Stepwise Refinement Short Hashing for Image Retrieval

Yuan Sun
Sichuan University
Chengdu, China
sunyuan_work@163.com

Dezhong Peng*†
Sichuan University
Chengdu, China
pengdz@scu.edu.cn

Jian Dai
Beijing Institute of Technology
Beijing, China
daijian1000@163.com

Zhenwen Ren*
Southwest University of Science and Technology
Mianyang, China
rzw@njust.edu.cn

ABSTRACT

Due to significant advantages in terms of storage cost and query speed, hashing learning has attracted much attention for image retrieval. Existing hashing methods often acquiescently use long hash codes to guarantee performance, which greatly limits flexibility and scalability. Nevertheless, short hash codes are more suitable for devices with limited computing resources. When these methods use extremely short hash codes, it is difficult to meet the actual performance demand due to the information loss caused by the avalanche of dimension truncation. To address this issue, we propose a novel stepwise refinement short hashing (SRSH) for image retrieval that extracts critical features from high-dimensional image data to learn high-quality hash codes. Specifically, we propose a three-step coupled refinement strategy to relax a single hash function into three more flexible mapping matrices, such that the hash function can have more flexible to approximate precise hash codes and alleviate the information loss. Then, we adopt pairwise similarity preserving to promote coarse and fine hash codes to inherit intrinsic semantic structure from original data. Extensive experiments demonstrate the superior performance of SRSH on four image datasets.

CCS CONCEPTS

• **Information systems** → **Retrieval models and ranking**;

KEYWORDS

Image retrieval, Stepwise Refinement, Learning to hash

ACM Reference Format:
Yuan Sun, Dezhong Peng, Jian Dai, and Zhenwen Ren. 2023. Stepwise Refinement Short Hashing for Image Retrieval. In *Proceedings of the 31st ACM International Conference on Multimedia (MM '23), October 29-November 3, 2023, Ottawa, ON, Canada.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3581783.3611864

*Corresponding Author.
†Dezhong Peng is also with Chengdu Ruibei Yingte Information Technology Co., Ltd.

MM '23, October 29-November 3, 2023, Ottawa, ON, Canada

ACM ISBN 979-8-4007-0108-5/23/10…$15.00
https://doi.org/10.1145/3581783.3611864

1 INTRODUCTION

With the growing amount of high-dimensional data, image retrieval [7, 13, 47] has been widely used in the practical application of large-scale data analysis. Due to significant advantages in terms of storage cost and query speed, image hashing [6, 7, 48] is currently a hot issue in the field of image retrieval. The goal of learning to hash [30, 34] is to embed raw images and semantic intrinsic associations into a string of compact hash codes by adopting the hash function. The similarities between hash codes can be calculated through Hamming distance which greatly accelerates computation speed and reduces space complexity. Therefore, learning to hash is widely used in large-scale image retrieval [8], cross-modal retrieval [17, 18, 24, 43], image classification [35, 36, 41], multi-view learning [2–4, 40], and so on[10–12].

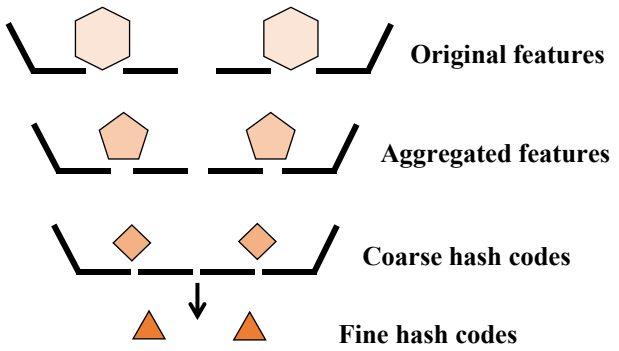


Figure 1: Illustration of the proposed SRSH. Similar to a sieving process physically comparing the size of particles with the aperture of a sieve, we design a three-step coupled refinement strategy with constraints to shrink the discrete solution space from loose to strict. Therefore, we can learn a richer set of hash functions to enhance discrimination.

Learning to hash requires that the learned hash codes should be as compact and short as possible. In practice, existing hashing methods [26, 31, 38] usually learn default long hash codes (*i.e.*, 64 or 128 bits) to represent image data. The computational complexity of the hashing model is usually approximately equal to $O(ln)$ or $O(l^2n)$ (where l is hash length and n is the number of training data). When facing with large-scale image data, the long hash codes is bound to cause large computational consumption, which greatly

limit flexibility and scalability. Contrarily, short hash codes are more suitable for devices with limited computing resources and have more fast training and query speed. For the training images from c categories, the length of hash codes should be no less than $log_2 c$, otherwise the hash length will not be able to distinguish all categories of samples. Therefore, we define the short-length l as an integer slightly larger than $log_2 c$. For example, if a dataset has 10 categories, the short-length of hash codes is 4 bits. However, when the length of hash code is extremely short, the retrieval performance will degrade dramatically due to the large information loss. Therefore, how to mitigate the large amount of information loss caused by short hash codes is a great challenge.

To overcome the above drawbacks, we propose a novel **stepwise refinement short hashing (SRSH)** framework that is tailored for learning high-quality short hash codes. As shown in Fig 1, our proposed SRSH is the process of gradually reducing the range of solution sets. That is, SRSH can progressively refine the original features and endow more freedom for hash function so that it can fit and address the problem of large information loss well as much as possible. Specifically, different from the traditional single hash strategy, SRSH refines features step by step via a three-step coupled strategy with constraints from loose to strict, thereby learning fine hash codes to mitigate large information loss caused by dimensional avalanches. In summary, the main contributions of our proposed method are summarized as follows:

- We propose a novel stepwise refinement short hashing (SRSH). For the first time, we propose a three-step coupled refinement strategy, which can shrink the discrete solution space and is more likely to identify stronger features by learning a richer set of projections.
- We propose to relax a single hash mapping matrix into three more flexible mapping matrices, which can make the hash function to have more flexible to fit short hash codes and alleviate the information loss.
- A plentiful experiments on four widely used large-scale datasets, *i.e.*, CIFAR-10, MNSIT, NUS-WIDE, and Caltech-256, demonstrate that SRSH outperforms the state-of-the-arts with different hash lengths, especially, short-length.

2 RELATED WORK

During past years, image hashing mainly has two branches, *i.e.*, unsupervised hashing [17, 33] and supervised hashing [15, 16]. Since unsupervised hashing [32] usually does not require intensive data annotation, this is more attractive in practice. For example, unsupervised contrastive cross-modal hashing [17] proposes a robust contrastive learning method to overcome the false-negative pair problem, thereby improving the quality of hash codes for unsupervised cross-modal retrieval. However, thanks to the strong supervision ability of label information, supervised hashing has made tremendous progress. For example, supervised discrete hashing (SDH) [31] adopts linear classifier to guide the generation of hash codes. To speed up the algorithm and improve the retrieval performance, fast supervised discrete hashing (FSDH) [9] transforms labels into hash codes and fast scalable supervised hashing (FSSH) [25] utilizes a pre-computed intermediate term to avoid the use of large $n \times n$ similarity matrices. Nevertheless, the static predefined similarity matrix could not accurately describe the true similarity of images. Supervised adaptive similarity matrix hashing (SASH) [34] is proposed that dynamically learns the adaptive similarity matrix to guide the generation of hash codes. To excavate the intrinsic latent features and the underlying topological structure, ordinal-preserving latent graph hashing (OLGH) [45] utilizes the latent representation to preserve the high-order topological structure. And probability ordinal-preserving semantic hashing (POSH) [46] proposes a novel ordinal-preserving hashing concept based on the non-parametric Bayesian theory to further explore triplet-level ordinal structure. Moreover, to overcome the problem of conceptual drift in non-stationary environments, deep incremental hashing (DIH) [37] is proposed that captures high-level descriptive features from images to learn discriminative binary codes.

Though existing hashing methods [39] can achieve efficient retrieval and storage, they mainly have two limitations: (1) When hash codes are long, it will inevitably lead to higher computational complexity. (2) When hash codes is extremely short, the performance is fairly poor and difficult to meet the demand. Therefore, how to learn high-quality short hash codes for scalable image retrieval is a great challenge [29]. To meet the performance and storage requirements in practical applications, supervised short length hashing (SSLH) [23] proposes a mutual reconstruction strategy to reduce the loss of semantic information, and a robust estimator term to enhance the robustness of short hash codes. Reinforced short length hashing (RSLH) [22] unifies mutual regression, semantic pairwise similarity, and a relaxed strategy for learning high-quality short hash codes. However, these two short hashing methods are not flexible and scaleable for large-scale image retrieval due to discrete cyclic coordinate descent (DCC) optimization. We also note that the existing hashing prototype directly learns a single-layer hash function to generate compact low-dimensionality hash codes from high-dimensional image data. This sudden dimensionality avalanche inevitably results in massive discriminative information loss, especially, the extremely low length of hash codes. In addition, some researchers propose strongly constrained discrete hashing (SCDH) [5], *i.e.*, using two extra strong hash constraints (bit balance and de-correlation constraints) on hash codes instead of a simple binary discretization constraint can encourage the generation of compact high-quality binary codes as much as possible to achieve better retrieval accuracy. However, hash codes learned from feature data is difficult to inherit these two strong constraint properties in one step due to this strict hash constraint problem. Overall, for short hashing practical applications, the problem of information loss is very serious due to the uncontrolled environment of image collection and the avalanche of dimension truncation.

3 PROPOSED METHOD

3.1 Notation

For image retrieval task, $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_n] \in \mathbb{R}^{d \times n}$ is denoted as training set with n images, where d is the feature dimension of each instance. The corresponding ground-truth class label matrix is represented as $\boldsymbol{L} = [\boldsymbol{l}_1, \boldsymbol{l}_2, \cdots, \boldsymbol{l}_n] \in \mathbb{R}^{c \times n}$, where c is the number of classes. $\mathbf{Y} = [\boldsymbol{y}_1, \boldsymbol{y}_2, \cdots, \boldsymbol{y}_m] \in \mathbb{R}^{d \times m}$ is denoted as query set with m images. Hashing learning methods aim to learn hash codes $\boldsymbol{B} = [\boldsymbol{b}_1, \boldsymbol{b}_2, \cdots, \boldsymbol{b}_n] \in \{-1, 1\}^{l \times n}$ in Hamming space to preserve

the intrinsic similarities from the original feature space. Follow the existing hashing methods, we suppose all image instances to be zero centered, *i.e.*, $\sum_{i=1}^{n} \boldsymbol{x}_i = 0$. To capture the non-linear underlying structure from the image instances, we adopt a popular kernelization operation [20] (such as RBF kernel mapping). In particular, we randomly choose k points as anchors (*i.e.*, $\boldsymbol{a}_j$) from the training images. The d-dimension feature of each image can be denoted as a kernel feature as

$$\phi(\boldsymbol{x}_i) = [exp(\frac{\|\boldsymbol{x}_i - \boldsymbol{a}_1\|_2^2}{-2\sigma^2}), ..., exp(\frac{\|\boldsymbol{x}_i - \boldsymbol{a}_k\|_2^2}{-2\sigma^2})]^\top \tag{1}$$

where $\sigma = \frac{1}{nk}\sum_{i=1}^{n}\sum_{j=1}^{k}\|\boldsymbol{x}_i - \boldsymbol{a}_j\|_2$ is the kernel width. For the sake of convenience, we use $\boldsymbol{H}_0$ to represent $\phi(\boldsymbol{X})$.

3.2 Formulation

The existing hashing methods project high-dimensional data into low-dimensional hash codes by a single-layer hash function, which could result in a large volume of the kernel discriminative information loss and the representation error. Particularly, when the length of hash codes is extremely short, the retrieval accuracy of existing methods will degrade dramatically due to the information loss caused by the avalanche of dimension truncation. To address the limitation, we propose a three-step coupled strategy with constraints from loose to strict to refine features and enhance the representation capability.

Aggregation feature: A single-layer hashing function can directly map high-dimensional image features with size $\mathbb{R}^{d\times n}$ into low-dimensional hash codes with size $\mathbb{R}^{l\times n}$. To alleviate the large loss of discriminative information caused by dimensionality avalanche, we learn the transition subspace to aggregate discriminative features. The motivation for introducing the transitional transformation space is that learning the hash function for three relatively simple tasks is more flexible than learning a hash projection matrix for a complex task. We thus use the first-layer hash function $\mathbf{W}_1 \in \mathbb{R}^{l\times k}$ to learn aggregated features $\boldsymbol{H}_1$ as follows

$$\min_{\boldsymbol{W}_1,\boldsymbol{H}_1} \|\boldsymbol{H}_1 - \boldsymbol{W}_1\boldsymbol{H}_0\|_F^2 + \lambda\|\boldsymbol{W}_1\|_F^2 \tag{2}$$

where λ is a regularization parameter.

Coarse hash codes: To further learn a set of discrete hash codes, we use the second-layer hash function to transform the aggregated discriminative information extracted in the first layer into coarse hash codes, meanwhile preserving the intrinsic similarities in the original space. Therefore, we enforce the discrete binary constraint to learn coarse hash codes. The quality of generated fine hash codes in the third layer can be improved. Formally, coarse hash codes $\boldsymbol{H}_2$ can be written as

$$\begin{aligned}&\min_{\boldsymbol{W}_2,\boldsymbol{H}_2} \|\boldsymbol{H}_2 - \boldsymbol{W}_2\boldsymbol{W}_1\boldsymbol{H}_0\|_F^2 + \lambda(\|\boldsymbol{W}_1\|_F^2 + \|\boldsymbol{W}_2\|_F^2)\\ &\Rightarrow \min_{\boldsymbol{W}_2,\boldsymbol{H}_2} \|\boldsymbol{H}_2 - \boldsymbol{W}_2\boldsymbol{H}_1\|_F^2 + \lambda\|\boldsymbol{W}_2\|_F^2\\ &\quad s.t.\ \boldsymbol{H}_2 \in \{-1,1\}^{l\times n}\end{aligned} \tag{3}$$

where λ is a regularization parameter and $\boldsymbol{W}_2$ is the second-layer hash function.

Fine hash codes: To unleash the full potential of hash codes and improve representation abilities, we impose bit de-correlation $\boldsymbol{H}_3^\top\boldsymbol{H}_3 = n\boldsymbol{I}$ and bit balance $\mathbf{1}\boldsymbol{H}_3 = \mathbf{0}$ [5] to obtain more efficient fine hash codes. We utilize structure consistency mapping $\boldsymbol{W}_3$ as the third-layer hash function. It can integrate the information of the previous two layers and effectively capture the resemblance among these two hash codes to output more compact fine hash codes $\boldsymbol{H}_3$. For the value of structure consistency, it should not be the identity matrix at least. The structure consistency mapping should take some of the responsibility, such that hash codes from the same classes can be kept close together. The final fine hash codes can update as follows

$$\begin{aligned}&\min_{\boldsymbol{W}_3,\boldsymbol{H}_3} \|\boldsymbol{H}_3 - \boldsymbol{W}_3\boldsymbol{W}_2\boldsymbol{W}_1\boldsymbol{H}_0\|_F^2\\ &\qquad + \lambda(\|\boldsymbol{W}_1\|_F^2 + \|\boldsymbol{W}_2\|_F^2 + \|\boldsymbol{W}_3\|_F^2)\\ &\Rightarrow \min_{\boldsymbol{W}_3,\boldsymbol{H}_3} \|\boldsymbol{H}_3 - \boldsymbol{W}_3\boldsymbol{H}_2\|_F^2 + \lambda\|\boldsymbol{W}_3\|_F^2\\ &s.t. \boldsymbol{H}_3 \in \{-1,1\}^{l\times n}, \mathbf{1}\boldsymbol{H}_3 = \mathbf{0}, \boldsymbol{H}_3^\top\boldsymbol{H}_3 = n\boldsymbol{I}\end{aligned} \tag{4}$$

We consider the three-layer hash mapping to be equally important. Therefore, we can obtain the following problem

$$\begin{aligned}&\min_{\boldsymbol{W}_i,\boldsymbol{H}_i} \sum_{i=1}^{3}\|\boldsymbol{H}_i - \boldsymbol{W}_i\boldsymbol{H}_{i-1}\|_F^2 + \lambda(\sum_{i=1}^{3}\|\boldsymbol{W}_i\|_F^2)\\ &s.t.\ \boldsymbol{H}_2 \in \{-1,1\}^{l\times n}, \boldsymbol{H}_3 \in \{-1,1\}^{l\times n},\\ &\qquad \mathbf{1}\boldsymbol{H}_3 = \mathbf{0}, \boldsymbol{H}_3^\top\boldsymbol{H}_3 = n\boldsymbol{I}\end{aligned} \tag{5}$$

Similarity preserving: To reduce semantic information loss, it is expected to deliver rich feature and semantic information to enhance discriminative hash code learning. If two images $(\boldsymbol{H}_{0i}, \boldsymbol{H}_{0j})$ are similar in the feature space, they should have similar hash codes $(\boldsymbol{H}_{3i}, \boldsymbol{H}_{3j})$ in Hamming space, and vice versa. We use cosine similarity s_{ij} to measure the semantic similarity between sample pairs $(\boldsymbol{x}_i, \boldsymbol{x}_j)$, *i.e.*, $s_{ij} = \cos(\boldsymbol{y}_i, \boldsymbol{y}_j)$. The semantic similarities can be transformed as $\min\sum_{i,j=1}^{n}\left(\boldsymbol{h}_i^T\boldsymbol{h}_j/l - s_{ij}\right)^2$, due to $\|\boldsymbol{h}_i\| = \sqrt{l}$. Hence, to embed the semantic information into coarse and fine hash codes as much as possible, we adopt the inner product between coarse and fine hash codes to preserve similarity:

$$\min_{\boldsymbol{H}_2,\boldsymbol{H}_3} \|\boldsymbol{H}_2^\top\boldsymbol{H}_3 - l\boldsymbol{L}^\top\boldsymbol{L}\|_F^2 \tag{6}$$

We can observe that this way can avoid using a large $n\times n$ pairwise similarity matrix. Obviously, space complexity and computational cost can be reduced during optimization.

Overall SRSH model: Based on the above insights, the proposed SRSH refines the original features hierarchically to reduce the information loss of short hash codes, and then further inherits the intrinsic similarity information to obtain high discriminant fine hash codes. Afterward, SRSH can be formulated as

$$\begin{aligned}&\min_{\boldsymbol{W}_i,\boldsymbol{H}_i} \sum_{i=1}^{3}\|\boldsymbol{H}_i - \boldsymbol{W}_i\boldsymbol{H}_{i-1}\|_F^2 + \lambda\sum_{i=1}^{3}\|\boldsymbol{W}_i\|_F^2\\ &\qquad + \alpha\|\boldsymbol{H}_2^\top\boldsymbol{H}_3 - l\boldsymbol{L}^\top\boldsymbol{L}\|_F^2\\ &s.t.\ \boldsymbol{H}_2 \in \{-1,1\}^{l\times n}, \boldsymbol{H}_3 \in \{-1,1\}^{l\times n},\\ &\qquad \mathbf{1}\boldsymbol{H}_3 = \mathbf{0}, \boldsymbol{H}_3^T\boldsymbol{H}_3 = n\boldsymbol{I}\end{aligned} \tag{7}$$

In summary, SRSH adopts a three-step coupled refinement strategy with constraints to shrink the discrete solution space from loose to strict and learns a richer set of hash functions to enhance

discrimination ability of short-length hash codes. Specifically, we aggregate beneficial feature information in the first-layer transition subspace and refine coarse hash codes in the second-layer Hamming space. Finally, we learn fine hash codes with strong hash constraints, such that the learned fine hash codes can not only capture the low-level feature distributions but also be correlated with high-level semantics.

3.3 Optimization

In this subsection, we represent how the challenging discrete optimization problems of our SRSH are effectively solved. From the objective function, it can be observed each sub-problem is convex. The overall objective function (7) can be optimized by the iterative optimization strategy. Specifically, we update each variable to be solved while fixing the others and iterate this process until convergence.

► **Step-1 update $\{\boldsymbol{W}_i\}_{i=1}^3$-subproblem**:

Fixing other variables variables, $\{\boldsymbol{W}_i\}_{i=1}^3$-subproblem of (7) can be rewritten as

$$\min_{\boldsymbol{W}_i} \|\boldsymbol{H}_i - \boldsymbol{W}_i\boldsymbol{H}_{i-1}\|_F^2 + \lambda\|\boldsymbol{W}_i\|_2^2 \tag{8}$$

Let its derivative with respect to $\boldsymbol{W}_i$ to zero, the closed form solution of $\boldsymbol{W}_i$ for problem (8) is

$$\boldsymbol{W}_i = (\boldsymbol{H}_i\boldsymbol{H}_{i-1}^\top)(\boldsymbol{H}_{i-1}\boldsymbol{H}_{i-1}^\top + \lambda\boldsymbol{I})^{-1} \tag{9}$$

► **Step-2 update $\boldsymbol{H}_1$-subproblem**:

Fixing other variables variables $\{\boldsymbol{W}_i\}_{i=1}^3$ and $\{\boldsymbol{H}_i\}_{i=2}^3$, $\boldsymbol{H}_1$-subproblem of (9) with respect to $\boldsymbol{H}_1$ can reduce to the following formulation

$$\min_{\boldsymbol{H}_1} \sum_{i=1}^{2} \|\boldsymbol{H}_i - \boldsymbol{W}_i\boldsymbol{H}_{i-1}\|_F^2 \tag{10}$$

Taking the partial derivative of Eq. (10) with respect to $\boldsymbol{H}_1$ and setting it to zero, the closed form solution can be obtained as follows

$$\boldsymbol{H}_1 = (\boldsymbol{W}_2^\top\boldsymbol{W}_2 + \boldsymbol{I})^{-1}(\boldsymbol{W}_1\boldsymbol{H}_0 + \boldsymbol{W}_2^\top\boldsymbol{H}_2) \tag{11}$$

► **Step-3 update $\boldsymbol{H}_2$-subproblem**:

Fixing other variables variables $\{\boldsymbol{W}_i\}_{i=1}^3$ and $\boldsymbol{H}_i$ (i=1, 3), $\boldsymbol{H}_2$-subproblem can further reduce to the following formulation

$$\begin{aligned} &\min_{\boldsymbol{H}_2} \sum_{i=2}^{3} \|\boldsymbol{H}_i - \boldsymbol{W}_i\boldsymbol{H}_{i-1}\|_F^2 + \alpha\|\boldsymbol{H}_2^\top\boldsymbol{H}_3 - l\boldsymbol{L}^\top\boldsymbol{L}\|_F^2 \\ &s.t.\ \boldsymbol{H}_2 \in \{-1,1\}^{l\times n} \end{aligned} \tag{12}$$

Setting the derivative of Eq. (12) with respect to $\boldsymbol{H}_2$ to zero, and we can obtain

$$\boldsymbol{H}_2 = sign(\boldsymbol{W}_2\boldsymbol{H}_1 + \boldsymbol{W}_3^\top\boldsymbol{H}_3 + \alpha l\boldsymbol{H}_3\boldsymbol{L}^\top\boldsymbol{L}) \tag{13}$$

► **Step-4 update $\boldsymbol{H}_3$-subproblem:**

Fixing other variables variables $\{\boldsymbol{W}_i\}_{i=1}^3$ and $\{\boldsymbol{H}_i\}_{i=1}^2$, $\boldsymbol{H}_3$-subproblem of (7) can be rewritten as

$$\begin{aligned} &\min_{\boldsymbol{H}_3} \sum_{i=2}^{3} \|\boldsymbol{H}_i - \boldsymbol{W}_i\boldsymbol{H}_{i-1}\|_F^2 + \alpha\|\boldsymbol{H}_2^\top\boldsymbol{H}_3 - l\boldsymbol{L}^\top\boldsymbol{L}\|_F^2 \\ &s.t.\ \boldsymbol{H}_3 \in \{-1,1\}^{l\times n}, \boldsymbol{1}\boldsymbol{H}_3 = \boldsymbol{0}, \boldsymbol{H}_3\boldsymbol{H}_3^T = n\boldsymbol{I} \end{aligned} \tag{14}$$

With simple matrix manipulation, Eq.(18) can be transformed as

$$\begin{aligned} &\max_{\boldsymbol{H}_3} \boldsymbol{H}_3(\boldsymbol{W}_3^\top\boldsymbol{H}_2 + \alpha l\boldsymbol{H}_2\boldsymbol{L}^\top\boldsymbol{L})^\top \\ &s.t.\ \boldsymbol{H}_3 \in \{-1,1\}^{l\times n}, \boldsymbol{1}\boldsymbol{H}_3 = \boldsymbol{0}, \boldsymbol{H}_3^\top\boldsymbol{H}_3 = n\boldsymbol{I} \end{aligned} \tag{15}$$

Let $\boldsymbol{E} = (\boldsymbol{W}_3^\top\boldsymbol{H}_2 + \alpha l\boldsymbol{H}_2\boldsymbol{L}^\top\boldsymbol{L})^\top$. Thereupon, to better solve the closed-form solution, we drop out the discrete constraint and obtain the following the formula.

$$\max_{\boldsymbol{H}_3} \boldsymbol{H}_3\boldsymbol{E} \quad s.t.\ \boldsymbol{1}\boldsymbol{H}_3 = \boldsymbol{0}, \boldsymbol{H}_3^\top\boldsymbol{H}_3 = n\boldsymbol{I} \tag{16}$$

Its the closed-form solution is

$$\boldsymbol{H}_3 = \sqrt{n}[\boldsymbol{U}, \widetilde{\boldsymbol{U}}][\boldsymbol{V}, \widetilde{\boldsymbol{V}}]^T \tag{17}$$

Here, $\boldsymbol{U} = [\boldsymbol{U}_1, \boldsymbol{U}_2, ..., \boldsymbol{U}_{\hat{l}}]$ and $\boldsymbol{V} = [\boldsymbol{V}_1, \boldsymbol{V}_2, ..., \boldsymbol{V}_{\hat{l}}]$ are calculated by the SVD of $\boldsymbol{J}\boldsymbol{E}$, where $\boldsymbol{J} = \boldsymbol{I}_n - \frac{1}{n}\boldsymbol{1}_n\boldsymbol{1}_n^\top$, *i.e.*,

$$\boldsymbol{J}\boldsymbol{E} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{U}^\top = \sum_{i=1}^{\hat{r}} \sigma_i u_k v_k^T \tag{18}$$

where eigenvalues is ordered by $0 \geq \sigma_{\hat{l}} \geq \cdots \geq \sigma_1$. $\widetilde{\boldsymbol{U}}$ and $\widetilde{\boldsymbol{V}}$ can be obtained by the Gram-Schmidt process. Note that $\widetilde{\boldsymbol{U}}$ and $\widetilde{\boldsymbol{D}}$ will be empty if $\hat{l} = l$.

Algorithm 1 Stepwise Refinement Short Hashing

Require: Training set $\boldsymbol{X}$ with label matrix $\boldsymbol{L}$, query set $\boldsymbol{Y}$, hash length l, and the number of anchors k

```
Parameter: α and λ;
Initialize: W_i and H_i;
Obtain kernel features H_0 by kernel trick;
repeat
    Update W_i via (9);
    Update H_1 via (11);
    Update H_2 via (13);
    Update H_3 via (15);
until Satisfy the stop criteria;
```

Ensure: Hash function $\boldsymbol{W}_i$ and hash codes $\boldsymbol{H}_i$.

3.4 Computational Complexity

The computational complexity of our SRSH is built on top of six optimizing sub-problems. Since SRSH has fast convergence property, the number of iteration t is small. Hence, we only concern the complexity of each iteration. Here, the feature dimension is k, the number of classes is c, and l is hash length. For the $\boldsymbol{W}_1$-subproblem, the main complexity is $O\left(nlk + nk^2 + l^3\right)$. For the $\boldsymbol{W}_2$-subproblem, the complexity is $O\left(2nl^2 + l^3\right)$. For the $\boldsymbol{W}_3$-subproblem, the main complexity is $O\left(2nl^2 + l^3\right)$. For the $\boldsymbol{H}_1$-subproblem, the complexity is $O\left(2l^3 + nlk + nl^2\right)$. For the $\boldsymbol{H}_2$-subproblem, the complexity is $O\left(2cln + nl^2\right)$. For the $\boldsymbol{H}_3$-subproblem, the main complexity is the SVD of a $l \times n$ matrix, whose complexity is $O(l^2n)$. In practice, $k \ll n$ and c and l are small. Hence, the computation complexity is linear to n, and SRSH is indeed highly efficient for large-scale data.

3.5 Convergence Analysis

For the convergence analysis, the six subproblems corresponding to all variables are convex with respect to one variable and have a closed-form optimal solution. We can easily know the whole objective function is monotonously decreasing for each iteration. In addition, the objective function has a lower bound of zero. According to the bounded monotone convergence theorem, SRSH can converge to at least one local minimum.

4 EXPERIMENTS

4.1 Datasets

We perform experiments on four large-scale datasets, *i.e.*, CIFAR-10 [27], MNIST [42], NUS-WIDE [1], and Caltech-256 [46].

CIFAR-10 consists of 60000 images from 10 classes collected from 80 million tiny images. Each class has 6000 images. We employ a 512-dimension GIST feature vector to represent each image, which is extracted from the original color image with the size of 32×32. For CIFAR-10, we randomly select 1000 images used for retrieval and the rest of 59000 images as the training set. **MNIST** is composed of 70000 handwritten digits images from 10 classes (*i.e.*, from 0 to 9). In our experiments, we crop and normalize each image to 28×28, and then employ a 784dimensional feature vector to represent each image. For MNIST, we randomly choose 1000 images as the query set and the rest of 69000 images as the training set. **NUS-WIDE** is collected from a real-world large-scale multi-label image dataset Flickr, which contains 269648 images with 81 ground-truth labels. Following the setting in [31], we choose the 21 most frequent labels with 195834 images to evaluate the effect. In our experiments, we use a 500-dimensional bag-of-words feature vector to represent each image. For NUS-WIDE, we randomly sample 2100 images as the query set, and the remained images as the training set. In this multi-label dataset, if two images share at least one common label, they are semantically similar, and vice versa. **Caltech-256** [45] contains 29780 samples from 256 classes. Each category has at least 80 images. We extract the 1024-dimensional deep features by using CNN pre-trained on ImageNet, which are obtained from the last fully-connected layer of the neural network. In our experiments, we randomly select 1000 images as the query set and the rest as the training set.

4.2 Experimental Setting

To demonstrate the effectiveness, we compare our method SRSH with some representative hashing methods, including FSDH [9], FSSH [25], SSLH [23], RSLH [22], SCDH [5], POPSH [46] , OLGH [45], SASH [34], and LBSE [21]. Since SRSH is a shallow method, we do not compare them with deep hashing methods. Since SSLH, RSLH, and SASH have high space complexity, we randomly sample 2000 images as training sets on the CIFAR-10 and MNIST datasets, and sample 10500 images on the NUS-WIDE dataset. For fairness, we use the public codes with default parameters provided by the respective authors. In the experiments, the length l of short hash codes is a little bigger than $log_2 c$, which are 3.32, 3.32, 4.39, and 8 on the CIFAR-10, MNIST, NUS-WIDE, and Caltech256, respectively. Hence, we set short-length as 4, 4, 5, and 8 bits. The maximum bit length is set to 64. All experiments are performed on our windows PC with an Intel Core i9 (3.2GHz) CPU and 64GB RAM. For our SRSH, we set anchors as $k = 1500$. According parameter sensitivity analysis, on four datasets, we set α and λ as $\{10^{-2}, 10^{-5}\}$, $\{10^{-2}, 10^{-4}\}$, $\{10^{2}, 10^{-4}\}$, and $\{10^{-1}, 1\}$, respectively.

4.3 Evaluation Protocols

We use four widely used evaluation protocols [19], *i.e.*, mean average precision (mAP) [14], Precision@topN [44], NDCG@100, and precision-recall curves [28]. mAP is the mean of the overall average precision values. Precision@topN represents precision of the top N retrieved samples returned to the users. NDCG@100 represents normalized discounted cumulative gain at rank 100. For these evaluation protocols, a larger value means better retrieval performance.

4.4 The mAP Results

We report the mAP results with different bits on four benchmark datasets in the left side of tables 1 to 4, respectively. According to these experimental results, SRSH outperforms all compared methods in most cases, which show its effectiveness. Due to the three-step coupled refinement strategy, for short-length codes, we obtain absolute boosts of 5.53%, 1.89%, 3.14%, and 2.48% on four datasets, respectively. It indicates that the proposed SRSH has a distinct advantage for short-length hash codes. We can observe that the mAP is increased and the improvement of mAP reduces, with the increase in length of hash codes. For large class image data, all methods require longer hash codes to achieve satisfactory performance. Moreover, since the state-of-the-art short-length hashing methods (*i.e.*, SSLH, RSLH, and SASH) use the pairwise similarity matrix to preserve semantics, these methods can result in high space complexity and insufficient memory. We thus randomly sample some images for training instead of using the entire training set. Clearly, they cannot achieve satisfactory performance.

4.5 The Precision@TopN Results

The Precision@TopN results of all compared methods on four datasets are summarized in the middle side of tables 1 to 4, respectively. the proposed SRSH achieves the highest precision results in terms of Precision@5000 on the first three datasets and Precision@50 on the last one. It indicates SRSH can extract critical features from high-dimensional image data to learn high-quality short-length hash codes, thereby improving retrieval performance. Specifically, compared to the state-of-the-art hashing methods, for short-length hash codes, we obtain the boosts of 5.05%, 2.24%, 0.75%, and 4.23% on all datasets, respectively. As the increase in length of hash codes, Precision@TopN is also increased. From these results, it can demonstrate that our SRSH can achieve very delightful retrieval performance, when we use short-length hash codes.

4.6 The NDCG@100 Results

The NDCG@100 results of all compared methods on four datasets are summarized in the right side of tables 1 to 4, respectively. We can see that there is a trend towards better NDCG@100 results for all comparison methods, as the bit length increases. It indicates that long-length hash codes can provide more discriminative information. For short-length hash codes, our SRSH method can obtain

Table 1: The retrieval performance results (%) with different bits on the CIFAR-10 dataset. The best values are shown in bold.

Method	mAP					Precision@5000					NDCG@100				
	4	8	16	32	64	4	8	16	32	64	4	8	16	32	64
SDH [31]	25.11	31.45	38.24	42.22	45.23	29.37	39.36	46.60	49.92	52.29	31.34	42.02	49.30	51.49	53.52
FSDH [9]	26.50	33.95	37.82	42.40	46.22	31.58	40.74	45.21	49.75	53.16	31.90	42.70	46.71	51.85	53.91
FSSH [25]	22.25	37.49	60.21	65.85	67.03	19.52	33.98	53.32	59.63	60.03	19.73	34.19	53.12	59.68	60.01
SSLH [23]	20.52	26.68	33.60	37.86	39.98	22.29	33.48	44.49	47.97	49.48	23.06	33.62	44.31	47.97	49.76
RSLH [22]	27.48	30.64	28.81	44.92	46.47	21.34	21.80	39.88	32.06	34.86	26.99	30.34	39.85	44.79	46.63
SCDH [5]	55.12	62.53	65.54	68.41	69.17	49.76	57.93	59.53	61.36	62.13	49.78	57.97	59.57	61.33	62.40
POPSH [46]	34.80	60.75	64.33	67.94	69.14	33.21	55.66	59.71	63.29	63.77	33.39	55.95	60.61	63.29	63.85
OLGH [45]	47.46	61.22	66.40	68.31	64.22	42.92	56.15	60.44	63.56	62.11	42.84	56.12	60.50	63.80	63.16
SASH [34]	15.87	16.89	21.36	24.42	27.11	18.20	21.56	28.25	34.14	39.46	17.99	21.36	27.74	33.46	38.76
LBSE[21]	51.23	61.50	68.07	69.49	71.09	46.23	55.73	62.52	62.29	64.07	46.43	55.77	62.54	62.97	64.21
Our SRSH	**60.65**	**64.93**	**68.11**	**73.24**	**73.47**	**54.81**	**60.05**	**64.04**	**66.88**	**66.80**	**54.90**	**59.95**	**64.39**	**66.87**	**66.78**

Table 2: The retrieval performance results (%) with different bits on the MNIST dataset. The best values are shown in bold.

Method	mAP					Precision@5000					NDCG@100				
	4	8	16	32	64	4	8	16	32	64	4	8	16	32	64
SDH [31]	51.98	79.17	90.62	92.91	93.24	53.74	53.74	92.15	93.76	93.59	54.30	81.10	91.52	92.71	93.05
FSDH [9]	69.18	89.91	91.38	93.57	93.99	71.78	71.78	92.99	93.02	93.55	71.77	90.60	92.63	93.40	93.85
FSSH [25]	37.33	37.64	94.57	94.59	95.17	36.94	36.94	93.57	93.43	94.13	36.67	36.87	93.56	93.48	91.15
SSLH [23]	37.33	37.64	94.57	94.59	95.17	36.94	36.94	93.57	93.43	94.13	36.67	36.87	93.56	93.48	91.15
RSLH [22]	72.26	66.86	89.31	92.46	93.66	67.71	67.71	91.88	90.42	92.31	71.65	86.51	91.84	92.34	93.71
SCDH [5]	82.91	91.57	87.42	91.68	91.30	81.67	81.67	85.40	90.18	90.00	81.55	90.29	85.40	90.12	90.00
POPSH [46]	79.99	92.88	95.02	96.03	96.26	79.80	79.80	94.21	95.04	95.36	79.57	93.24	94.30	95.08	95.33
OLGH [45]	73.57	91.29	95.00	95.85	95.82	72.73	72.73	94.19	94.92	95.07	72.68	90.27	94.21	94.94	95.23
SASH [34]	41.11	79.17	85.80	87.48	89.23	44.62	85.40	88.98	90.12	91.73	44.15	84.96	89.00	90.32	91.46
LBSE[21]	67.78	94.73	95.78	96.22	96.52	67.03	93.85	94.92	95.46	95.64	67.07	93.90	94.91	95.47	95.62
Our SRSH	**84.80**	**96.06**	**96.01**	**97.00**	**96.92**	**83.91**	**95.36**	**95.50**	**96.20**	**96.00**	**83.87**	**95.36**	**95.48**	**96.20**	**96.00**

Table 3: The retrieval performance results (%) with different bits on the NUS-WIDE dataset. The best values are shown in bold.

Method	mAP					Precision@5000					NDCG@100				
	5	8	16	32	64	5	8	16	32	64	5	8	16	32	64
SDH [31]	30.45	30.52	30.45	31.42	31.42	31.68	31.68	31.63	34.25	34.25	27.73	27.73	27.82	28.45	28.45
FSDH [9]	31.45	31.44	31.44	31.45	31.45	36.35	36.35	31.64	36.35	36.36	27.79	27.73	27.72	27.73	27.95
FSSH [25]	48.65	49.14	58.35	60.06	61.11	48.96	52.29	60.53	62.34	64.34	38.70	42.89	47.43	51.70	52.80
SSLH [23]	38.84	40.66	41.93	42.23	43.32	44.74	46.12	47.76	48.18	49.19	34.44	35.10	36.51	37.14	38.29
RSLH [22]	38.78	43.17	41.23	48.47	49.51	34.49	38.26	46.42	42.49	41.74	29.39	32.51	35.22	36.86	37.87
SCDH [5]	46.05	46.38	45.32	47.04	47.75	42.23	42.02	39.71	41.34	41.12	32.81	33.07	31.97	33.48	33.49
POPSH [46]	51.91	54.95	57.46	60.80	61.14	52.42	42.37	61.81	48.02	65.12	39.05	42.37	45.81	**48.02**	**50.19**
OLGH [45]	48.50	49.47	53.64	56.80	55.67	42.14	41.83	47.54	52.47	50.16	35.76	35.30	38.60	35.18	36.72
SASH [34]	34.13	34.70	33.52	36.09	38.86	37.04	37.80	37.34	42.06	45.21	28.20	28.84	28.51	31.86	34.47
LBSE[21]	54.81	56.56	60.97	62.29	**64.16**	53.90	57.38	**62.58**	65.72	68.02	36.99	41.11	45.11	47.01	47.89
Our SRSH	**57.95**	**60.91**	**62.73**	**64.84**	64.02	**54.65**	**58.75**	62.02	**66.92**	**68.65**	**40.52**	**45.78**	**46.98**	45.25	46.96

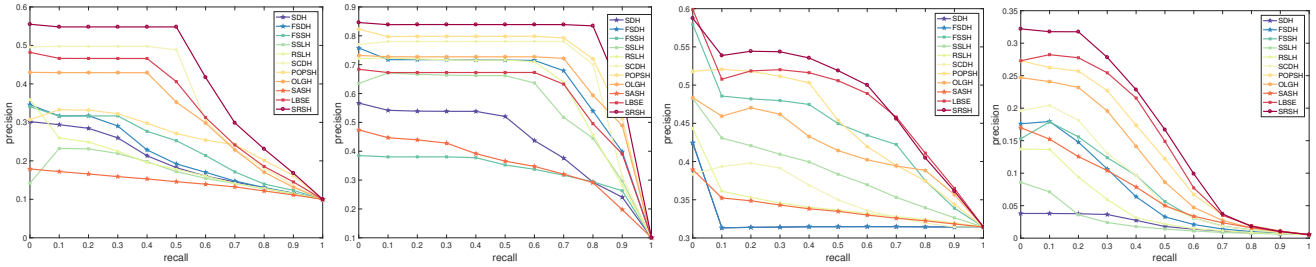
more improvement compared with these baselines. Specifically, since SRSH benefits from the three-step coupled refinement strategy to mitigate of information loss, we can get the performance improvement of 5.12%, 2.32%, 1.47%, and 3.30% on all datasets, respectively. When the number of samples is large (such as the NUS-WIDE), the data becomes more complex. Hence, image retrieval becomes more challenging.

4.7 Precision-Recall

Precision-Recall curves for short-length hash codes are plotted in Fig.2. According these experimental results, we can observe that the performance of SRSH over the best competitor are significant. These results show that our SRSH can extract critical features from high-dimensional image data to mitigate information loss. Hence,

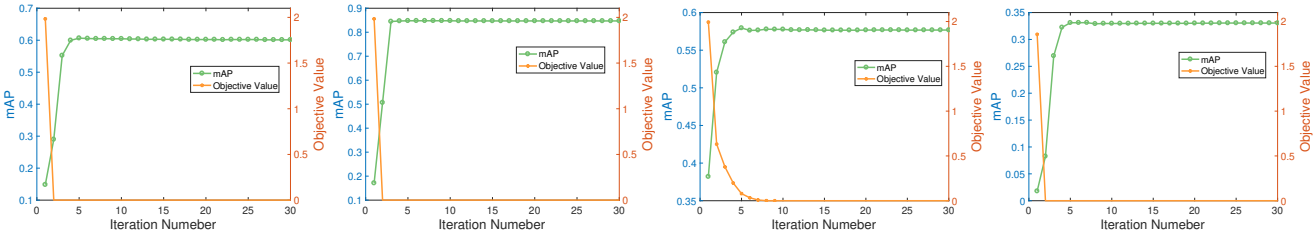
Table 4: The retrieval performance results (%) with different bits on the Caltech-256 dataset. The best values are shown in bold.

Method	mAP					Precision@50					NDCG@100				
	8	16	32	48	64	8	16	32	48	64	8	16	32	48	64
SDH [31]	3.53	29.21	35.91	41.00	44.76	3.54	39.90	49.31	54.67	58.44	3.64	34.48	42.31	47.31	51.04
FSDH [9]	14.93	29.14	39.60	44.69	47.58	18.72	40.10	53.48	58.38	61.02	18.90	35.05	46.13	50.89	53.70
FSSH [25]	19.69	40.53	52.96	58.58	63.09	18.74	38.37	50.51	56.01	60.20	18.63	38.61	50.72	56.20	60.41
SSLH [23]	7.25	9.29	9.73	10.25	12.90	7.87	10.56	11.32	12.68	15.69	7.87	10.59	11.15	12.02	15.03
RSLH [22]	14.25	20.89	43.44	50.33	53.26	10.94	30.69	30.25	35.32	38.33	14.20	26.34	36.76	42.39	45.42
SCDH [5]	25.92	36.88	55.65	60.93	63.55	24.54	34.72	53.44	58.28	60.96	24.62	34.75	53.54	58.48	61.13
POPSH [46]	27.86	51.86	62.25	68.67	69.80	26.24	49.80	59.91	66.17	66.70	26.67	50.06	60.04	66.27	66.89
OLGH [45]	25.69	47.77	60.12	62.67	62.00	24.09	45.80	58.58	62.19	62.87	24.53	45.99	58.21	61.37	61.22
SASH [34]	12.52	21.31	26.60	31.38	33.79	15.05	27.72	35.25	41.62	44.63	14.72	25.54	31.40	36.51	39.34
LBSE[21]	30.97	52.88	65.27	70.67	71.99	29.72	50.90	63.31	68.47	69.40	29.16	51.96	63.48	68.66	69.63
Our SRSH	**33.45**	**54.35**	**66.52**	**71.42**	**73.20**	**32.05**	**52.63**	**64.39**	**69.41**	**71.00**	**32.46**	**52.81**	**64.65**	**69.52**	**71.18**



(a) CIFAR-10 (b) MNIST (c) NUS-WIDE (d) Caltech256

Figure 2: The precision-recall curves of all methods for short-length hash codes on the four datasets



(a) CIFAR-10 (b) MNIST (c) NUS-WIDE (d) Caltech256

Figure 3: Convergence curves and mAP curves of SRSH with short-length hash codes on four datasets.

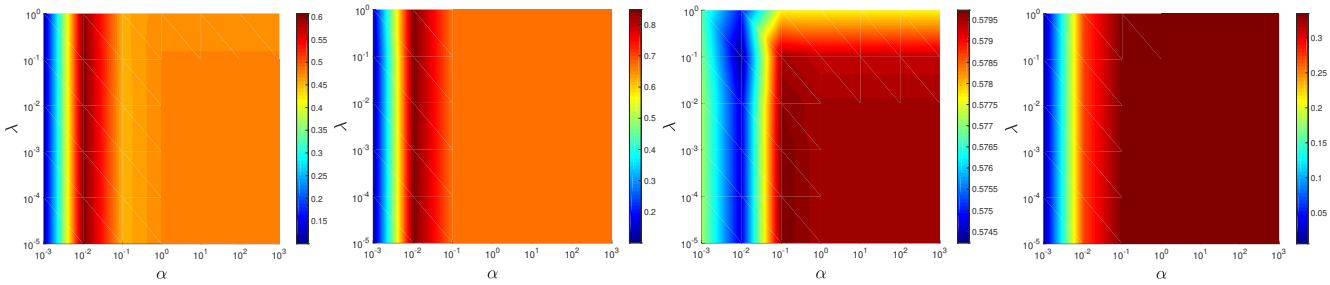
the proposed SRSH can obtain the better performance under short-length hash codes. Generally, according to precision-recall curves, the effectiveness of SRSH is further demonstrated, which is consistent with other three evaluation protocols.

4.8 Training Time

The time cost of image hashing main depends on the training step. On the NUS-WIDE, we record the training time of these baselines with different hash lengths in table 5. From the results, the proposed SRSH obtains acceptable training time that is scalable for large-scale data. Especially, for short-length hash codes, SRSH gets the second lowest training time. With the hash length increases, the training time also grows. Clearly, short-length hash codes can reduce more time cost. For short hashing methods (*i.e.*, SSLH and RSLH), they need more training time due to learning hash codes bit by bit, and greatly improves dramatically as the length of bits increases. Since the similarity preserving hashing methods consume more training time to make similarity approximation, SCDH and SASH need more training time. Therefore, the training time further demonstrates the effectiveness of SRSH. In additional, SRSH also has very competitive retrieval performance. Clearly, SRSH enjoys more potential to handle large-scale image retrieval tasks in practical application.

4.9 Convergence Experiments

To show the convergence property of our SRSH, we draw the convergence curves and mAP curves with short-length hash codes



(a) CIFAR-10 (b) MNIST (c) NUS-WIDE (d) Caltech256

Figure 4: The sensitivities of hyper-parameters on mAP results with short-length hash codes on four datasets.

Table 5: The training time (seconds) with different bits on the NUS-WIDE dataset.

Method	5	8	16	32	64
SDH [31]	15.35	16.76	22.46	46.91	142.05
FSDH [9]	14.59	14.78	15.08	16.21	18.35
FSSH [25]	8.52	8.61	8.72	9.17	9.89
SSLH [23]	10.03	13.85	39.04	87.79	262.82
RSLH [22]	108.23	126.69	160.34	164.12	233.57
SCDH [5]	19.87	20.34	22.67	26.77	79.24
POPSH [46]	2.51	2.72	2.78	2.86	5.97
OLGH [45]	6.76	7.42	7.54	8.16	9.62
SASH [34]	3469.75	3558.76	4501.92	6699.52	7724.92
LBSE[21]	24.1	25.97	16.7	20.24	24.75
Our SRSH	6.96	6.99	7.47	14.4	47.45

Table 6: Ablation results (mAP: %) on the CIFAR-10.

Methods	4	8	16	32	64
SRSH-1	60.65	64.93	68.11	72.94	72.31
SRSH-2	45.36	63.19	**69.39**	70.82	72.67
SRSH-3	50.34	63.78	45.56	25.05	18.11
SRSH-4	28.72	31.98	41.12	45.23	46.84
SRSH	**60.65**	**64.93**	68.11	**73.24**	**73.47**

on the four datasets. As shown in Fig.3, the objective values are monotonically reduced close to a fixed value after a few iterations (about five iterations). Moreover, we can find that the mAP curves gradually increase until the mAP results become stable. Hence, we can set the number of iterations as five.

4.10 Parameter Sensitivity Analysis

Our SRSH involves two trade-off parameters to be set properly, *i.e.*, α and λ. To observe the stability, we perform grid search experiments with different parameter settings to obtain the optimum performance for short-length hash codes. Specifically, we first vary parameter α in $[10^{-3}, 10^{-2}, \cdots, 10^{3}]$ and λ in $[10^{-5}, 10^{-4}, \cdots, 1]$, respectively. As shown in Fig.4, we can observe our SRSH works well within a wide range of α and λ. Hence, the results indicate our SRSH is insensitive to the two trade-off parameters.

4.11 Ablation Analysis

We perform ablation analysis on the CIFAR-10 to show the effect of the three-step coupled refinement strategy. Our method has four variations, *i.e.*, SRSH-1, SRSH-2, SRSH-3, and SRSH-4. Thereinto, SRSH-1 preserves three-layer hash function and removes the regularization term about the hash function. SRSH-2 and SRSH-3 preserves two-layer hash function. They remove the third and second level hash function, respectively. SRSH-4 represents the model with only single hash function. The mAP scores of SRSH and its variations are shown in table 6. We can see that SRSH obtains the best mAP with different lengths of hash codes, especially, the short-length hash codes. In general, the results shows the three-step coupled strategy with constraints from loose to strict can stepwise learn fine hash codes to mitigate large information loss caused by dimensional avalanches.

5 CONCLUSION

In this paper, we propose a novel stepwise refinement short hashing (SRSH) for image retrieval, which can refine critical features from high-dimensional image data to learn discriminative short-length hash codes. For the first time, in order to alleviate the information loss caused by the avalanche of dimension truncation, we propose a three-step coupled refinement strategy to identify stronger features by learning a richer set of projections and shrink the discrete solution space to obtain coarse and fine hash codes. To inherit intrinsic semantic structure, we further preserve the similarity of original space between coarse and fine hash codes. Comprehensive experimental results on four large-scale datasets demonstrate the effectiveness, superiority, and efficiency of our SRSH.

ACKNOWLEDGMENTS

This research was supported by the National Natural Science Foundation of China (Grant no. U19A2078, 61976182, 62106209), the Sichuan Science and Technology Planning Project (Grant no. 2023YFQ0020, 2023YFG0033, 2023ZHCG0016, 2022YFQ0014, 2022YFH0021), the Chengdu Science and Technology Project (Grant no. 2023-XT00-00004-GX), the Fundamental Research Funds for the Central Universities (Grant no. SCU2022JG002), and the Open Research Fund of Anhui Province Key Laboratory of Machine Vision Inspection (Grant no. KLMVI-2023-HIT-08).

REFERENCES

[1] Yuan Cao, Sheng Chen, Jie Gui, Heng Qi, Zhiyang Li, and Chao Liu. 2021. Hash learning with variable quantization for large-scale retrieval. *IEEE Transactions on Circuits and Systems for Video Technology* 32, 5 (2021), 2624–2637.

[2] Mansheng Chen, Tuo Liu, Chang-Dong Wang, Dong Huang, and Jian-Huang Lai. 2022. Adaptively-weighted Integral Space for Fast Multiview Clustering. In *The 30th ACM International Conference on Multimedia*. ACM, 3774–3782.

[3] Mansheng Chen, Chang-Dong Wang, Dong Huang, Jian-Huang Lai, and Philip S. Yu. 2022. Efficient Orthogonal Multi-view Subspace Clustering. In *The 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. ACM, 127–135.

[4] Man-Sheng Chen, Chang-Dong Wang, and Jian-Huang Lai. 2022. Low-rank Tensor Based Proximity Learning for Multi-view Clustering. *IEEE Transactions on Knowledge and Data Engineering* (2022).

[5] Yong Chen, Zhibao Tian, Hui Zhang, Jun Wang, and Dell Zhang. 2020. Strongly constrained discrete hashing. *IEEE Transactions on Image Processing* 29 (2020), 3596–3611.

[6] Hui Cui, Lei Zhu, Jingjing Li, Zhiyong Cheng, and Zheng Zhang. 2021. Two-pronged strategy: Lightweight augmented graph network hashing for scalable image retrieval. In *Proceedings of the 29th ACM International Conference on Multimedia*. 1432–1440.

[7] Hui Cui, Lei Zhu, Jingjing Li, Yang Yang, and Liqiang Nie. 2019. Scalable deep hashing for large-scale social image retrieval. *IEEE Transactions on image processing* 29 (2019), 1271–1284.

[8] Thanh-Toan Do, Khoa Le, Tuan Hoang, Huu Le, Tam V Nguyen, and Ngai-Man Cheung. 2019. Simultaneous feature aggregating and hashing for compact binary code learning. *IEEE Transactions on Image Processing* 28, 10 (2019), 4954–4969.

[9] Jie Gui, Tongliang Liu, Zhenan Sun, Dacheng Tao, and Tieniu Tan. 2017. Fast supervised discrete hashing. *IEEE transactions on pattern analysis and machine intelligence* 40, 2 (2017), 490–496.

[10] Rundong He, Zhongyi Han, Xiankai Lu, and Yilong Yin. 2022. RONF: reliable outlier synthesis under noisy feature space for out-of-distribution detection. In *Proceedings of the 30th ACM International Conference on Multimedia*. 4242–4251.

[11] Rundong He, Zhongyi Han, Xiankai Lu, and Yilong Yin. 2022. Safe-student for safe deep semi-supervised learning with unseen-class unlabeled data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 14585–14594.

[12] Rundong He, Zhongyi Han, Yang Yang, and Yilong Yin. 2022. Not all parameters should be treated equally: Deep safe semi-supervised learning under class distribution mismatch. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 36. 6874–6883.

[13] Shiyuan He, Bokun Wang, Zheng Wang, Yang Yang, Fumin Shen, Zi Huang, and Heng Tao Shen. 2019. Bidirectional discrete matrix factorization hashing for image search. *IEEE transactions on cybernetics* 50, 9 (2019), 4157–4168.

[14] Di Hu, Feiping Nie, and Xuelong Li. 2018. Discrete spectral hashing for efficient similarity retrieval. *IEEE Transactions on Image Processing* 28, 3 (2018), 1080–1091.

[15] Peng Hu, Xi Peng, Hongyuan Zhu, Jie Lin, Liangli Zhen, and Dezhong Peng. 2021. Joint Versus Independent Multiview Hashing for Cross-View Retrieval. *IEEE Transactions on Cybernetics* 51, 10 (2021), 4982–4993. https://doi.org/10.1109/TCYB.2020.3027614

[16] Peng Hu, Xu Wang, Liangli Zhen, and Dezhong Peng. 2019. Separated variational hashing networks for cross-modal retrieval. In *Proceedings of the 27th ACM International Conference on Multimedia*. 1721–1729.

[17] Peng Hu, Hongyuan Zhu, Jie Lin, Dezhong Peng, Yin-Ping Zhao, and Xi Peng. 2023. Unsupervised Contrastive Cross-Modal Hashing. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45, 3 (2023), 3877–3889. https://doi.org/10.1109/TPAMI.2022.3177356

[18] Huaxiong Li, Chao Zhang, Xiuyi Jia, Yang Gao, and Chunlin Chen. 2023. Adaptive Label Correlation Based Asymmetric Discrete Hashing for Cross-Modal Retrieval. *IEEE Transactions on Knowledge and Data Engineering* 35, 2 (2023), 1185–1199. https://doi.org/10.1109/TKDE.2021.3102119

[19] Mingbao Lin, Rongrong Ji, Hong Liu, and Yongjian Wu. 2018. Supervised online hashing via hadamard codebook learning. In *Proceedings of the 26th ACM international conference on Multimedia*. 1635–1643.

[20] Hong Liu, Rongrong Ji, Jingdong Wang, and Chunhua Shen. 2018. Ordinal constraint binary coding for approximate nearest neighbor search. *IEEE transactions on pattern analysis and machine intelligence* 41, 4 (2018), 941–955.

[21] Xingbo Liu, Xiao Kang, Xiushan Nie, Jie Guo, Shaohua Wang, and Yilong Yin. 2022. Learning Binary Semantic Embedding for Large-Scale Breast Histology Image Analysis. *IEEE Journal of Biomedical and Health Informatics* 26, 7 (2022), 3240–3250. https://doi.org/10.1109/JBHI.2022.3161341

[22] Xingbo Liu, Xiushan Nie, Qi Dai, Yupan Huang, Li Lian, and Yilong Yin. 2020. Reinforced short-length hashing. *IEEE Transactions on Circuits and Systems for Video Technology* 31, 9 (2020), 3655–3668.

[23] Xingbo Liu, Xiushan Nie, Quan Zhou, Xiaoming Xi, Lei Zhu, and Yilong Yin. 2019. Supervised Short-Length Hashing.. In *IJCAI*. 3031–3037.

[24] Kaiyi Luo, Chao Zhang, Huaxiong Li, Xiuyi Jia, and Chunlin Chen. 2023. Adaptive Marginalized Semantic Hashing for Unpaired Cross-Modal Retrieval. *IEEE Transactions on Multimedia* (2023), 1–14. https://doi.org/10.1109/TMM.2023.3245400

[25] Xin Luo, Liqiang Nie, Xiangnan He, Ye Wu, Zhen-Duo Chen, and Xin-Shun Xu. 2018. Fast scalable supervised hashing. In *The 41st international ACM SIGIR conference on research & development in information retrieval*. 735–744.

[26] Xiao Luo, Haixin Wang, Daqing Wu, Chong Chen, Minghua Deng, Jianqiang Huang, and Xian-Sheng Hua. 2023. A survey on deep hashing methods. *ACM Transactions on Knowledge Discovery from Data* 17, 1 (2023), 1–50.

[27] Xin Luo, Ye Wu, and Xin-Shun Xu. 2018. Scalable supervised discrete hashing for large-scale search. In *Proceedings of the 2018 World Wide Web Conference*. 1603–1612.

[28] Xin Luo, Peng-Fei Zhang, Zi Huang, Liqiang Nie, and Xin-Shun Xu. 2019. Discrete hashing with multiple supervision. *IEEE Transactions on Image Processing* 28, 6 (2019), 2962–2975.

[29] Yadan Luo, Zi Huang, Yang Li, Fumin Shen, Yang Yang, and Peng Cui. 2020. Collaborative learning for extremely low bit asymmetric hashing. *IEEE Transactions on Knowledge and Data Engineering* 33, 12 (2020), 3675–3685.

[30] Xiushan Nie, Xingbo Liu, Jie Guo, Letian Wang, and Yilong Yin. 2022. Supervised Discrete Multiple-Length Hashing for Image Retrieval. *IEEE Transactions on Big Data* 01 (2022), 1–1.

[31] Fumin Shen, Chunhua Shen, Wei Liu, and Heng Tao Shen. 2015. Supervised discrete hashing. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Vol. 7. IEEE, 37–45.

[32] Fumin Shen, Yan Xu, Li Liu, Yang Yang, Zi Huang, and Heng Tao Shen. 2018. Unsupervised deep hashing with similarity-adaptive and discrete optimization. *IEEE transactions on pattern analysis and machine intelligence* 40, 12 (2018), 3034–3044.

[33] Dan Shi, Lei Zhu, Jingjing Li, Zheng Zhang, and Xiaojun Chang. 2023. Unsupervised Adaptive Feature Selection With Binary Hashing. *IEEE Transactions on Image Processing* 32 (2023), 838–853. https://doi.org/10.1109/TIP.2023.3234497

[34] Yang Shi, Xiushan Nie, Xingbo Liu, Li Zou, and Yilong Yin. 2022. Supervised Adaptive Similarity Matrix Hashing. *IEEE Transactions on Image Processing* 31 (2022), 2755–2766.

[35] Yuan Sun, Dezhong Peng, Haixiao Huang, and Zhenwen Ren. 2022. Feature and Semantic Views Consensus Hashing for Image Set Classification. In *Proceedings of the 30th ACM International Conference on Multimedia*. 2097–2105.

[36] Yuan Sun, Xu Wang, Dezhong Peng, Zhenwen Ren, and Xiaobo Shen. 2023. Hierarchical Hashing Learning for Image Set Classification. *IEEE Transactions on Image Processing* 32 (2023), 1732–1744.

[37] Xing Tian, Wing W. Y. Ng, and Huihui Xu. 2022. Deep Incremental Hashing for Semantic Image Retrieval with Concept Drift. *IEEE Transactions on Big Data* (2022), 1–13. https://doi.org/10.1109/TBDATA.2022.3233457

[38] Jingdong Wang, Ting Zhang, Nicu Sebe, Heng Tao Shen, et al. 2017. A survey on learning to hash. *IEEE transactions on pattern analysis and machine intelligence* 40, 4 (2017), 769–790.

[39] Xiangxi Xu, Zhi-Hui Lai, and Yu dong Chen. 2020. Relaxed Locality Preserving Supervised Discrete Hashing. *IEEE Transactions on Big Data* 01 (2020), 1–1.

[40] Xihong Yang, Xiaochang Hu, Sihang Zhou, Xinwang Liu, and En Zhu. 2022. Interpolation-based contrastive learning for few-label semi-supervised learning. *IEEE Transactions on Neural Networks and Learning Systems* (2022).

[41] Xihong Yang, Yue Liu, Sihang Zhou, Siwei Wang, Wenxuan Tu, Qun Zheng, Xinwang Liu, Liming Fang, and En Zhu. 2023. Cluster-guided Contrastive Graph Clustering Network. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 37. 10834–10842.

[42] Li Yuan, Tao Wang, Xiaopeng Zhang, Francis EH Tay, Zequn Jie, Wei Liu, and Jiashi Feng. 2020. Central similarity quantization for efficient image and video retrieval. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 3083–3092.

[43] Chao Zhang, Huaxiong Li, Yang Gao, and Chunlin Chen. 2023. Weakly-Supervised Enhanced Semantic-Aware Hashing for Cross-Modal Retrieval. *IEEE Transactions on Knowledge and Data Engineering* 35, 6 (2023), 6475–6488. https://doi.org/10.1109/TKDE.2022.3172216

[44] Zheng Zhang, Zhihui Lai, Zi Huang, Wai Keung Wong, Guo-Sen Xie, Li Liu, and Ling Shao. 2019. Scalable supervised asymmetric hashing with semantic and latent factor embedding. *IEEE Transactions on Image Processing* 28, 10 (2019), 4803–4818.

[45] Zheng Zhang and Chi-Man Pun. 2022. Learning ordinal constraint binary codes for fast similarity search. *Information Processing & Management* 59, 3 (2022), 102919.

[46] Zheng Zhang, Xiaofeng Zhu, Guangming Lu, and Yudong Zhang. 2021. Probability ordinal-preserving semantic hashing for large-scale image retrieval. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 15, 3 (2021), 1–22.

[47] Chaoqun Zheng, Lei Zhu, Zhiyong Cheng, Jingjing Li, and An-An Liu. 2020. Adaptive partial multi-view hashing for efficient social image retrieval. *IEEE Transactions on Multimedia* 23 (2020), 4079–4092.

[48] Lei Zhu, Zi Huang, Zhihui Li, Liang Xie, and Heng Tao Shen. 2018. Exploring auxiliary context: discrete semantic transfer hashing for scalable image retrieval. *IEEE transactions on neural networks and learning systems* 29, 11 (2018), 5264–5276.