Multimedia Tools and Applications (2021) 80:16215–16227
https://doi.org/10.1007/s11042-020-08799-5

Supervised discrete hashing through similarity learning

Hong Wang[1] · Xingbo Liu[2] · Xiushan Nie[3]

Received: 4 April 2019 / Revised: 29 November 2019 / Accepted: 28 February 2020 /
Published online: 11 March 2020

Abstract
Supervised hashing has achieved better accuracy than unsupervised hashing in many practical applications owing to its use of semantic label information. However, the mutual relationship between semantic labels is always ignored when leveraging label information. In addition, the major challenge in learning hash is of handling the discrete constraints imposed on the hash codes, which typically transform the hash optimization into NP-hard problems. To address these issues, a form of supervised discrete hashing through learning mutual similarities is proposed. Different from the existing supervised hashing methods that learn hash codes from least-squares classification by regressing the hash codes to their corresponding labels, we leverage the mutual relation between different semantic labels to learn more stable hash codes. In addition, the proposed method can simultaneously learn the discrete hash codes for training samples and the projections between the original features and their corresponding hash codes for the out-of-sample cases. Experiments have beeen performed on two public datasets. The experimental results demonstrate the superiority of the proposed method.

Keywords Supervised hashing · Discrete hashing · Similarity learning · Mutual relationship

1 Introduction

With the explosive growth in data usage, the approximate nearest neighbor (ANN) search in large databases is becoming increasingly key and is attracting considerable research attention in lots of fields, including information retrieval, computer vision and data mining. The ANN attempts to find an approximate nearest neighbor for a query point in a large database; hashing is one of the primary technologies in the ANN, as it can achieve better performance than other methods in the ANN search applications [3, 17, 22, 31, 33].

✉ Xiushan Nie
niexiushan@163.com

[1] School of Journalism and Communication, Shandong Normal University, Jinan, 250014, China

[2] School of Computer Science and Technology, Shandong University, Jinan, 250014, China

[3] School of Computer Science and Technology, Shandong Jianzhu University, Jinan, 250014, China

Hashing consists of attempting to convert medias, such as images and videos, to a set of short binary codes, and the binary code, which can be called hash code, should preserve the semantic similarity among the data in the original space. With these binary hash codes, users can readily conduct the task of the ANN on a large-scale dataset because of the high efficiency of the pairwise or triplet-wise comparison with binary codes in the Hamming distance [36]. There are two primary categories in the existing hashing techniques, which are data-independent methods [4, 12, 13] and data-dependent [32, 34, 55, 59] methods. In the first category, the hashing methods use no data for training. Data-independent hashing methods use random projections to extract the feature representation. Locality sensitive hashing (LSH) [12] and the variants of LSH [4, 13] are the best-known data-independent algorithms. The main idea behind the LSH family of methods is to return the same bit for the neighborhoods in the original space with a high probability based on a hashing function. LSH has exhibited interesting theoretical properties and a performance guarantee. However, LSH-based methods have a major limitation, as they typically require longer hash codes to achieve a better retrieval precision performance, thus reducing retrieval recall. Multiple hashing table-based methods can alleviate this problem partially, but they inevitably lead to increased storage cost and retrieval time.

The second category, called data-dependent hashing, uses data to train the model. Data-dependent hashing, also known as learning-to-hash or learned-based hashing, attempts to learn the hash functions so that the nearest neighbors in the Hamming space approximate those in the original space [48, 58]. The analysis of the underlying characteristics of data can lead to better retrieval performance. As a result, data-dependent hashing has become more and more popular, as the learned compact hash codes can effectively and efficiently search massive amounts of data. Generally, there are two main types of data-dependent hashing methods: unsupervised and supervised.

Unsupervised hashing methods employ unlabeled data in the training process, and the hash functions are learned mainly from the original data structure, and does not make any use of the label information. This type of hashing includes spectral hashing [47], principal component hashing [30], principal component analysis [11], iterative quantization (PCA-ITQ) [6], scalable graph hashing with feature transformation [14], and inductive manifold hashing (IMH) [38]. However, as the label information of the input data does not be considered in unsupervised hashing methods, certain useful information, critical to pattern classification, may be lost. Therefore, various supervised hashing methods have been proposed based on their enhanced discriminative recognition abilities. The K-means hashing (KMH) [9] was proposed to generate hash codes through K-means clustering and quantization simultaneously. Anchor graph hashing (AGH) [26] was proposed to estimate data similarity through anchors. Recently, with the rapid advances in deep learning, a CNN has achieved a significant breakthrough in the visual area. Deep Hashing (DH) in [28] is a deep unsupervised framework, and aims to learn hierarchical non-linear transformations and minimize the loss between a compact real-valued vector and the learned binary code. Similarity-Adaptive Discrete Hashing (SADH) [39] proposed an unsupervised architecture as an alternative approach to deep model training, similarity updating and code optimization.

In contrast with unsupervised methods, supervised hashing methods fully utilize class labels [27]. For example, in [44], Wang et al. proposed a semi-supervised hashing (SSH) framework, and in this framework they minimize empirical errors over the labeled set. Liu et al. [25] proposed kernel-based supervised hashing (KSH), and Lin et al. [19] proposed fast supervised hashing using graph cuts and decision trees. In [6], ITQ was extended to CCA-ITQ, which uses label information to perform CCA and maximize the correlation of

samples from the same class first, and then performs ITQ to minimize the quantization loss. LDA Hashing [41] was proposed to minimize the variation of data within the same classes and maximize the variation of data across different classes. Supervised discrete hashing (SDH) [36] was introduced to learn hash codes using a regularization algorithm and yield an analytic solution for the regularization sub-problem. As for deep hashing models, in [23], a novel supervised deep hashing model (DSH) was proposed to perform simultaneous feature representation learning and hash code learning using pairwise labels. In [16] and [35], asymmetric deep supervised hashing (ADSH) and deep asymmetric pairwise hashing (DAPH) were proposed to learn hash mappings in an asymmetric deep architecture during the training process. Certain ranking-based methods, such as the ones described in [46] and [45], are also considered supervised methods [7].

As is known, hash codes are binary, and discrete constraints always lead to the mixed-integer optimization problem which is NP-hard [24]. To address this issue, the hashing methods first relax the discrete optimization problem by discarding the discrete constraints. A quantization step can then be performed, turning the real numbers into the approximate hash code using thresholding, and we call this process relaxation, which can simplify the original discrete optimization considerably. However, this type of approximation is suboptimal, leading to a low quality, and often producing a less effective hash code as a result of accumulated quantization errors. To overcome this issue, Lin et al. [20] tried to solve the discrete optimization problem directly. However, their method is typically time-consuming and unscalable. Shen et al. [37] proposed a discrete hashing method using a cyclic coordinate descent. Kang et al. [17] proposed a column sampling-based discrete supervised hashing method(COSDISH) which first learns the binary code and then trains binary classifiers, with each bit corresponding to one classifier. Generally, in these approaches, the learning of the hash is formulated in terms of least squares classification regressing each hash code to its corresponding label. This strategy does not fully leverage the mutual relations between labels. In addition, despite considering this relation in some methods, such as COSDISH, they are still prone to dividing the hash learning into two steps. To address the issues mentioned above, we propose in this study a supervised discrete hashing method with similarity learning (SDHSL). SDHSL not only fully leverages the mutual relations across semantic labels, but also directly learns the discrete hash code and projection for out-of-sample extensions simultaneously. The main contributions of this study are summarized as follows:

- *Learning hash codes from the relative similarity between semantic labels*. The primary purpose of hashing is the search of a close neighbor. Hence the relative similarity between semantic labels is quite evidently crucial for learning. In contrast with the existing supervised methods that always consider hash codes as features of samples and then regress them to labels, the proposed method learns hash codes from label similarity. The objective function can then be solved by a 2-approximation algorithm. Existing works have proven that the 2-approximation algorithm is more stable as the category label is provided [17, 29, 52].
- *Directly learning discrete hash codes and the projection for out-of-samples simultaneously*. In the proposed SDHSL, we jointly learn the discrete hash codes for training samples and the projection that transforms the original sample into a hash code, thus leading the proposed method to be more efficient.
- Experimental results on some large-scale datasets illustrate that SDHSL can outperform the existing methods in the task of image retrieval.

The remainder of this paper is organized as follows. In Section 2, we describe the proposed SDHSL method in detail, and in Section 3, we evaluate it through experiments. Section 4 consists of the conclusion of our study.

2 Proposed Method

This section describes the proposed SDHSL method in detail. We first present the SDHSL formulation and then describe the process of optimization.

2.1 Problem Formulation

Assume that we have a training set $\mathbf{X}$ consisting of n instances (i.e., $\mathbf{X} = (\mathbf{x}_i)_{i=1}^n$), where $\mathbf{x}_i \in R^f$ is the feature vector. The semantic label matrix $\mathbf{Y} = \{\mathbf{y}_i\}_{i=1}^n$ is also available, with $\mathbf{y}_i = \{y_{ij}\} \in R^c$ being the label vector of the i_{th} instance and c the number of categories in the training set. If the i_{th} instance belongs to the j_{th} category, we have $y_{ij} = 1$; otherwise, 0. In addition, the hash matrix is defined as $\mathbf{H} = \{\mathbf{h}_i\}_{i=1}^n$. Some notations and definitions are shown in Table 1.

Learning a binary code matrix $\mathbf{H} \in \{-1, 1\}^{n \times L}$ using semantic labels is the purpose of supervised hashing, with L being the length of the hash code. Most supervised hashing methods effectively utilize the label information by minimizing the term $\left\|\mathbf{Y} - \mathbf{W}^T\mathbf{H}\right\|^2$, which is accomplished through the methods in [36, 42, 50]. However, this strategy leads to additional time spent updating the auxiliary variable $\mathbf{W}$ during training, and the linear projection $\mathbf{W}$ is a weak metric in terms of bridging the gap between hash codes and semantic labels. In addition, the linear regression used in $\left\|\mathbf{Y} - \mathbf{W}^T\mathbf{H}\right\|^2$ is less stable for generating discrete hash codes [8, 10].

By contrast, semantic similarity based on labels is also used for learning hash codes in some supervised hashing methods. We assume that $\{\mathbf{S} = S_{ij} | S_{ij} \in \{-1, 1\}^{n \times n}\}$ is a fully observed semantic similarity matrix with no missing entries, where $S_{ij} = 1$ indicates that the $i - th$ and j_{th} samples are semantically similar (i.e., they share the same label), and $S_{ij} = -1$ means that the $i - th$ and j_{th} samples are semantically dissimilar (i.e., they have different labels). The hash codes in the Hamming space should preserve the similarity

Table 1 Notations

Notation	Description
X	Training set
S	Pairwise-based similarity matrix
P	Projection matrix from feature to hash code
W	Projection matrix from hash code matrix to label matrix
Y	Label matrix
H	Hash code matrix
x_i	The i_{th} sample
h_i	Hash code of the i_{th} sample
n	Number of training samples
L	Length of hash code

between the samples in the original space. Therefore, we can formulate this as the following minimization problem:

$$\min_{\mathbf{H}} \left\| \mathbf{H}^T\mathbf{H} - L \cdot \mathbf{S} \right\|^2 \qquad \text{s.t.} \quad \mathbf{H} \in \{-1, 1\}^{L\times n}, \tag{1}$$

where L is the length of the hash codes and $\|\cdot\|$ is the ℓ_2 norm.

In general, S_{ij} is an element of the similarity matrix and is usually 0 or 1 in traditional methods, based on the similarity or dissimilarity of the labels for samples x_i and x_j, respectively. However, in the proposed method, we adopt $\{-1, 1\}$ rather than $\{0, 1\}$ for S_{ij} during hash learning. The reason as follows.

We are given two samples $\mathbf{x}_i$ and $\mathbf{x}_j$, with hash codes $\mathbf{h}_i$ and $\mathbf{h}_j$, respectively. The equation for learning hash codes can be formulated as follows:

$$\min_{\mathbf{h}_i, \mathbf{h}_j} \left\| \mathbf{h}_i^T\mathbf{h}_j - L \cdot S_{i,j} \right\|^2 \qquad \text{s.t.} \quad \mathbf{h}_i, \mathbf{h}_j \in \{-1, 1\}^{L\times 1}. \tag{2}$$

If $\mathbf{x}_i$ and $\mathbf{x}_j$ are dissimilar, and $S_{i,j} = 0$, the optimal solution for $\mathbf{h}_i$ and $\mathbf{h}_j$ is that $\left[\frac{L}{2}\right]$ bits of the hash codes are identical (*i.e.,* $L - \left[\frac{L}{2}\right]$ bits of the hash codes are different). Thus, the Hamming distance for $\mathbf{h}_i$ and $\mathbf{h}_j$ is $L - \left[\frac{L}{2}\right]$, which seems to be a waste of hash length. Even worse, if the optimal solution for $\mathbf{h}_i$ and $\mathbf{h}_j$ is difficult to obtain, there are two equivalent suboptimal solutions. Specifically, it is equivalent for problem $\left\|\mathbf{h}_i^T\mathbf{h}_j - L \cdot \mathbf{S}_{i,j}\right\|^2$ if $\mathbf{h}_i$ has $\left[\frac{L}{2}\right] + k$ or $\left[\frac{L}{2}\right] - k$ (k as an integer in $(0, L - \left[\frac{L}{2}\right])$) bits of hash codes which are the same as $\mathbf{h}_j$. The two equivalent suboptimal solutions destabilize the process for generating hash codes.

If $\mathbf{x}_i$ and $\mathbf{x}_j$ are dissimilar and $S_{i,j} = -1$, the optimal solution for $\mathbf{h}_i$ and $\mathbf{h}_j$ implies that L bits of hash codes are different. Thus, the Hamming distance for $\mathbf{h}_i$ and $\mathbf{h}_j$ is L, which makes full use of the hash length.

The problem defined in (1) means that the semantic similarity should be preserved by the binary codes in the Hamming space. This term has been used in certain supervised hashing methods, namely, [18, 20, 21, 49, 56], and [17]. However, most of these methods use a two-step learning strategy, which first learns the hash code and then trains a classifier for the out-of-sample generalization. In this study, we learn the hash codes and the projection between the samples and their hash codes together.

Learning-based hashing is known to typically involve learning a projection from the original samples to generate hash codes. In this study, we adopt a linear projection $\mathbf{P}$ to bridge the gap between the Hamming and original feature space. To learn robust hash codes, the $\ell_{2,p}$-norm ($0 < p \leq 2$) is adopted as its ability to alleviate sample noise has been proven [51, 53, 54]. Given a matrix $\mathbf{X} = \{\mathbf{x}_i\} \in R^{f\times n}$, the $\ell_{2,p}$-norm is defined as $||\mathbf{X}||_{2,p} = \sum_{i=1}^{n} ||\mathbf{x}_i||_2^p$. Compared with the ℓ_2-norm, the $\ell_{2,p}$-norm can suppress the influence of potential noise and expand the applicable range. Furthermore, the $\ell_{2,p}$-norm can also flexibly adapt to different levels of hash code noise.

To avoid a trivial solution for $\mathbf{P}$, we use an additional constraint by setting the diagonal entries $\mathbf{P}^T\mathbf{P}$ to $\mathbf{1}$. The projection learning term can be formulated as:

$$\begin{aligned} &\min_{\mathbf{H},\mathbf{P}} \left\| \mathbf{H} - \mathbf{P}^T\mathbf{X} \right\|_{2,p} \\ &\text{s.t.} \quad \mathbf{H} \in \{-1, 1\}^{L\times n}, diag(\mathbf{P}^T\mathbf{P}) = \mathbf{1}, 0 < p \leq 2, \end{aligned} \tag{3}$$

where $\mathbf{1}$ is a constant vector whose elements are 1.

In addition, based on the property of the hash codes, each hash bit has a 50% chance of being -1 or 1 in the hash vector, which is called the balance [15, 35]. In this study, the property balance can be formulated as:

$$\min_{\mathbf{H}} \|\mathbf{H1}\|^2 \qquad \text{s.t.} \quad \mathbf{H} \in \{-1, 1\}^{L\times n}, \tag{4}$$

Finally, the objective function for SDHSL can be defined as:

$$\begin{aligned} &\min_{\mathbf{H},\mathbf{P}} \left\|\mathbf{H}^T\mathbf{H} - L\cdot\mathbf{S}\right\|^2 + \alpha\left\|\mathbf{H} - \mathbf{P}^T\mathbf{X}\right\|_{2,p} + \beta\|\mathbf{H1}\|^2 \\ &\text{s.t.} \quad \mathbf{H} \in \{-1, 1\}^{L\times n}, diag(\mathbf{P}^T\mathbf{P}) = \mathbf{1}, 0 < p \le 2, \end{aligned} \tag{5}$$

where α and β are penalty parameters.

2.2 Optimization

Directly optimizing (5) is a challenge as it is both non-convex and non-smooth. However, obtaining a solution becomes easier when one considers one variable while keeping another fixed.

First, we rewrite (5) as follows:

$$\begin{aligned} &\min_{\mathbf{H},\mathbf{P}} \left\|\mathbf{H}^T\mathbf{H} - L\cdot\mathbf{S}\right\|^2 + \alpha\cdot Tr((\mathbf{H} - \mathbf{P}^T\mathbf{X})\mathbf{D}(\mathbf{H} - \mathbf{P}^T\mathbf{X})^T) + \beta\|\mathbf{H1}\|^2 \\ &\text{s.t.} \quad \mathbf{H} \in \{-1, 1\}^{L\times n}, diag(\mathbf{P}^T\mathbf{P}) = \mathbf{1}, 0 < p \le 2, \end{aligned} \tag{6}$$

where $Tr(\cdot)$ is the trace of "$\cdot$", and $\mathbf{D}$ is a diagonal matrix, and the i_{th} diagonal element of $\mathbf{D}$ is defined as:

$$D_{i,i} = \frac{1}{\frac{2}{p}\|\mathbf{r}_i\|_2^{2-p}}, \tag{7}$$

where $\mathbf{r}_i$ is the i_{th} column of matrix $(\mathbf{H} - \mathbf{P}^T\mathbf{X})$.

In other words, (6) can be solved using an iterative framework comprising the following two steps until convergence. We can directly obtain a discrete solution for the hash codes without relaxation:

Step 1: Learn $\mathbf{H}$ with $\mathbf{P}$ fixed. The problem in (6) is then reduced to:

$$\begin{aligned} &\min_{\mathbf{H}} \left\|\mathbf{H}^T\mathbf{H} - L\cdot\mathbf{S}\right\|^2 + \alpha\cdot Tr((\mathbf{H} - \mathbf{P}^T\mathbf{X})\mathbf{D}(\mathbf{H} - \mathbf{P}^T\mathbf{X})^T) + \beta\|\mathbf{H1}\|^2 \\ &\text{s.t.} \quad \mathbf{H} \in \{-1, 1\}^{L\times n}, 0 < p \le 2. \end{aligned} \tag{8}$$

Inspired by the recent advance in non-convex and non-smooth optimization [1, 2] and discrete proximal linearized minimization (DPLM) [40], the discrete solution for $\mathbf{H}$ can be solved through an iterative process as follows. The convergence about DPLM can also been seen in [40].

We first define the objective function in (8) as $L(\mathbf{H})$. Thus, (8) can be transformed into an unconstrained problem as follows:

$$\min_{\mathbf{H}} \delta(\mathbf{H}) + L(\mathbf{H}) \tag{9}$$

where $\delta(\mathbf{H})$ is defined as:

$$\delta(\mathbf{H}) = \begin{cases} 0, & \mathbf{H} \in \{-1, 1\}. \\ \infty, & \mathbf{H} \notin \{-1, 1\}. \end{cases} \tag{10}$$

Using the proximal and forward-split algorithms [1] as inspiration, (9) can be formulated as:

$$\mathbf{H}^{j+1} = \arg\min_{\mathbf{H}} \delta(\mathbf{H}) + \frac{\lambda}{2}\left\|\mathbf{H} - \mathbf{H}^{j} + \frac{1}{\lambda}\bigtriangledown L(\mathbf{H}^{j})\right\| \tag{11}$$

Equation (11) can then be transformed into a constraint problem as follows:

$$\begin{aligned} &\min_{\mathbf{H}} \left\|\mathbf{H} - \mathbf{H}^{j} + \frac{1}{\lambda}\bigtriangledown L(\mathbf{H}^{j})\right\| \\ &\text{s.t.} \quad \mathbf{H} \in \{-1, 1\}^{L\times n}, 0 < p \leq 2. \end{aligned} \tag{12}$$

Now, determining that $\mathbf{H}$ has an optimal discrete solution is straightforward:

$$\mathbf{H}^{j+1} = sgn(\mathbf{H}^{j} - \frac{1}{\lambda}\bigtriangledown L(\mathbf{H}^{j})), \tag{13}$$

where $\mathbf{H}^{j}$ is the solution of $\mathbf{H}$ obtained after the j_{th} iteration, λ is a parameter, and $\bigtriangledown L(\mathbf{H})$ is defined as:

$$\bigtriangledown L(\mathbf{H}) = 4\mathbf{H}\mathbf{H}^{T}\mathbf{H} - 4L\mathbf{H}\mathbf{S} + 2(\mathbf{H} - \mathbf{P}^{T}\mathbf{X})\mathbf{D} + 2\beta\mathbf{H}\mathbf{1}\mathbf{1}^{T}. \tag{14}$$

Step 2: Learn $\mathbf{P}$ with $\mathbf{H}$ fixed. The problem in (6) is reduced to:

$$\begin{aligned} &\min_{\mathbf{P}} Tr((\mathbf{H} - \mathbf{P}^{T}\mathbf{X})^{T}\mathbf{D}(\mathbf{H} - \mathbf{P}^{T}\mathbf{X})) \\ &\text{s.t.} \quad diag(\mathbf{P}^{T}\mathbf{P}) = \mathbf{1}, 0 < p \leq 2. \end{aligned} \tag{15}$$

Setting the derivative of the objective function in (15) with $\mathbf{P}$ to zero:

$$\mathbf{X}\mathbf{X}^{T}\mathbf{P}\mathbf{D} - \mathbf{X}\mathbf{H}^{T}\mathbf{D} = 0, \tag{16}$$

the closed-form solution for $\mathbf{P}$ can be calculated as:

$$\mathbf{P} = (\mathbf{X}\mathbf{X}^{T})^{-1}\mathbf{X}\mathbf{H}^{T}. \tag{17}$$

To satisfy the constraint $diag(\mathbf{P}^{T}\mathbf{P}) = \mathbf{1}$, we project each row of $\mathbf{P}$ onto the unit norm ball after each update:

$$\mathbf{P}_{j} \leftarrow \frac{\mathbf{P}_{j}}{\|\mathbf{P}_{j}\|}, \tag{18}$$

where $\mathbf{P}_{j}$ is the j_{th} row of $\mathbf{P}$.

In general, the proposed discrete problem is solved based on DPLM [40]. In fact, the proposed discrete problem is reformulated as minimizing the sum of a smooth loss term with a non-smooth indicator function, and then it can be efficiently solved by an iterative procedure with each iteration admitting an analytical discrete solution, which can be converge very fast. The convergence can be achieved within a few iterations.

2.3 Time complexity

Let the symbols n, f and L represent the total number of training samples, the dimension of the sample feature and the length of the hash code, respectively. The time complexity for

learning the hash code $\mathbf{H}$ is $O(nL^2 + n^2L + nfL)$, and the time complexity for learning the projection $\mathbf{P}$ is $O(nf^2 + nfL + nL)$. As L is much smaller than n and f, the total training time complexity of the proposed method can be reduced to $O(n^2L + nf^2 + nfL)$.

3 Experiments

In this section, the datasets and evaluation metrics used in our experiments are described and additional implementation details are provided. Two image datasets CIFAR-10 NUS-WIDE were used for performance evaluation of the proposed SDHSL, and the performance comparison with several state-of-the-art methods was also presented. Our experiments were conducted on an Intel(R) Core(TM) i7-4790 CPU with 16 GB of RAM. The hyperparameters we set are listed in the section on implementation details.

3.1 Experimental Settings

3.1.1 Datasets

We used the two datasets of CIFAR-10 NUS-WIDE, which are widely used for image retrieval.

CIFAR-10: There are 60,000 images in this database, 50,000 for training and 10,000 for testing. Ten classes are included in the image dataset with 6,000 images in every class.

NUS-WIDE: There are 269648 images with 81 classes in this database. We chose the 10 classes that contained the most images. Hence, the total number of images used in the experiment is 67994. The bag-of-visual-words SIFT feature with 500-dimension is used for each image as the input feature vector. In addition, 20000 images are chosen as the training set and 500 images as the test set.

3.1.2 Evaluation Metric

To evaluate the proposed hashing method, we used an evaluation metric commonly used in image retrieval, viz. the mean average precision (MAP). MAP is the average of the average precision (AP) values of the top retrieved samples.

$$\mathrm{MAP} = \frac{1}{Q}\sum_{r=1}^{Q}\mathrm{AP}(i), \tag{19}$$

where Q is the number of query images, and $\mathrm{AP}(i)$ is the AP of the i_{th} instance. AP is defined as:

$$\mathrm{AP} = \frac{1}{R}\sum_{r=1}^{G} precision(r)\sigma(r), \tag{20}$$

where R is the number of relevant samples in the G samples retrieved. Here, $\sigma(r) = 1$ if the r_{th} sample is relevant to the query; otherwise, $\sigma(r) = 0$.

3.1.3 Implementation Details.

We compared our method with the following: SPLH [43], KSH [25], LFH [57], FastH [20], ITQ [5], SDH [36], and COSDISH [17]. The hyperparameters were all initialized based on the authors' suggestions. For each method, we conducted five experiments and provided the average result of these five experiments. During training, four hyperparameters were used with our proposed SDHSL method:, α, β, p and λ. We empirically set α to 0.01, β to 10, p to 1.2 and λ to 0.1.

3.2 Experimental Results and Analysis

3.2.1 Performance Evaluation

Table 2 shows the MAP of the proposed SDHSL method and baselines. Compared with the baseline methods, we can see that our method outperforms the baselines on both CIFAR-10 and NUS-WIDE datasets in most cases.

From Table 2, it appears that SRDML achieves the best results on the under most of cases. In particular, the proposed SDHSL method achieves an improvement of over 20% in some cases over the other methods. As the length of the hash code increases, our method performs even better. On the CIFAR-10 dataset, the MAP performance is better than most of the baseline methods, although our performance trails that of COSDISH when the hash code length is 8 and 64. The COSDISH also considers label similarity during hash learning. However, it divides the generation of the hash code of the training samples and the hash function learning into two steps; these are learned simultaneously in the proposed method, and the strategy used in SDHSL is more efficient.

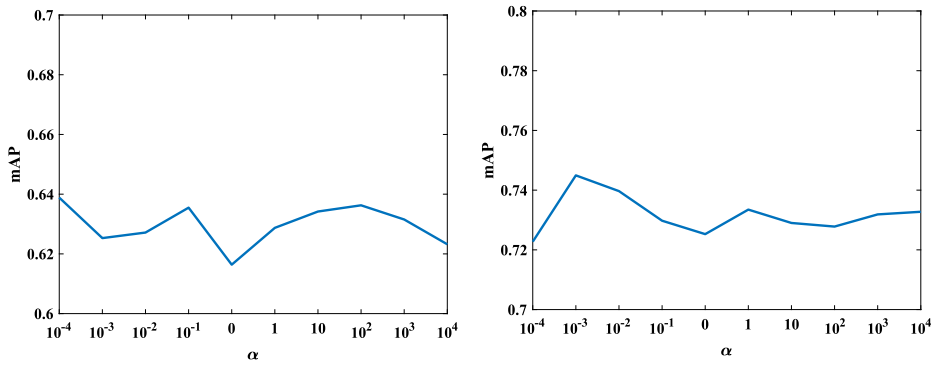
3.2.2 Parameter analysis

In this study, we learn the hash codes for the training samples and the projection between the original features and their corresponding hash codes for out-of-sample extensions simultaneously. Therefore, the parameter α balancing these two terms in the objective function is important. In addition, the $\ell_{2,p}$-norm is adopted in the proposed method, potentially leading

Table 2 Performance in terms of MAP

Method	CIFAR-10				NUS-WIDE			
	8 bits	16 bits	32 bits	64 bits	8 bits	16 bits	32 bits	64 bits
SPLH	0.1588	0.1635	0.1701	0.1730	0.3769	0.4077	0.4147	0.4071
KSH	0.2334	0.2662	0.2923	0.3128	0.4275	0.4546	0.4645	0.4688
TSH	0.2365	0.3080	0.3455	0.3663	0.4593	0.4784	0.4857	0.4955
LFH	0.2908	0.4098	0.5446	0.6182	0.5437	0.5929	0.6025	0.6136
SDH	0.2642	0.3994	0.4145	0.4346	0.4739	0.4674	0.4908	0.4944
FastH	0.4230	0.5216	0.5970	0.6446	0.5014	0.5296	0.5541	0.5736
COSDISH	0.4986	0.5768	0.6191	0.6371	0.5454	0.5940	0.6218	0.6329
SDHSL	0.4126	**0.6243**	**0.6242**	0.6237	**0.7039**	**0.7226**	**0.7296**	**0.7237**

The best results are shown in bold



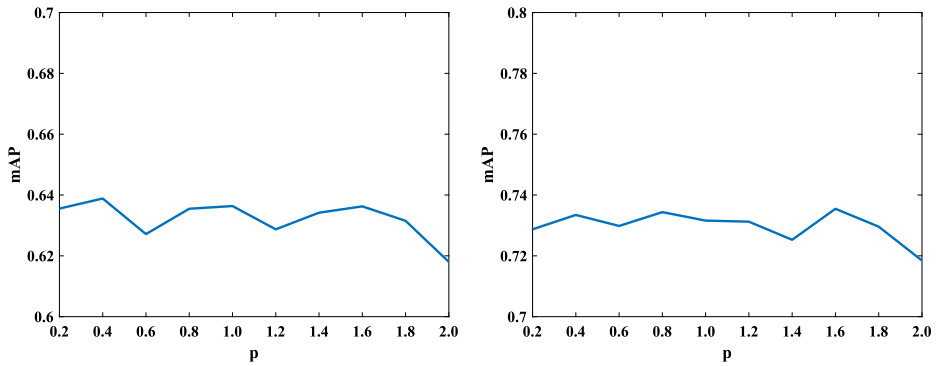
(a) On Cifar-10 (b) On NUS-WIDE

Fig. 1 Performance influence of parameter α

to a more stable learned hash code. Hence, α and p are the two most important parameters in the proposed method. In order to gauge their sensitivity, we conduct experiments to evaluate the influence on performance of parameters α and p. We draw the MAP curves with different values of α and p in Figs. 1 and 2, respectively, with α ranging from 10^{-4} to 10^4 and p ranging from 0.2 to 2.0. During the experiments, we set the code length to 64. We repeat the experiments five times for each value of α and p and extract the average MAP performance.

In Fig. 1, we can see that the MAP performance has low sensitivity regarding the parameter α. In addition, when the value of α is 0, the performance is the lowest, which means that learning the hash codes for the training samples and the projection for out-of-samples simultaneously is beneficial to the aggregate MAP performance.

Figure 2 shows the influence of the parameter p. It is apparent that the MAP performance is satisfactory when the values of p are lower than 2. In particular, when the value of p is 2 (i.e., the $\ell_{2,p}$-norm is ℓ_2-norm), the MAP performance deteriorates. This phenomenon also proves that the $\ell_{2,p}$-norm ($(0 < p \leq 2)$) adopted in the proposed method leads to potentially more stable hash codes.



(a) On CIFAR-10 (b) On NUS-WIDE

Fig. 2 Performance influence of parameter p

4 Conclusion

In this study, a novel discrete hashing method called SDHSL is proposed, which learns hash codes based on semantic label similarity. The proposed SDHSL jointly learns the hash codes of the training samples and the hash functions to obtain hash codes for samples outside the training set. The experiments performed on two benchmark datasets confirmed the superiority of our method under various retrieval scenarios.

References

1. Attouch H, Bolte J, Svaiter BF (2013) Convergence of descent methods for semi-algebraic and tame problems: proximal algorithms, forward–backward splitting, and regularized gauss–seidel methods. Math Program 137(1-2):91–129
2. Bolte J, Sabach S, Teboulle M (2014) Proximal alternating linearized minimization for nonconvex and nonsmooth problems. Math Program 146(1-2):459–494
3. Cheng Z, Shen J (2016) On effective location-aware music recommendation. ACM Trans Inf Syst (TOIS) 34(2):13
4. Dasgupta A, Kumar R, Sarlós T (2011) Fast locality-sensitive hashing. In: Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, pp 1073–1081
5. Gong Y, Lazebnik S (2011) Iterative quantization: a procrustean approach to learning binary codes. In: IEEE Conference on computer vision and pattern recognition, pp 817–824
6. Gong Y, Lazebnik S, Gordo A, Perronnin F (2013) Iterative quantization: a procrustean approach to learning binary codes for large-scale image retrieval. IEEE Trans Pattern Anal Mach Intell 35(12):2916–2929
7. Gui J, Liu T, Sun Z, Tao D, Tan T (2016) Supervised discrete hashing with relaxation. IEEE transactions on neural networks and learning systems
8. Gui J, Liu T, Sun Z, Tao D, Tan T (2017) Fast supervised discrete hashing. IEEE Trans Pattern Anal Mach Intell PP(99):1–1
9. He K, Wen F, Sun J (2013) K-means hashing: an affinity-preserving quantization method for learning binary compact codes. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 2938–2945
10. Hoerl AE, Kennard RW (1970) Ridge regression: Applications to nonorthogonal problems. Technometrics, pp 69–82
11. Hu Z, Pan G, Wang Y, Wu Z (2016) Sparse principal component analysis via rotation and truncation. IEEE Trans Neur Net Learn Syst 27(4):875–890
12. Indyk P, Motwani R (1998) Approximate nearest neighbors: towards removing the curse of dimensionality. In: Proceedings of the thirtieth annual ACM symposium on Theory of computing, ACM, pp 604–613
13. Ji J, Li J, Yan S, Tian Q, Zhang B (2013) Min-max hash for jaccard similarity. In: 2013 IEEE 13th international conference on data mining (ICDM), IEEE, pp 301–309
14. Jiang QY, Li WJ (2015) Scalable graph hashing with feature transformation. In: IJCAI, pp 2248–2254
15. Jiang QY, Li WJ (2016) Deep cross-modal hashing. In: Arxiv
16. Jiang QY, Li WJ (2018) Asymmetric deep supervised hashing. In: Thirty-second AAAI conference on artificial intelligence
17. Kang WC, Li WJ, Zhou ZH (2016) Column sampling based discrete supervised hashing. In: AAAI, pp 1230–1236
18. Leng C, Cheng J, Wu J, Zhang X, Lu H (2014) Supervised hashing with soft constraints. In: ACM International conference on information and knowledge management, pp 1851–1854
19. Lin G, Shen C, van den Hengel A (2015) Supervised hashing using graph cuts and boosted decision trees. IEEE Trans Pattern Anal Mach Intell 37(11):2317–2331
20. Lin G, Shen C, Shi Q, Hengel AVD, Suter D (2014) Fast supervised hashing with decision trees for high-dimensional data. In: Computer vision and pattern recognition, pp 1971–1978
21. Lin G, Shen C, Suter D, Hengel AVD (2013) A general two-step approach to learning-based hashing. pp 2552–2559
22. Liu C, Lu T, Wang X, Cheng Z, Sun J, Hoi SC (2019) Compositional coding for collaborative filtering. arXiv:1905.03752

23. Liu H, Wang R, Shan S, Chen X (2016) Deep supervised hashing for fast image retrieval. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 2064–2072
24. Liu W, Mu C, Kumar S, Chang SF (2014) Discrete graph hashing. In: Advances in neural information processing systems, pp 3419–3427
25. Liu W, Wang J, Ji R, Jiang YG (2012) Supervised hashing with kernels. In: Computer vision and pattern recognition, pp 2074–2081
26. Liu W, Wang J, Kumar S, Chang SF (2011) Hashing with graphs. In: International conference on machine learning, ICML 2011, bellevue, Washington, USA, june 28 - july, pp 1–8
27. Liu X, Nie X, Zeng W, Cui C, Zhu L, Yin Y (2018) Fast discrete cross-modal hashing with regressing from semantic labels. In: 2018 ACM Multimedia conference on multimedia conference, ACM, pp 1662–1669
28. Lu J, Liong VE, Zhou J (2017) Deep hashing for scalable image search. IEEE Trans Image Proces 26(5):2352–2367
29. Luxburg UV (2010) Clustering stability: an overview. Foundations & Trends®in Machine Learning 2(3):2010
30. Matsushita Y, Wada T (2009) Principal component hashing: an accelerated approximate nearest neighbor search. In: Pacific-rim symposium on image and video technology, Springer, pp 374–385
31. Nie X, Jing W, Cui C, Zhang J, Zhu L, Yin Y (2019) Joint multi-view hashing for large-scale near-duplicate video retrieval. IEEE Transactions on Knowledge and Data Engineering
32. Nie X, Li X, Chai Y, Cui C, Xi X, Yin Y (2018) Robust image fingerprinting based on feature point relationship mining. IEEE Trans Inf Foren Sec 13(6):1509–1523
33. Nie X, Yin Y, Sun J, Liu J, Cui C (2017) Comprehensive feature-based robust video fingerprinting using tensor model. IEEE Trans Multimedia 19(4):785–796
34. Sánchez J, Perronnin F (2011) High-dimensional signature compression for large-scale image classification. In: 2011 IEEE conference on computer vision and pattern recognition (CVPR), IEEE, pp 1665–1672
35. Shen F, Gao X, Liu L, Yang Y, Shen HT (2017) Deep asymmetric pairwise hashing. In: Proceedings of the 2017 ACM on multimedia conference, ACM, pp 1522–1530
36. Shen F, Shen C, Liu W, Tao Shen H (2015) Supervised discrete hashing. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 37–45
37. Shen F, Shen C, Shi Q, Van dHA, Tang Z, Shen HT (2015) Hashing on nonlinear manifolds. IEEE Transactions on Image Processing A Publication of the IEEE Signal Processing Society 24(6):1839–1851
38. Shen F, Shen C, Shi Q, Van Den Hengel A, Tang Z (2013) Inductive hashing on manifolds. In: 2013 IEEE conference on Computer vision and pattern recognition (CVPR), IEEE, pp 1562–1569
39. Shen F, Xu Y, Liu L, Yang Y, Huang Z, Shen HT (2018) Unsupervised deep hashing with similarity-adaptive and discrete optimization. IEEE Trans Pattern Anal Mach Intell 40(12):3034–3044
40. Shen F, Zhou X, Yang Y, Song J, Shen HT, Tao D (2016) A fast optimization method for general binary code learning. IEEE Trans Image Process 25(12):5610–5621
41. Strecha C, Bronstein A, Bronstein M, Fua P (2012) Ldahash: Improved matching with smaller descriptors. IEEE Trans Pattern Anal Mach Intell 34(1):66–78
42. Tang J, Wang K, Shao L (2016) Supervised matrix factorization hashing for cross-modal retrieval. IEEE Trans Image Process 25(7):3157–3166
43. Wang J, Kumar S, Chang SF (2010) Sequential projection learning for hashing with compact codes. In: International conference on international conference on machine learning, pp 1127–1134
44. Wang J, Kumar S, Chang SF (2012) Semi-supervised hashing for large-scale search. IEEE Trans Pattern Anal Mach Intell 34(12):2393–2406
45. Wang J, Liu W, Sun AX, Jiang YG (2013) Learning hash codes with listwise supervision. In: Proceedings of the IEEE international conference on computer vision, pp 3032–3039
46. Wang Q, Zhang Z, Si L (2015) Ranking preserving hashing for fast similarity search. In: IJCAI, pp 3911–3917
47. Weiss Y, Torralba A, Fergus R (2009) Spectral hashing. In: Advances in neural information processing systems, pp 1753–1760
48. Wu C, Zhu J, Cai D, Chen C, Bu J (2013) Semi-supervised nonlinear hashing using bootstrap sequential projection learning. IEEE Trans Knowl Data Eng 25(6):1380–1393
49. Xia R, Pan Y, Lai H, Liu C, Yan S (2012) Supervised hashing for image retrieval via image representation learning. In: AAAI Conference on artificial intelligence
50. Xu X, Shen F, Yang Y, Shen HT, Li X (2017) Learning discriminative binary codes for large-scale cross-modal retrieval. IEEE Trans Image Process, pp 2494–2507

51. Yan TK, Xu XS, Guo S, Huang Z, Wang XL (2016) Supervised robust discrete multimodal hashing for cross-media retrieval. In: Proceedings of the 25th ACM international on conference on information and knowledge management, ACM, pp 1271–1280
52. Yang R (2013) New results on some quadratic programming problems. Dissertations & Theses - Gradworks
53. Yang Y, Ma Z, Yang Y, Nie F, Shen HT (2015) Multitask spectral clustering by exploring intertask correlation. IEEE Trans Cybernetics 45(5):1083–1094
54. Yang Y, Zha ZJ, Gao Y, Zhu X, Chua TS (2015) Corrections to "exploiting web images for semantic video indexing via robust sample-specific loss"[oct 14 1677-1689]. IEEE Trans Multimedia 17(2):256–256
55. Yu Z, Wang H, Lin X, Wang M (2016) Understanding short texts through semantic enrichment and hashing. IEEE Trans Knowl Data Eng 28(2):566–579
56. Zhang D, Li WJ (2014) Large-scale supervised multimodal hashing with semantic correlation maximization. In: Twenty-eighth AAAI conference on artificial intelligence, pp 2177–2183
57. Zhang P, Zhang W, Li WJ, Guo M (2014) Supervised hashing with latent factor models
58. Zhang Q, Wang Y, Qian J, Huang X (2016) A mixed generative-discriminative based hashing method. IEEE Trans Knowl Data Eng 28(4):845–857
59. Zhu L, Shen J, Xie L, Cheng Z (2017) Unsupervised visual hashing with semantic assistant for content-based image retrieval. IEEE Trans Knowl Data Eng 29(2):472–486

Hong Wang is an associate professor in the school of journalism and communication from Shandong Normal University. She has recieved her master degree from Shandong Normal University. Her current research interests lie in the area of digital media technology and information intrieval.