第 34 卷第 6 期
2017 年 6 月

控 制 理 论 与 应 用
Control Theory & Applications

Vol. 34 No. 6
Jun. 2017

DOI: 10.7641/CTA.2017.60608

# 编码选择哈希算法解决大规模图像检索问题

田 星, 陆筱怡, 吴永贤[†], 黄家健
(华南理工大学 计算机科学与工程学院, 广东 广州 510006)

**摘要:** 哈希算法已被广泛应用于解决大规模图像检索的问题. 在已有的哈希算法中, 无监督哈希算法因为不需要数据库中图片的语义信息而被广泛应用. 平移不变核局部敏感哈希(SKLSH)算法就是一种较为代表性的无监督哈希算法. 该算法随机的产生哈希函数, 并没有考虑所产生的哈希函数的具体检索效果. 因此, SKLSH算法可能产生一些检索效果表现较差的哈希函数. 在本文中, 提出了编码选择哈希算法(BSH). BSH算法根据SKLSH算法产生的哈希函数的具体检索效果来进行挑选. 挑选的标准主要根据哈希函数在3个方面的表现: 相似性符合度, 信息包含量, 和编码独立性. 然后, BSH算法还使用了一种基于贪心的选择方法来找到哈希函数的最优组合. BSH算法和其他代表性的哈希算法在两个真实图像库上进行了检索效果的对比实验. 实验结果表明, 相比于最初的SKLSH算法和其他哈希算法, BSH算法在检索准确度上有着明显的提高.

**关键词:** 无监督哈希算法; 编码选择; 大规模图像检索
**中图分类号:** TP273 **文献标识码:** A

# Bit selection hashing for large scale image retrieval

TIAN Xing, LU Xiao-yi, WING W Y NG[†], HUANG Jia-jian
(College of Computer Science and Engineering, South China University of Technology, Guangzhou Guangdong 510006, China)

**Abstract:** Hashing methods have been widely used for solving large scale image retrieval problems. Among existing hashing methods, unsupervised hashing methods are most widely used because they do not require semantic information of images in the database. The shift-invariant kernelized locality-sensitive hashing (SKLSH) is a representative unsupervised hashing method which generates hash functions randomly without considering the performance of each projection. Therefore, weak hash functions yielding low retrieval performances may be generated by the SKLSH. In this work, we propose the bit selection hashing (BSH) which selects hash bits for the SKLSH based on the performance of hash bit projections in three aspects: similarity fitness, information capacity, and code independence. Then, a greedy selection method is applied to find the optimal combination of hash bits for the BSH. Two real world image databases are used to compare the performance of the proposed BSH with other representative hashing methods. Experimental results show that the BSH yields a significant improvement in comparison to the original SKLSH and other hashing methods.

**Key words:** unsupervised hashing method; bit selection; large scale image retrieval

## 1 Introduction

Owing to the popularization of social websites, image sharing websites and convenient digital image taking devices, a huge volume of images is uploaded to the Internet in every hour. Therefore, fast image retrieval methods are important to efficiently utilizing the huge volume of image data on the Internet. Tree-based methods[1–2] are proposed for accurate image searches. However, tree-based methods are space expensive, especially, when the amount and the dimensions of images are large. In the worst case, the searching performance is degraded to a linear scan and the space for storing the tree structure could be even larger than the original image data[3]. Due to these disadvantages, tree-based retrieval methods are not well adaptive to the current big data environment on the Internet.

In large scale image retrieval problems, exact similarity computations are too time consuming and not necessary. Owing to the nature of big data, a set of approximate nearest neighbors provides a good enough retrieval results and may be no worse than those found by exact similarity. Hashing methods find approximate nearest neighbor points using sub-linear time complexity and linear space complexity. Hashing methods firstly find $B$ hashing functions (projections) and map the data from the high dimensional real-valued feature space to a low dimensional binary space. Each image is represented by a binary hash code. Each hash func-

Received 12 August 2016; accepted 2 March 2017.
[†]Corresponding author. E-mail: wingng@ieee.org.
Recommended by Associate Editor YU Zhu-liang.
Supported by National Natural Science Foundation of China (61272201, 61572201) and National Undergraduate Innovative and Entrepreneurial Training Program (201510561072).

tion divides the original feature space into two parts and these two parts are represented by 0 and 1, respectively. With $B$ hash functions, the original feature space is at most being divided into $2B$ buckets by different hash codes. Each image in the database is represented by a hash code (instead of the original image data) and only the hash code is used during the query. Images in the same bucket (the same hash code) or neighboring buckets are considered to be similar. In this way, the similarity computation between two images is converted to a fast Hamming distance calculation using fast $XOR$ operations.

Many hashing methods have been proposed to solve image retrieval problems. According to whether semantic labels information of images is used for training, current hashing methods are generally divided into three categories: unsupervised, semi-supervised, and supervised hashing methods. Unsupervised hashing methods train hash functions without using the semantic information provided by the database. These methods are always time efficient but yield relatively low retrieval accuracies. Supervised hashing methods use semantic information for training hash functions which are computationally expensive but achieve better accuracies in comparison to unsupervised hashing methods. Semi-supervised hashing methods require partly labeled database for training which take benefits from both supervised and unsupervised hashing methods.

In the big data environment on the Internet, it is cost-expensive to require images for training to be labeled. Therefore, unsupervised hashing methods are more suitable for large scale image retrieval on the Internet. The locality sensitive hashing (LSH) method[4–6] is one of the most famous unsupervised hashing methods which creates hash functions randomly without considering the data distribution and semantic information of the training image set. The locality sensitive binary codes form shift-invariant kernels (SKLSH)[7] is an important improvement to the LSH. The SKLSH uses random Fourier features of the dataset to build hash functions. As projections of the SKLSH are selected randomly, some projections may have little distinguishability for the dataset and may even highly similar. Due to the aforementioned problem, the projection selection hashing (PSH)[8] applies the relief feature selection method[9] to select better hash codes and corresponding projections. However, the PSH treats the projection selection problems as a feature selection problem for pattern classification instead of image retrieval. The PSH evaluates the quality of a projection based on whether it could distinguish the similar and dissimilar images only without regarding the information independence between projections and the information capacity of each projection. For hashing methods, we hope the information in each projection is maximized and hash functions are independent with each other for less information redundancy. In this paper, a new bit selection hashing (BSH) based on the SKLSH is proposed by selecting the optimal bit projection combination for large scale image retrieval problems.

Major contributions of this paper are summarized as follows:

· Propose a similarity fitness evaluation method based on the Hamming distance threshold to improve the unsupervised retrieval performance.

· Propose a performance evaluation method for hash functions by combining the similarity fitness, the information capacity, and the code independence of hash functions.

· Propose the bit selection hashing based on the selection of unsupervised hash functions to form the optimal combinations of hash functions in greedy manner.

· The proposed BSH enjoys both benefits of high performance of the SKLSH and the small storage by selecting informative bits only.

In Section 2, related works on existing hashing methods are introduced in detail. The proposed BSH method is proposed in Section 3. Experimental results of the BSH and comparison with current hashing methods are given in Section 4. Section 5 concludes this work and gives ideas of feature works.

## 2 Related works

As aforementioned, hashing methods can be divided into three categories: unsupervised, semi-supervised, and supervised hashing methods. Supervised hashing methods such as the LDA[10], the LSH with learned metric[11], and the online hashing[12], usually achieve higher retrieval accuracies but also require a fully labeled train dataset which is impractical for real data environment on the Internet. The LDA hashing method applies linear discriminant analysis to the dataset to find projection functions which minimize distances between similar data pairs and maximize distances between dissimilar data pairs in the Hamming space. The LSH with learned metric method applies the LSH in a metric space learned from the semantic information instead of the original feature space. The online hash updates hashing functions based on the new semantic information in an online manner.

Semi-supervised hashing methods use partially labeled dataset to train hash functions, for instances, the complementary hashing[13], the DCH[14], the SPLH[15], the BSPLH[3], and the SCEM–SSH[16]. The SPLH[15] trains each hash function by correcting the error made by the previous hash function while the BSPLH[3] trains each hash function by correcting the error made by all the previous hash functions. The complementary hashing[13] and the DCH[14] are both multi-hash extensions of the SPLH. The SCEM–SSH[16] maximizes the conditional entropy of a hash function with respect to all previous hash functions.

There are two major streams of unsupervised hashing methods: data dependent and data independent methods. Data dependent methods create hashing functions based on the data distribution e.g. the ITQ[17], the spectral embedded hashing[18], the two-phase hashing[19], and the ACH[20]. The ITQ hashing method[17] applies the Principle Component Analysis (PCA) to find orthogonal hash projections based on the maximum variance directions of the data. The projection is then rotated to minimize the quantization loss between rotated data and its nearest vertex of a unit binary hypercube. The spectral embedded hashing introduces a new regularizer to the objective function of the spectral hashing[21] to control the mismatch between the resultant Hamming embedding[18]. The two-phase hashing[19] firstly maps the data from the original feature space onto a high dimensional real-valued space to preserve pairwise similarity using the SKLSH projection. Then, data is mapped from the high dimensional real-valued space onto a low dimensional Hamming space via a minimization of the reconstruction error. The ACH[20] employs a short hash code to store images in the database for storage efficiency and a long hash code for query and similarity comparison for high retrieval precision.

The LSH[4] and the SKLSH[7] are the two most representative data independent hashing methods. The LSH randomly creates hash function to split the feature space into hash buckets. It is proved that the probability of two images falling into the same hash bucket is directly proportional to the similarity between these images[4].

Most hashing methods project images from the high dimensional feature space to a low dimensional Hamming space. In the SKLSH, hash functions are found randomly via a random Fourier features (RFF) method. The RFF ($\varPhi_{\omega,b}(x)$) is defined as

$$\varPhi_{\omega,b}(x) = \sqrt{2}\cos(\omega \cdot x + b), \tag{1}$$

where $x \in R^D$, $\omega \sim P_{\rm K}$ and $b \sim$ Unif$[0, 2\pi]$ , and $P_{\rm K}$ denotes the probability measure of the dataset. Then, the SKLSH performs a random binary quantization to the RFFs using a random threshold $t \sim$ Unif$[-1, 1]$ and defines the hash function as follows:

$$h_{\omega,b,t}(x) = \frac{1}{2}[1 + \operatorname{sgn}(\varPhi_{\omega,b}(x) + t)], \tag{2}$$

where $\operatorname{sgn}(u) = 1$ if $u \geqslant 0$ and $-1$ otherwise.

The hash functions of the SKLSH are randomly selected from the RFF, therefore the quality of each hash function does not necessary to yield good performance. Similar to the LSH, the SKLSH guarantees that the accuracy of the similarity is directly proportional to the number of hash functions (bits). However, a huge number of hash bits may reduce the scalability of the SKLSH. Moreover, each hash function is expected to divide the original feature space into two parts such that similar images are divided into the same side of the projection hyperplane while dissimilar images are divided into different sides.

The projection selection hashing (PSH) treats each hash function as a feature of a classification problem and utilizes the relief feature selection method to select hash functions[8]. The relief method randomly selects $m$ samples ($R$). For each sample in $R$, it finds $k$ nearest samples (nearest hit set $H$) among samples similar to $R$ and $k$ nearest dissimilar samples (nearest miss set $M$). The weight ($W[F]$) of all features are set to zero for initialization and then a weight is updated to evaluate the quality of a feature as follows:

$$W[F] = W[F] - \sum_{j=1}^{k} \operatorname{diff}(F, R_i, H_j)/(mk) + \sum_{j=1}^{k} \operatorname{diff}(F, R_i, M_j)/(mk), \tag{3}$$

where $R_i$ and $W[F]$ and $\operatorname{diff}(F, \cdot, \cdot)$ denote the $i$th sample in $R$, the weight for the feature $F$ and the value difference in the feature $F$ between the selected sample in $R$ and its nearest hit or misses. For two samples $s_i$ and $s_j$, $\operatorname{diff}(F, s_i, s_j) = 0$ if these two samples share the same value in feature $F$ and $\operatorname{diff}(F, s_i, s_j) = 1$ otherwise. However unsupervised dataset does not carry label information for samples, samples yielding Hamming distances smaller than a pre-selected threshold is regarded as similar samples and other samples are treated as dissimilar samples. The PSH attempts to find better hash functions for the SKLSH in terms to improve its performance. However, treating the hash function selection problem as a feature selection problem for classification may not be appropriate. PSH method select hash functions just rely on the similarity preservation which is not good enough. The major concern of selecting hash functions for large scale image retrieval problems should rely on retrieval performance and information provided by each hash function. What's more, code independence is also a useful standard to select the best hash functions. A hash bit selection method is proposed based on the similarity preservation and the mutual independence of hash functions[22]. However, it requires a very time consuming Euclidean distance computation for finding the similarity preservation of a hash function which may not be practical for large scale image retrieval problems. Moreover, the mutual independence in [22] is computed immediately by maximizing the sum of mutual information of all selected hash functions which may be misled by a single hash function yielding very large mutual information value with others. Therefore, the proposed BSH method uses a Hamming distance-based method to compute the similarity fitness for better scalability and a greedy search maximizing the worst independent pair of hash functions iteratively in the BSH will be a better solution.

## 3 Bit selection hashing

In order to improve the performance of the SKLSH for large scale image retrieval problems, we propose the BSH to select better hash functions for the SKLSH. The BSH firstly creates a candidate set by generating excessive number of hash functions using the SKLSH. Then, hash functions in the candidate set are weighted for selection. These hash functions are weighted according to its performance in three metrics: similarity fitness($\gamma$), information capacity($\delta$), and code independence($\varphi$). The computations of these three metrics will be introduced in Section 3.1.

The BSH performs a greedy search to select the hash function from a candidate set yielding the largest weight (i.e. max $W_i$) iteratively until a pre-selected number of hash functions ($B$) is reached. The selected hash function is added to the selected set and removed from the candidate set. Hash functions in the selected set are finally used to generate hash codes for images in the database for retrieval.

The first hash function is selected using both the similarity fitness and the information capacity only. The selected hash function set is currently empty, so the code independence cannot be calculated. Such that, the hash function yielding the best retrieval precision and separate samples in the feature spaces most evenly is selected as the first selected hash function of the BSH. Therefore, the weight for the $i$th hash function ($W_i$) is computed as follows:

$$W_i = \gamma_i \times \delta_i. \tag{4}$$

Then, for the 2nd to $B$th hash functions, in addition to its own fitness and information capacity, correlations (or dependences) among different hash functions should be minimized. In other word, when the code independence of one candidate hash function is evaluated, we focus on the relationship between this hash function and its most depending hash function in the selected hash function set. Therefore, the most independent candidate hash function with respect to all selected hash functions is preferred. If the average value of correlations among all hash functions is minimized, it is easily being misled by a single hash function with very large correlation to the candidate hash function. Therefore, the BSH prefers a hash function yielding the largest minimum pairwise independence ($\varphi$) with selected hash functions. The weight for the $i$th hash function is computed as follows:

$$W_i = \gamma_i \times \delta_i \times \min(\varphi_{ij}), \tag{5}$$

where $\varphi_{ij}$ denotes the $\varphi$ between the $i$th candidate hash functions and the $j$th selected hash functions. The flow diagram of the BSH for generating the selected set is shown as Fig.1.

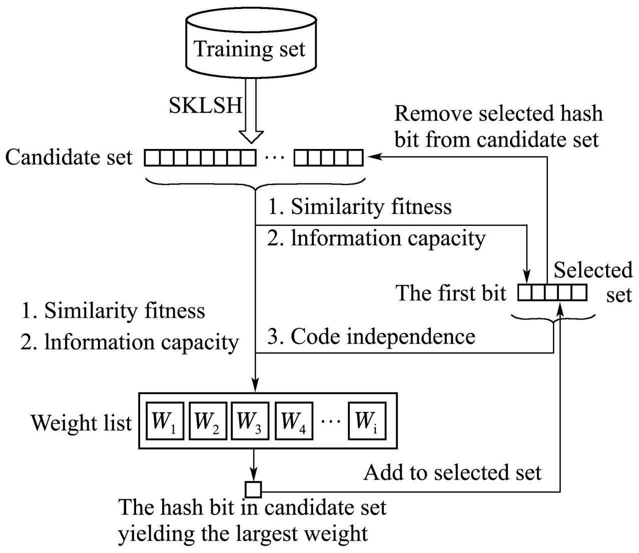


Fig. 1 Flow diagram of the BSH to generate the selected hash bit set

### 3.1 Metric calculation for weighting

#### 3.1.1 Semantic fitness

The similarity fitness ($\gamma$) evaluates the distinguishability of a hash function between similar and dissimilar samples. Generally, a hash function can be regarded as a hyperplane to divide the original data space into two sides. Then, similar image pairs are preferred to be divided into the same side of this hyperplane while dissimilar images are preferred to be divided into different sides. In this way, similar images share the same hash code while dissimilar images have different hash code. Therefore, similarity fitness is used to evaluate this ability of a hash function. Firstly a subset of samples ($X_l \in \mathbb{R}^{m\times d}$) is randomly selected from the original dataset ($X \in \mathbb{R}^{N\times d}$) where $d$, $m$, and $N$ denote the number of dimension of samples, the number of selected samples, and the number of samples in the whole dataset, respectively. Let $S$ be the pairwise similarity matrix of samples in $X_l$. Given a Hamming distance threshold $t$, the similarity between samples $x_j$ and $x_k$ in $X_l$ ($S_{jk}$) is defined as follows:

$$S_{jk} = \begin{cases} +1, & \text{if } x_j \text{ and } x_k \text{ are similar,} \\ -1, & \text{otherwise.} \end{cases} \tag{6}$$

Samples $x_j$ and $x_k$ in $X_l$, are expected to have the same (different) hash value if $S_{jk} = +1$ ($S_{jk} = -1$). If this expectation is fulfilled, samples $x_j$ and $x_k$ are correctly hashed. Therefore, for the $i$th hash function and samples $x_j$ and $x_k$, we define:

$$J_i(x_j, x_k) = \begin{cases} +1, & \text{correctly hashed,} \\ -1, & \text{otherwise.} \end{cases} \tag{7}$$

The fitness of the $i$th hash function among all selected samples is then computed as follows:

$$r_i = \sum_{x_j, x_k \in X_l, j\neq k} J_i(x_j, x_k). \tag{8}$$

In the best case, all sample pairs are correctly hashed, so the upper bound of $r_i$ is $m^2 - m$. Meanwhile, the lower bound in which all samples are incorrectly hashed is $-m^2+m$. Finally, the similarity fitness of the $i$th hash function ($\gamma_i$) is computed as follows:

$$\gamma_i = \frac{r_i - (m - m^2)}{2(m^2 - m)}. \tag{9}$$

### 3.1.2 Information capacity

In information theory, a variable is expected to contain more information if it has a larger entropy value. This idea is introduced into hashing methods training for large scale image retrieval problems [15]. When a hash bit (i.e. hash function) is regarded as a variable, it contains the largest amount of information when the hyperplane of this hash function divides the data space into two halves evenly. We use information capacity ($\delta$), i.e. the entropy of the hash bit, to denote the entropy yielded by a hash projection. The hash values of all images in the database are computed. For the $i$th hash function, $p_i^{(1)}$ and $p_i^{(0)}$ denote the probabilities of hash values of samples being equal to 1 and 0, respectively. Then $\delta_i$ for the $i$th hash function is computed as follows:

$$\delta_i = -p_i^{(1)}\log_2 p_i^{(1)} - p_i^{(0)}\log_2 p_i^{(0)}. \tag{10}$$

### 3.1.3 Code independence

For high code efficiency, hash projections should be independent with each other. Mutual information is used in the BSH to evaluate the code independence ($\varphi$) between a candidate and a selected hash function. Let $v_i \in \{0,1\}$ and $v_j \in \{0,1\}$ be the hash value of the $i$th and the $j$th hash functions, respectively, for a given sample. Then the code independence between the $i$th and the $j$th hash functions ($\varphi_{ij}$) is computed as follows:

$$\varphi_{ij} = 1 - \sum_{y\in\{0,1\}}\sum_{z\in\{0,1\}} p_{ij}^{(y,z)}\log_2\frac{p_{ij}^{(y,z)}}{p_i^{(y)}p_j^{(z)}}, \tag{11}$$

where $p_{ij}^{(y,z)}$ denotes the joint probability of $v_i = y$ and $v_j = z$. If the $i$th and the $j$th hash functions are highly correlated, a large mutual information value is yielded and therefore a small $\varphi_{ij}$ is yielded. The pseudo-code of the BSH is shown in the Algorithm 1.

**Algorithm 1** The BSH.

**Input** The database $X$, the number of bits $B$, additional bits parameter $a$, the threshold $t$ for judging similar images.

**Output** $B$ selected hash functions.

```
1) Train $(B + a)$ hash functions using the SKLSH based on $X$.
2) Calculate the similarity fitness $\gamma$ by threshold $t$ and the information capacity $\delta$ for all hash functions to get their weights using equation (4).
3) Selected set = null set.
4) Remove the hash function yielding the largest weight from the candidate set and add it to the selected set.
5) While the number of hash functions in the selected set is less than $B$.
6) Calculate weights for hash functions in the candidate set using equation (5).
7) Remove the hash function yielding the largest weight from the candidate set and add it to the selected set.
8) End.
```

## 3.2 Time complexity analysis

The time complexity of the training phase of the BSH is $\mathrm{O}(NdL + m^2L + NL^2)$, where $L$ denotes the total number of hash functions trained by the SKLSH for selection, i.e. $L = B + a$. Owing to the fact that $m \ll N$ and $L \ll d$ in general, the time complexity of the BSH training can be simplified to be $\mathrm{O}(NdL)$. In the query phase, the time complexity for computing the hash code of a query is $\mathrm{O}(dB)$. Both $d$ and $B$ are constants and the time complexity for query phase of the BSH is the same to other hashing methods in comparisons. Therefore, the online query phase of the BSH is very first.

## 4 Experimental results

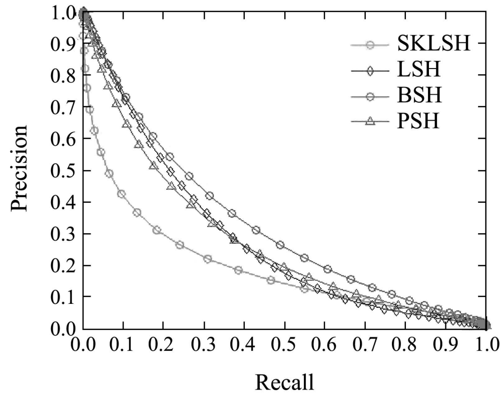
In our experiment, two real world image sets (the CIFAR10 and the MNIST) are used to compare the performance of the BSH and current hashing methods. The CIFAR10 image dataset consists of 6,000 images evenly distributed to 10 classes and each image is represented by a 512–dimensional GIST feature vector. The MNIST is a handwritten digit image set consisting of 70,000 grey-scale images from 10 classes and each image is represented by a 784–dimensional pixels vector. For each dataset, 1,000 samples are randomly selected as the testing set and the rest of samples are used as the training set. Hashing methods are compared using the number of hash bits ($B$) equals 128, 256, and 512, respectively.

The BSH is compared with the SKLSH, the PSH, and the LSH. The BSH is an improvement to the SKLSH while the PSH is another extension of the SKLSH, so they are compared. The LSH serves as the baseline of comparisons. For unsupervised image retrieval problem, images with a Euclidean distance from a sample smaller than a threshold are regarded as similar images of it. The average Euclidean distance of the 50th nearest neighbor of each sample in the database is used as the threshold value[17]. For the BSH, $m$ is set to 1000. Similar to the PSH, the BSH also has two parameters: additional bits ($a$) and threshold ($t$). Given a problem of selecting $B$ hash functions, the set of candidate hash functions consists of $(B + a)$ hash functions for selection. If the Hamming distance between two images is larger than the threshold ($t$), they are dissimilar and they are similar otherwise. This similarity measure is applied in equation (6). Parameters selected for both the BSH and the PSH for each experiment are listed in Table 1. Figs.2 and 3 show the precision-recall curves of different hashing methods on the datasets CIFAR10 and

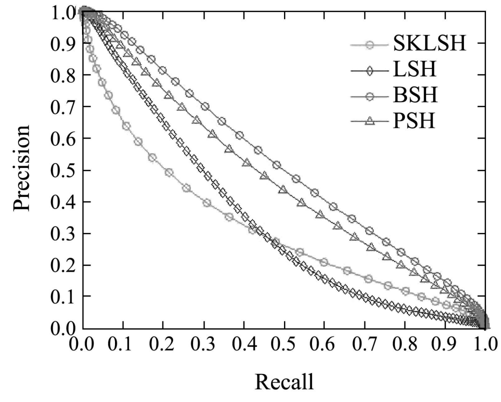
MNIST using 128 bits (a), 256 bits (b) and 512 bits (c), respectively. Experimental results show that the SKLSH improves more significantly in comparison to the LSH when the number of bits ($B$) increases. The PSH only considering the ability of hash functions for distinguishing similar and dissimilar images is not enough to find the optimal hash code combination. In contrast, the BSH combines three metrics to evaluate the performance of each hash function comprehensively. Moreover, in cases shown in Figs.2(a) and 2(b), the PSH does not yield obvious improvement to the SKLSH. In Figs.2(c) and 3(c), the performance of the PSH is very close to that of the BSH because a longer hash code is used which reduces the effects of hash bit selection. In all experiments, the BSH yields the best performance in terms of largest area under precision-recall curves. This shows that the code independence and the information capacity in the BSH are significant for achieving higher retrieval performances.

Table 1 Values of parameters selected in our experiments

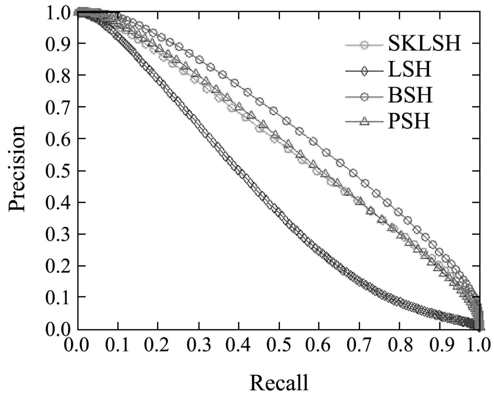
| CIFAR10 | Additional bits | Threshold |
|---|---|---|
| 128 | 128 | 55 |
| 256 | 128 | 58 |
| 512 | 128 | 43 |
| **MNIST** | **Additional bits** | **Threshold** |
| 128 | 64 | 52 |
| 256 | 128 | 65 |
| 512 | 128 | 49 |



(a) CIFAR10_128bits

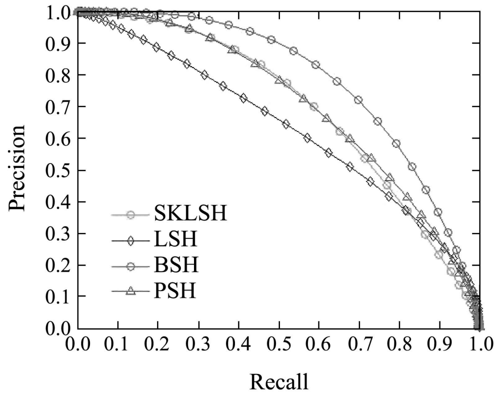


(b) CIFAR10_256bits

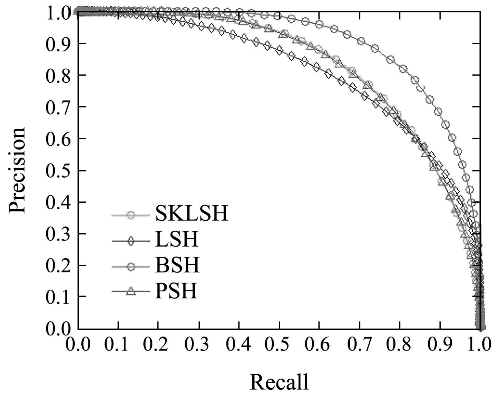


(c) CIFAR10_512bits

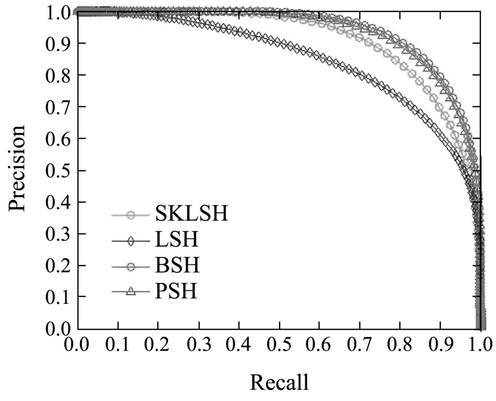
Fig. 2 Recall-precision curve of hash methods on CIFAR10 dataset with (a) 128 bits, (b) 256 bits, (c) 512 bits



(a) MNIST_128bits



(b) MNIST_256bits



(c) MNIST_512bits

Fig. 3 Recall-precision curve of hash methods on MNIST dataset with (a) 128 bits, (b) 256 bits, (c) 512 bits

## 5 Conclusions

In this paper, we propose the bit selection hashing (BSH) to improve the image retrieval performance of the SKLSH which selects hash functions randomly. The BSH evaluates a hash function with three metrics: information capacity, similarity fitness, and code independence. By generating more hash functions randomly at the beginning, a better set of hash function is selected to form efficient hash code for image retrieval. Experimental results show that the BSH yields better retrieval performance than other hashing methods in comparison.

In our future works, we will extend the bit selection hashing to semi-supervised hashing methods for large scale semantic image retrieval problems. In the BSH, a Hamming distance threshold is applied to judge whether two images are similar or not. If a partly labeled database is used for training, the semantic information can be directly used which is more reliable. A new semantic-based fitness will be researched for semi-supervised cases. Furthermore, the combination model of the three metrics should be improved to further enhance the retrieval performance.

## References:

[1] SILPA-ANAN C, HARTLEY R. Optimised kd-trees for fast image descriptor matching [C] //*Conference on Computer Vision and Pattern Recognition*. Anchorage, AK, USA: IEEE, 2008: 1 – 8.

[2] BEYGELZIMER A, KAKADE S, LANGFORD J. Cover trees for nearest neighbor [C] //*Proceedings of the 23rd International Conference on Machine learning*. New York, USA: ACM, 2006: 97 – 104.

[3] WU C, ZHU J, CAI D, et al. Semi-supervised nonlinear hashing using bootstrap sequential projection learning [J]. *IEEE Transactions on Knowledge and Data Engineering*, 2013, 25(6): 1380 – 1393.

[4] DATAR M, IMMORLICA N, INDYK P, et al. Locality-sensitive hashing scheme based on p-stable distributions [C] //*Proceedings of the twentieth annual symposium on Computational geometry*, Brooklyn, New York, USA: ACM, 2004: 253 – 262.

[5] GIONIS A, INDYK P, MOTWANI R. Similarity search in high dimensions via hashing [C] //*Proceedings of the 25th Very Large Data Bases Conference*. Edinburgh, Scotland: Morgan Kaufmann, 1999, 99(6): 518 – 529.

[6] ANDONI A, INDYK P. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions [C] //*47th Annual IEEE Symposium on Foundations of Computer Science*. Berkeley, CA, USA: IEEE, 2006: 459 – 468.

[7] RAGINSKY M, LAZEBNIK S. Locality-sensitive binary codes from shift-invariant kernels [C] //*Advances in Neural Information Processing Systems*. Vancouver, BC, Canada: MIT Press, 2009, 22: 1509 – 1517.

[8] ZENG Z, WU Z, NG W W Y. Projection selection hashing [C] //*International Conference on Wavelet Analysis and Pattern Recognition*. Guangzhou, China: IEEE, 2015: 185 – 191.

[9] KIRA K, RENDELL L A. A practical approach to feature selection [C] //*Proceedings of the Ninth International Workshop on Machine learning*. Aberdeen, Scotland, United Kingdom: Morgan Kaufmann, 1992.

[10] STRECHA C, BRONSTEIN A M, BRONSTEIN M M, et al. LDA-hash: Improved matching with smaller descriptors [J]. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2012, 34(1): 66 – 78.

[11] KULIS B, JAIN P, GRAUMAN K. Fast similarity search for learned metrics [J]. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2009, 31(12): 2143 – 2157.

[12] HUANG L K, YANG Q, ZHENG W S. Online hashing [C] //*Proceedings of International Joint Conference on Artificial Intelligence*. Beijing, China: IJCAI, 2013: 1422 – 1428.

[13] XU H, WANG J, LI Z, et al. Complementary hashing for approximate nearest neighbor search [C] //*IEEE International Conference on Computer Vision*. Barcelona, Spain: IEEE, 2011: 1631 – 1638.

[14] LI P, CHENG J, LU H. Hashing with dual complementary projection learning for fast image retrieval [J]. *Neurocomputing*, 2013, 120(23): 83 – 89.

[15] WANG J, KUMAR S, CHANG S F. Semi-supervised hashing for large-scale search [J]. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2012, 34(12): 2393 – 2406.

[16] NG W W Y, LV Y, ZENG Z, et al. Sequential conditional entropy maximization semi-supervised hashing for semantic image retrieval [J]. *International Journal on Machine Learning and Cybernetics*, online first, 2016.

[17] GONG Y, LAZEBNIK S, GORDO A, et al. Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval [J]. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2013, 35(12): 2916 – 2929.

[18] CHEN L, XU D, TSANG I W H, et al. Spectral embedded hashing for scalable image retrieval [J]. *IEEE Transactions on Cybernetics*, 2014, 44(7): 1180 – 1190.

[19] NG W W Y, LV Y, YEUNG D S, et al. Two-phase mapping hashing [J]. *Neurocomputing*, 2015, 151(3): 1423 – 1429.

[20] LV Y, NG W W Y, ZENG Z, et al. Asymmetric cyclical hashing for large scale image retrieval [J]. *IEEE Transactions on Multimedia*, 2015, 17(8): 1225 – 1235.

[21] WEISS Y, TORRALBA A, FERGUS R. Spectral hashing [C] //*Advances in Neural Information Processing Systems*. Vancouver, BC, Canada: MIT Press, 2009: 1753 – 1760.

[22] LIU X, HE J, LANG B, et al. Hash bit selection: a unified solution for selection problems in hashing [C] //*Conference on Computer Vision and Pattern Recognition*. Portland, OR, USA: IEEE, 2013: 1570 – 1577.

## 作者简介:

**田　星**　(1991–), 男, 博士研究生, 目前的研究方向包括图像检索和机器学习, E-mail: shawntian123@gmail.com;

**陆筱怡**　(1995–), 女, 目前的研究方向包括图像检索和机器学习, E-mail: luxy901@gmail.com;

**吴永贤**　(1978–), 男, 博士, 教授, 博士生导师, IEEE高级会员. 长期从事机器学习领域中的神经网络泛化能力研究, 为大数据中的机器学习与模式识别、大规模信息检索和深度学习等应用提供了一个创新而高效的训练方法. 相关研究已在包括IEEE TNNLS, IEEE T-Cyb和IEEE TMM等国际权威期刊及国际会议上发表了过百篇文章, 主持3项国家自然科学基金及1项教育部新世纪人才计划; 2016年获选为中国计算机学会计算机视觉专委会委员; 2010年起担任Springer出版SCI国际期刊JMLC副主编; 2010年获选担任2011–2013年间IEEE SMC学会董事会成员; 担任ICMLC2015国际会议程序委员会主席; 担任IEEE TNN等10余个知名高水平国际期刊评审员, E-mail: wingng@ieee.org;

**黄家健**　(1995–), 男, 目前的研究方向包括图像检索和机器学习, E-mail: kuchien.wong@gmail.com.