Therefore, the self-paced learning schedule is necessary for the proposed RCH loss.

Furthermore, we analyze the impact of the hard-to-easy strategy on the proposed SPRCH loss. Specifically, we set the

TABLE IV
COMPARED RESULTS IN TERMS OF MAP (%) OF THE PROPOSED RCH LOSS AND SPRCH LOSS ON THE CIFAR-100 AND COCO DATASETS

Loss	Self-paced factors in Eq. (7)	CIFAR-100 mAP@1K				COCO mAP@5K			
		16 bits	32 bits	48 bits	64 bits	16 bits	32 bits	48 bits	64 bits
L_{RCH}		**50.9**	60.6	62.6	62.9	82.8	84.9	85.6	86.0
L_{SPRCH}	✓	**50.9**	**61.4**	**63.7**	**64.2**	**83.0**	**85.6**	**86.4**	**86.8**

The best results are highlighted in bold.

TABLE V
COMPARED RESULTS IN TERMS OF MAP (%) OF DIFFERENT KINDS OF SELF-PACED LEARNING STRATEGIES FOR SPRCH LOSS ON THE SINGLE-LABEL DATASET CIFAR-100 AND MULTI-LABEL DATASET COCO

Loss	Self-paced learning strategy	CIFAR-100 mAP@1K				COCO mAP@5K			
		16 bits	32 bits	48 bits	64 bits	16 bits	32 bits	48 bits	64 bits
L_{SPRCH}	hard-to-easy	50.5	61.0	62.8	63.4	**83.0**	85.4	85.9	86.1
L_{SPRCH}	easy-to-hard	**50.9**	**61.4**	**63.7**	**64.2**	**83.0**	**85.6**	**86.4**	**86.8**

The best results are highlighted in bold.

TABLE VI
COMPARED RESULTS IN TERMS OF MAP (%) OF DIFFERENT KINDS OF HASH METHODS FOR DIFFERENT HASH BITS ON THE SINGLE-LABEL DATASETS INCLUDING CIFAR-100 AND IMAGENET-100

Method			CIFAR-100 mAP@1K				ImageNet-100 mAP@1K			
			16 bits	32 bits	48 bits	64 bits	16 bits	32 bits	48 bits	64 bits
Shallow		SH [38]	9.9	13.7	15.6	16.8	27.8	47.1	55.1	57.2
		ITQ [41]	10.6	15.3	17.4	19.1	30.5	48.5	56.2	60.6
		SDH [43]	22.3	28.6	31.8	32.7	61.9	67.0	68.6	71.4
Deep	Point-to-point	DSH [17]	34.3	42.3	43.9	44.4	65.3	72.8	74.6	74.3
		HashNet [19]	24.7	36.5	43.5	46.1	45.5	57.7	66.0	66.9
		DCH [21]	39.9	44.8	42.6	39.7	73.1	76.7	75.8	73.8
		IDHN [24]	-	-	-	-	-	-	-	-
	Point-to-class	GreedyHash [28]	41.8	55.6	59.9	60.9	72.5	78.6	80.2	80.9
		DPN [31]	43.6	56.9	61.4	61.9	72.4	78.9	80.6	81.3
		CSQ [13]	<u>44.7</u>	<u>60.2</u>	<u>61.1</u>	<u>62.7</u>	73.6	80.1	80.2	81.8
		OrthoHash [33]	34.8	50.2	56.7	60.1	<u>75.5</u>	<u>81.5</u>	<u>82.7</u>	<u>83.4</u>
	Ours	SPRCH	**50.9**	**61.4**	**63.7**	**64.2**	**78.1**	**82.4**	**83.0**	**83.5**

Point-to-point and point-to-class respectively indicate the hash methods which learn hash functions exploring the point-to-point relation and the point-to-class relation. Ours is the proposed SPRCH method which learns hash functions exploring the collaboration of two kinds of relations and the hardness of pairs. All deep hash methods use ResNet18 as backbone. The best and second best results are highlighted in bold and underlined respectively.

self-paced factors positively related to the dynamic parameter σ, which can be reformulated as

$$\begin{cases} \alpha(\boldsymbol{a}, \boldsymbol{a}^+) = \exp(-1 - S(\boldsymbol{a}, \boldsymbol{a}^+))^{(1-\sigma)/4}, \\ \alpha(\boldsymbol{a}, \boldsymbol{a}^-) = \exp(-1 + S(\boldsymbol{a}, \boldsymbol{a}^-))^{1-\sigma}. \end{cases} \tag{14}$$

At the beginning of the training process, the dynamic parameter σ is close to 0, which makes self-paced factors for hard pairs larger than that for simple pairs. After that, the dynamic parameter σ increases gradually, which makes self-paced factors close to 1 and the deep hash model learn all pairs equally.

We conduct the experiment to analyze these two strategies and show the results in Table V. It can be observed that the mAP results of the easy-to-hard strategy are much better than that of hard-to-easy ones. This indicates the effectiveness of our proposed hard-to-easy strategy for self-paced learning schedule.

F. Compared With Previous Works

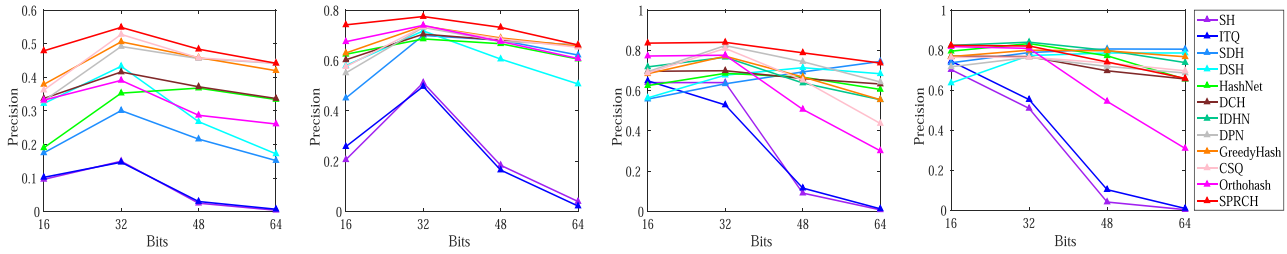
In this section, we compare the retrieval performance of our proposed SPRCH method with that of eleven previous hash methods, which are introduced in Section IV-B. The experimental results in terms of mAP for different hash bits on four large-scale image retrieval datasets can be seen in Tables VI and VII. We can draw the following observations from the experimental results.

First, compared with all shallow and deep hash methods, our proposed SPRCH method achieves much better retrieval performance, especially on short-bit hash codes. For example, for 16-bit hash codes, the mAP of the SPRCH method outperforms that of the best baseline method by the margin of 6.2% on the single-label CIFAR-100 dataset, and outperforms that of the best baseline method by the margin of 8.4% on the multi-label COCO dataset. Second, compared with shallow hash methods [38], [41], [43], the deep hash method performs better

TABLE VII
COMPARED RESULTS IN TERMS OF MAP (%) OF DIFFERENT KINDS OF HASH METHODS FOR DIFFERENT HASH BITS ON THE MULTI-LABEL DATASETS INCLUDING COCO AND NUS-WIDE

Method			COCO mAP@5K				NUS-WIDE mAP@5K			
			16 bits	32 bits	48 bits	64 bits	16 bits	32 bits	48 bits	64 bits
Shallow		SH [38]	62.3	63.7	63.9	63.4	68.2	65.6	65.6	64.1
		ITQ [41]	62.4	67.8	70.2	70.9	72.9	74.4	76.4	76.6
		SDH [43]	61.4	65.4	68.2	71.1	77.4	79.0	79.5	79.9
Deep	Point-to-point	DSH [17]	63.6	66.4	69.5	70.3	74.6	79.0	79.1	79.4
		HashNet [19]	67.1	68.9	70.4	71.6	82.0	84.5	85.3	85.5
		DCH [21]	71.9	74.6	75.2	75.7	80.1	81.1	81.0	81.2
		IDHN [24]	73.1	77.3	77.5	77.0	83.0	84.6	84.9	85.1
	Point-to-class	GreedyHash [28]	68.2	74.1	75.9	77.0	79.2	80.7	81.7	82.0
		DPN [31]	71.7	78.9	81.5	82.5	77.0	79.1	79.4	80.1
		CSQ [13]	70.6	79.7	81.1	84.0	79.2	80.4	79.9	80.9
		OrthoHash [33]	74.6	81.6	83.8	85.4	81.7	83.9	85.3	**85.9**
	Ours	SPRCH	**83.0**	**85.6**	**86.4**	**86.8**	**83.1**	**84.7**	**85.4**	**85.9**

Point-to-point and point-to-class respectively indicate the hash methods which learn hash functions exploring the point-to-point relation and the point-to-class relation. Ours is the proposed SPRCH method which learns hash functions exploring the collaboration of two kinds of relations and the hardness of pairs. All deep hash methods use ResNet18 as backbone. The best and second best results are highlighted in bold and underlined respectively.



(a) CIFAR-100 (b) ImageNet-100 (c) COCO (d) NUS-WIDE

Fig. 5. P@H = 2 curves of all hash methods with different bits on the CIFAR-100, ImageNet-100, COCO and NUS-WIDE datasets.

in most cases. This is mainly because that the features learned by deep neural networks contain more valuable information. Finally, our proposed SPRCH method achieves better retrieval results than those deep hash methods [13], [17], [19], [21], [24], [28], [31], [33] only considering one kind of relation and ignoring the hardness of pairs. It verifies the effectiveness of our motivation to explore the collaboration of two kinds of relations and the hardness of pairs. We also evaluate the retrieval performance of hash methods in terms of the P@H = 2 curves, PR curves and P@N curves. Fig. 5 shows the P@H = 2 curves of all hash methods with different hash bits. The precision of our proposed SPRCH method outperforms that of compared hash methods on the returned results within Hamming radius 2 in most cases. It indicates that the hash codes learned using our proposed SPRCH method are more compact than compared hash methods. The PR curves of all hash methods with 16-bit and 32-bit hash codes are respectively presented in Figs. 6 and 7. The PR curves of our proposed SPRCH method cover that of compared hash methods in most cases, which confirms that the entire retrieval performance of our proposed SPRCH method is better. Figs. 8 and 9 respectively illustrate the P@N curves of all hash methods with 16-bit and 32-bit hash codes. The precision of our proposed SPRCH method is larger than that of compared hash methods on the top 1,000 returned results for different datasets.

G. Parameter Analysis

Following the previous works [44], [46], [52], [53], we analyze the sensitivity of the temperature factor τ and batch size n_b in the proposed contrastive hash loss. To thoroughly investigate their influences, we conduct experiments on the CIFAR-100 and COCO datasets.

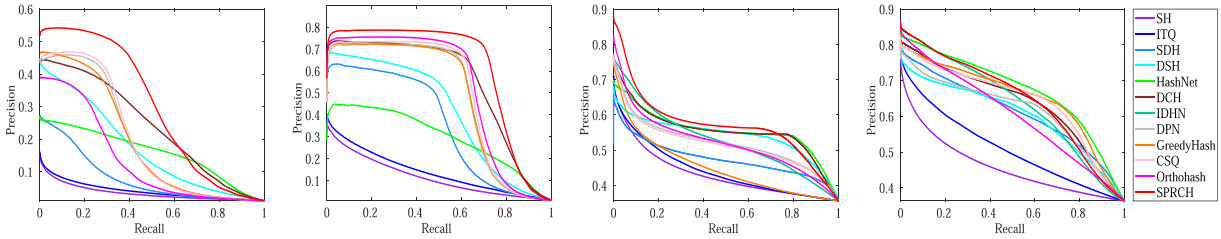
The value of temperature factor τ is set within $\{0.1, 0.2, 0.3, 0.4, 0.5\}$ for the proposed RCH loss in (1) and SPRCH loss in (12). The experimental results of 5 different temperature factors on the CIFAR-100 dataset can be seen in Table VIII. As we can see in Table VIII, we can simply find that the mAP of RCH loss or SPRCH loss increases gradually when the temperature factor τ is close to 0.3, and decreases gradually when the temperature factor τ is far away from 0.3. Therefore, in this work, we choose $\tau = 0.3$ for our proposed hash method.

we set the batch size n_b within $\{32, 64, 128, 256, 512\}$ for the proposed SPRCH loss in (12). The experimental results on the single-label dataset CIFAR-100 and multi-label dataset COCO are shown in Table IX. As we can see in Table IX, for the single-label dataset CIFAR-100, the mAP values will drop when increasing the batch size. For the multi-label dataset COCO, the best mAP result will be obtained when the batch size is close to 128. Therefore, we set the batch size to 128 for all the single-label and multi-label datasets.

TABLE VIII
COMPARED RESULTS IN TERMS OF MAP@1 K (%) OF OUR PROPOSED RCH LOSS AND SPRCH LOSS USING DIFFERENT TEMPERATURE FACTORS FOR DIFFERENT HASH BITS ON THE CIFAR-100 DATASET

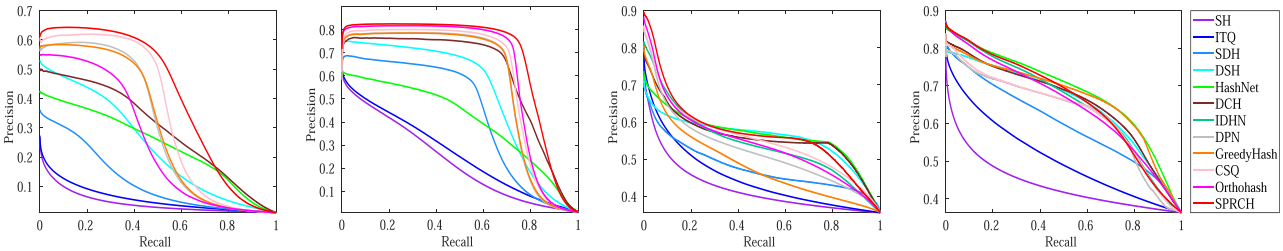
Loss	RCH loss					SPRCH loss				
Temperature	$\tau = 0.1$	$\tau = 0.2$	$\tau = 0.3$	$\tau = 0.4$	$\tau = 0.5$	$\tau = 0.1$	$\tau = 0.2$	$\tau = 0.3$	$\tau = 0.4$	$\tau = 0.5$
16 bits	6.8	42.7	**50.9**	48.9	44.6	6.2	43.0	**50.9**	50.7	49.2
32 bits	7.8	53.5	**60.6**	55.9	51.1	6.6	55.8	**61.4**	59.6	57.7
48 bits	8.1	56.3	**62.6**	57.6	49.7	7.0	59.6	**63.7**	62.4	59.4
64 bits	9.4	57.6	**62.9**	57.5	50.3	7.7	61.5	**64.2**	62.3	59.6

The best results are highlighted in bold.



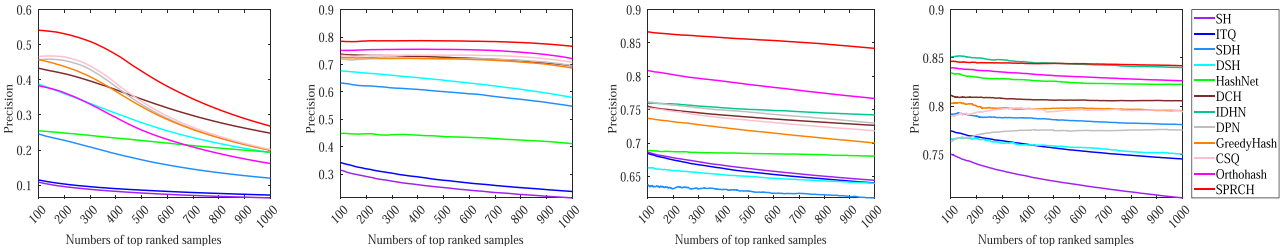
(a) CIFAR-100 (b) ImageNet-100 (c) COCO (d) NUS-WIDE

Fig. 6. PR curves of all hash methods with 16-bit hash codes on the CIFAR-100, ImageNet-100, COCO and NUS-WIDE datasets.



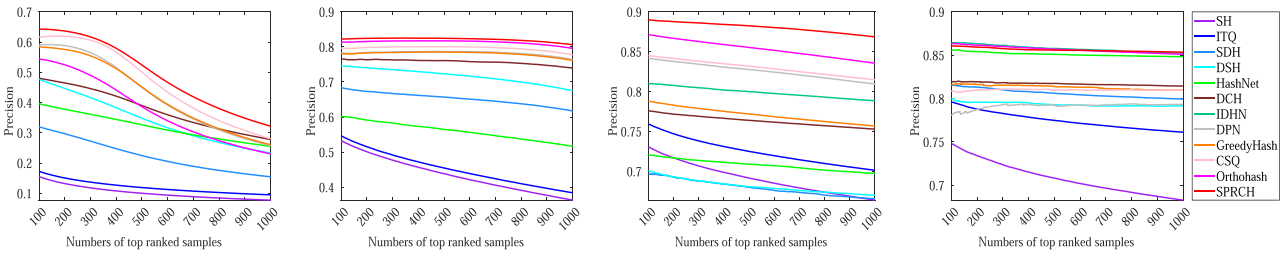
(a) CIFAR-100 (b) ImageNet-100 (c) COCO (d) NUS-WIDE

Fig. 7. PR curves of all hash methods with 32-bit hash codes on the CIFAR-100, ImageNet-100, COCO and NUS-WIDE datasets.



(a) CIFAR-100 (b) ImageNet-100 (c) COCO (d) NUS-WIDE

Fig. 8. P@N curves of all hash methods with 16-bit hash codes on the CIFAR-100, ImageNet-100, COCO and NUS-WIDE datasets.



(a) CIFAR-100 (b) ImageNet-100 (c) COCO (d) NUS-WIDE

Fig. 9. P@N curves of all hash methods with 32-bit hash codes on the CIFAR-100, ImageNet-100, COCO and NUS-WIDE datasets.

TABLE IX
COMPARED RESULTS IN TERMS OF MAP (%) OF THE PROPOSED SPRCH LOSS USING DIFFERENT BATCH SIZES ON THE CIFAR-100 AND COCO DATASETS

Batch size	CIFAR-100 mAP@1K				COCO mAP@5K			
	16 bits	32 bits	48 bits	64 bits	16 bits	32 bits	48 bits	64 bits
32	**53.9**	**62.9**	**64.9**	**64.5**	81.9	84.7	85.5	85.6
64	52.2	62.9	64.4	64.3	82.9	85.4	86.3	86.5
128	50.9	61.4	63.7	64.2	83.0	85.6	**86.4**	**86.8**
256	48.1	58.7	61.5	62.3	**83.2**	**85.7**	86.3	86.6
512	44.8	54.9	57.6	59.0	82.6	85.3	85.9	86.3

The best results are highlighted in bold.

V. CONCLUSION

In this article, we propose a novel Self-Paced Relational Contrastive Hashing (SPRCH) method for the image retireval task. Considering the collaboration of the point-to-point and point-to-class relations, the proposed SPRCH method explores the above two kinds of relations using contrastive learning at the same time. In order to reduce the drastic imbalance between the pairs with the point-to-point and point-to-class relations, weighting factors are added to rebalance their weights. Besides, a self-paced learning schedule is introduced to give hard pairs higher weights, which makes deep hash model learn valuable information from hard pairs gradually. Experimental results confirm that our proposed method outperforms the state-of-the-art supervised deep hash methods on four large-scale image retrieval datasets.

REFERENCES

[1] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[2] Z. Li, H. Tang, Z. Peng, G.-J. Qi, and J. Tang, "Knowledge-guided semantic transfer network for few-shot image recognition," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Feb. 06, 2023, doi: 10.1109/TNNLS.2023.3240195.

[3] J. Tang, Z. Li, M. Wang, and R. Zhao, "Neighborhood discriminant hashing for large-scale image retrieval," *IEEE Trans. Image Process.*, vol. 24, no. 9, pp. 2827–2840, Sep. 2015.

[4] J. Gui, T. Liu, Z. Sun, D. Tao, and T. Tan, "Supervised discrete hashing with relaxation," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 3, pp. 608–617, Mar. 2018.

[5] Z. Li and J. Tang, "Weakly supervised deep metric learning for community-contributed image retrieval," *IEEE Trans. Multimedia*, vol. 17, no. 11, pp. 1989–1999, Nov. 2015.

[6] L. Zhu, Z. Huang, Z. Li, L. Xie, and H. T. Shen, "Exploring auxiliary context: Discrete semantic transfer hashing for scalable image retrieval," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 11, pp. 5264–5276, Nov. 2018.

[7] E. Yang, T. Liu, C. Deng, W. Liu, and D. Tao, "DistillHash: Unsupervised deep hashing by distilling data pairs," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 2946–2955.

[8] Z. Li, J. Tang, and T. Mei, "Deep collaborative embedding for social image understanding," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 9, pp. 2070–2083, Sep. 2019.

[9] X. Lu, L. Zhu, J. Li, H. Zhang, and H. T. Shen, "Efficient supervised discrete multi-view hashing for large-scale multimedia search," *IEEE Trans. Multimedia*, vol. 22, no. 8, pp. 2048–2060, Aug. 2020.

[10] C. Deng, E. Yang, T. Liu, and D. Tao, "Two-stream deep hashing with class-specific centers for supervised image search," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 6, pp. 2189–2201, Jun. 2020.

[11] Y. Li, W. Liu, and J. Huang, "Sub-selective quantization for learning binary codes in large-scale image search," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 6, pp. 1526–1532, Jun. 2018.

[12] Z. Wang, Z. Zhang, Y. Luo, Z. Huang, and H. T. Shen, "Deep collaborative discrete hashing with semantic-invariant structure construction," *IEEE Trans. Multimedia*, vol. 23, pp. 1274–1286, 2020.

[13] L. Yuan et al., "Central similarity quantization for efficient image and video retrieval," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 3083–3092.

[14] Z. Li, J. Tang, L. Zhang, and J. Yang, "Weakly-supervised semantic guided hashing for social image retrieval," *Int. J. Comput. Vis.*, vol. 128, pp. 2265–2278, 2020.

[15] D. Zhai et al., "Supervised distributed hashing for large-scale multimedia retrieval," *IEEE Trans. Multimedia*, vol. 20, no. 3, pp. 675–686, Mar. 2018.

[16] X. Xiang, Y. Zhang, L. Jin, Z. Li, and J. Tang, "Sub-region localized hashing for fine-grained image retrieval," *IEEE Trans. Image Process.*, vol. 31, pp. 314–326, 2021.

[17] H. Liu, R. Wang, S. Shan, and X. Chen, "Deep supervised hashing for fast image retrieval," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 2064–2072.

[18] L. Jin, K. Li, H. Hu, G.-J. Qi, and J. Tang, "Semantic neighbor graph hashing for multimodal retrieval," *IEEE Trans. Image Process.*, vol. 27, no. 3, pp. 1405–1417, Mar. 2018.

[19] Z. Cao, M. Long, J. Wang, and P. S. Yu, "HashNet: Deep learning to hash by continuation," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 5608–5617.

[20] C. Li, S. Gao, C. Deng, D. Xie, and W. Liu, "Cross-modal learning with adversarial samples," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, Art. no. 968.

[21] Y. Cao, M. Long, B. Liu, and J. Wang, "Deep cauchy hashing for hamming space retrieval," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 1229–1237.

[22] X. Ma, T. Zhang, and C. Xu, "Multi-level correlation adversarial hashing for cross-modal retrieval," *IEEE Trans. Multimedia*, vol. 22, no. 12, pp. 3101–3114, Dec. 2020.

[23] L. Jin et al., "Deep semantic-preserving ordinal hashing for cross-modal similarity search," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 5, pp. 1429–1440, May 2019.

[24] Z. Zhang, Q. Zou, Y. Lin, L. Chen, and S. Wang, "Improved deep hashing with soft pairwise similarity for multi-label image retrieval," *IEEE Trans. Multimedia*, vol. 22, no. 2, pp. 540–553, Feb. 2020.

[25] H. Lai, Y. Pan, Y. Liu, and S. Yan, "Simultaneous feature learning and hash coding with deep neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 3270–3278.

[26] J. Tang and Z. Li, "Weakly supervised multimodal hashing for scalable social image retrieval," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 28, no. 10, pp. 2730–2741, Oct. 2018.

[27] R. Zhang, L. Lin, R. Zhang, W. Zuo, and L. Zhang, "Bit-scalable deep hashing with regularized similarity learning for image retrieval and person re-identification," *IEEE Trans. Image Process.*, vol. 24, no. 12, pp. 4766–4779, Dec. 2015.

[28] S. Su, C. Zhang, K. Han, and Y. Tian, "Greedy hash: Towards fast optimization for accurate hash coding in CNN," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 806–815.

[29] L. Jin et al., "Deep ordinal hashing with spatial attention," *IEEE Trans. Image Process.*, vol. 28, no. 5, pp. 2173–2186, May 2019.

[30] X. Lu, L. Liu, L. Nie, X. Chang, and H. Zhang, "Semantic-driven interpretable deep multi-modal hashing for large-scale multimedia retrieval," *IEEE Trans. Multimedia*, vol. 23, pp. 4541–4554, 2020.

[31] L. Fan, K. W. Ng, C. Ju, T. Zhang, and C. S. Chan, "Deep polarized network for supervised learning of accurate binary hashing codes," in *Proc. Int. Joint Conf. Artif. Intell.*, 2020, pp. 825–831.

[32] C. Li et al., "Self-supervised adversarial hashing networks for cross-modal retrieval," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 4242–4251.

[33] J. T. Hoe et al., "One loss for all: Deep hashing with a single cosine similarity based learning objective," in *Proc. Adv. Neural Inf. Process. Syst.*, 2021, pp. 24286–24298.

[34] J. Tang, J. Lin, Z. Li, and J. Yang, "Discriminative deep quantization hashing for face image retrieval," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 12, pp. 6154–6162, Dec. 2018.

[35] K. D. Doan, P. Yang, and P. Li, "One loss for quantization: Deep hashing with discrete Wasserstein distributional matching," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 9447–9457.

[36] Y. Xie et al., "Label-affinity self-adaptive central similarity hashing for image retrieval," *IEEE Trans. Multimedia*, early access, Feb. 23, 2023, doi: 10.1109/TMM.2023.3248170.
[37] J. Wang, L. Jin, Z. Li, and J. Tang, "Crossmodal knowledge distillation hashing (in chinese)," *Sci. Sin. Tech.*, vol. 52, no. 5, pp. 713–726, 2022.
[38] Y. Weiss, A. Torralba, and R. Fergus, "Spectral hashing," in *Proc. Adv. Neural Inf. Process. Syst.*, 2008, pp. 1753–1760.
[39] B. Kulis and K. Grauman, "Kernelized locality-sensitive hashing for scalable image search," in *Proc. IEEE 12th Int. Conf. Comput. Vis.*, 2009, pp. 2130–2137.
[40] M. Norouzi and D. M. Blei, "Minimal loss hashing for compact binary codes," in *Proc. Int. Conf. Mach. Learn.*, 2011, pp. 353–360.
[41] Y. Gong, S. Lazebnik, A. Gordo, and F. Perronnin, "Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 12, pp. 2916–2929, Dec. 2013.
[42] M. Norouzi, D. J. Fleet, and R. R. Salakhutdinov, "Hamming distance metric learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1061–1069.
[43] F. Shen, C. Shen, W. Liu, and H. Tao Shen, "Supervised discrete hashing," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 37–45.
[44] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 1597–1607.
[45] T. Chen, S. Kornblith, K. Swersky, M. Norouzi, and G. E. Hinton, "Big self-supervised models are strong semi-supervised learners," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 22243–22255.
[46] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 9729–9738.
[47] X. Chen, H. Fan, R. Girshick, and K. He, "Improved baselines with momentum contrastive learning," 2020, *arXiv:2003.04297*.
[48] X. Chen, S. Xie, and K. He, "An empirical study of training self-supervised vision transformers," in *Proc. Int. Conf. Comput. Vis.*, 2021, pp. 9640–9649.
[49] M. Caron et al., "Unsupervised learning of visual features by contrasting cluster assignments," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 9912–9924.
[50] J.-B. Grill et al., "Bootstrap your own latent–A new approach to self-supervised learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 21271–21284.
[51] J. Zbontar, L. Jing, I. Misra, Y. LeCun, and S. Deny, "Barlow twins: Self-supervised learning via redundancy reduction," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 12310–12320.
[52] P. Khosla et al., "Supervised contrastive learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 18661–18673.
[53] M. Kang and J. Park, "ContraGAN: Contrastive learning for conditional image generation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 21357–21369.
[54] G. Mikriukov, M. Ravanbakhsh, and B. Demir, "Deep unsupervised contrastive hashing for large-scale cross-modal text-image retrieval in remote sensing," in *IEEE Int. Conf. Acoustics, Speech Signal Process.*, 2022, pp. 4463–4467.
[55] P. Hu et al., "Unsupervised contrastive cross-modal hashing," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 3, pp. 3877–3889, Mar. 2023.
[56] Z. Qiu, Q. Su, Z. Ou, J. Yu, and C. Chen, "Unsupervised hashing with contrastive information bottleneck," in *Proc. Int. Joint Conf. Artif. Intell.*, 2021, pp. 959–965.
[57] J. Wang, Z. Zeng, B. Chen, T. Dai, and S.-T. Xia, "Contrastive quantization with code memory for unsupervised image retrieval," in *Proc. AAAI Conf. Artif. Intell.*, 2022, pp. 2468–2476.
[58] M. Kumar, B. Packer, and D. Koller, "Self-paced learning for latent variable models," in *Proc. Adv. Neural Inf. Process. Syst.*, 2010, pp. 1189–1197.
[59] L. Jiang et al., "Self-paced learning with diversity," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2078–2086.
[60] L. Lin, K. Wang, D. Meng, W. Zuo, and L. Zhang, "Active self-paced learning for cost-effective and progressive face identification," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 1, pp. 7–19, Jan. 2018.
[61] S. Guo et al., "CurriculumNet: Weakly supervised learning from large-scale web images," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 135–150.
[62] L. Jiang, Z. Zhou, T. Leung, L.-J. Li, and L. Fei-Fei, "MentorNet: Learning data-driven curriculum for very deep neural networks on corrupted labels," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 2304–2313.
[63] D. Zhang, J. Han, L. Zhao, and D. Meng, "Leveraging prior-knowledge for weakly supervised object detection under a collaborative self-paced curriculum learning framework," *Int. J. Comput. Vis.*, vol. 127, no. 4, pp. 363–380, 2019.
[64] S. Jin et al., "SSAH: Semi-supervised adversarial deep hashing with self-paced hard sample generation," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 11157–11164.
[65] A. Krizhevsky et al., "Learning multiple layers of features from tiny images," Univ. Toronto, Toronto, Canada, Tech. Rep., 2009.
[66] O. Russakovsky et al., "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, 2015.
[67] T.-Y. Lin et al., "Microsoft COCO: Common objects in context," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 740–755.
[68] T.-S. Chua et al., "Nus-wide: A real-world web image database from national university of Singapore," in *Proc. ACM Int. Conf. Image Video Retrieval*, 2009, pp. 1–9.

Zhengyun Lu received the B.E. degree in electronic and information engineering in 2017 from the Nanjing University of Science and Technology, Nanjing, China, where he is currently working toward the Ph.D. degree in computer science and technology. His research interests include multimedia computing, deep learning, and image retrieval.

Lu Jin received the Ph.D. degree from the Nanjing University of Science and Technology, Nanjing, China, in 2019. She is currently a Associate Professor with the Nanjing University of Science and Technology. Her research interests include computer vision and multimedia retrieval.

Zechao Li (Senior Member, IEEE) received the B.E. degree from the University of Science and Technology of China, Hefei, China, in 2008, and the Ph.D. degree from the National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing, China, in 2013 . He is currently a Professor with the Nanjing University of Science and Technology, Nanjing, China. He has authored more than 70 papers in the top-tier journals and conferences. His research interests include largescale multimedia analysis, computer vision, and pattern recognition. He was the recipient of the best paper award in ACM Multimedia Asia 2020, and best student paper award in ICIMCS 2018. He is an Associate Editor for the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS and *Information Sciences*.

Jinhui Tang (Senior Member, IEEE) received the B.E. and Ph.D. degrees from the University of Science and Technology of China, Hefei, China, in 2003 and 2008, respectively. He is currently a Professor with the Nanjing University of Science and Technology, Nanjing, China. He has authored more than 200 articles in top tier journals and conferences. His research interests include multimedia analysis and computer vision. Dr.Tang was the recipient of the Best Paper Awards in ACM MM 2007 and ACM MM Asia 2020, Best Paper Runner-Up in ACM MM 2015. He was an Associate Editor for IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, IEEE TRANSACTIONS ON MULTIMEDIA, and IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY. He is a Fellow of IAPR.