# Minimal Similarity Loss Hashing based on Optimized Self-taught Clustering Method

Zhen Wang
School of Computer Science and Technology
Shandong University of Technology, Zibo, China

Chenyu Wang
School of Computer Science and Technology
Shandong University of Technology, Zibo, China

Anlei Jiao
School of Computer Science and Technology
Shandong University of Technology, Zibo, China

## ABSTRACT
In With the rapid growth of the amount of web images, how to fast respond the approximate nearest neighbors (ANN) search task has been concerned by researchers. As hashing algorithms map the floating point data into binary code and achieve ANN search task according to Hamming distances, it has the advantages of low computational time complexity. To obtain an excellent ANN search performance based on compact binary code, many algorithms adopt machine learning mechanism to train hashing functions. For instance, k-means hashing (KMH) and stacked k-means hashing (SKMH) utilize k-means clustering mechanism to learn encoding centers. Due to KMH and SKMH randomly generate initial centers, many centers would converge to the same solution. To fix the above problem, a novel hashing method termed as minimal similarity loss hashing (MSLH) is proposed to generate the initial centers with maximum average distance by a self-taught mechanism. Furthermore, MSLH defines both the similarity loss and the quantization loss as the objective function. By minimizing the similarity loss, MSLH can approximate the data pairs' Euclidean distance by their Hamming distance. The encoding results with minimal quantization loss map the nearest neighbors into the same binary code, which well adaptive to the data distribution. The ANN search comparative experiments on three public available datasets including NUS-WIDE and 22K LabelME show that MSLH can achieve a superior performance.

## General Terms
Image retrieval, Hashing algorithm.

## Keywords
Hashing algorithm, self-taught clustering, Iterative optimization mechanism, Similarity preserving.

## 1. INTRODUCTION
With the rapid development of Internet technology, the amount of data such as images and videos is increasing rapidly. More and more researchers have focused on how to quickly search similar images from a large database. Recently, hashing algorithms [1, 2, 3] have been widely utilized in image search task. Hashing algorithms can map high dimensional floating point data into compact binary codes, and return ANN search results according to their Hamming distances. It has the advantage of low storage occupancy and fast respond.

The classical method, local sensitive hashing (LSH) [4], randomly generates linear hashing functions, and maps the floating point data into binary codes according to their projection signs. As the learning process of LSH is independent from training data, it needs longer binary codes to obtain a satisfy ANN search performance. To avoid the above problem, machine learning mechanisms are introduced to learn compact binary codes while satisfying similarity preserving restriction. Spectral hashing [5] learns binary codes by graph partition, and it requires the data distribution should be uniform. Moreover, the time complexity of constructing similar graph is relative higher. Principal component hashing [6] project the data by the principal component functions, and generate the binary code according to the projected signs. Random rotation hashing (RR hashing) [7] rotates the PCA-projected data with a random matrix, and it encodes the data according to the vertices of a hyper cubic. In contrast, iterative quantization hashing (ITQ) [8] finds the optimal rotation matrix by machine learning mechanism. In RR hashing [7] and ITQ [8] method, the fixed encoding centers make the encoding results not well adaptive to the data distribution. In contrast, k-means hashing (KMH) [9] and stacked k-means hashing (SKMH) [10] learns encoding centers by minimizing the quantization error, and adopts the k-means clustering method to learn the encoding centers. However, the initial centers in KMH are randomly picked which would lead inferior clustering results.

In this paper, a novel hashing method termed as minimal similarity loss hashing (MSLH) is proposed. MSLH learns initial centers by a self-taught mechanism. The flowchart of the proposed method is shown in Figure 1. Firstly, the dimensions of training data are uniformly divided into different sub-spaces, and MSLH parallel computes all sub-codes. In each sub-space, MSLH initializes the encoding centers with maximum average distance by a self-taught mechanism. During the training procedure, an iterative optimization mechanism is adopted to learn the encoding centers by simultaneously minimizing the similarity loss and the quantization loss. Finally, MSLH encodes the data as the same binary code as their nearest encoding center, and return their nearest neighbors according to the Hamming distances.

The main contributions of this paper can be concluded as following:

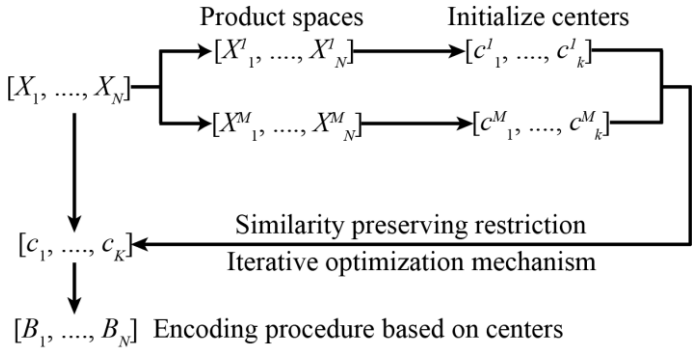


**Fig. 1. The framework of minimal similarity loss hashing (MSLH)**

1. The self-taught mechanism is proposed to learn the initial clustering centers with maximum average distance, which can guarantee the clustering results well adaptive to data distribution.
2. MSLH simultaneously minimizes the quantization loss and similarity loss by an iterative mechanism. When the algorithm converges, the learnt encoding centers are consistent with the clustering results, and their binary codes can well preserve the original similarity relationship.
3. The product quantization method is employed to divide the dimensions of data into different sub-spaces, and MSLH parallel computes all sub-codes to reduce the training time complexity.

## 2. ALGORITHM

To guarantee that the approximate nearest neighbors (ANN) search results obtained in the Hamming space are consistent with those in the Euclidean space, MSLH requires the nearest neighbors in the Euclidean space have the same binary code and their Hamming distances can approximate their Euclidean distance. In this paper, an optimized self-taught clustering algorithm is proposed to cluster the nearest neighbors to the same group, and encode the samples belongs to the identical clustering group as the same binary code. For the samples in different clustering group, MSLH demands their binary codes should minimize the value of similarity loss function.

### 2.1 The Optimized Self-taught Clustering Algorithm

To cluster the nearest neighbors into the same group, the classical method, k-means clustering algorithm, can be adopted to learn the clustering groups. However, k-means clustering algorithm randomly pick $K$ ($K$ clustering groups are learnt) samples as initialize centers, and the gradient descent algorithm is adopted to learn the clustering groups which would lead local optimal solutions. As a result, different initial centers may lead diverse convergence results, and the clustering groups do not well match the samples' distribution in the Euclidean space.

To fix the above problem, MSLH proposes an optimized self-taught clustering algorithm which generates the initial centers by maximizing the average distance values among all centers.

An excellent clustering method should enlarge the distances among different clusters and minimize the distances among the samples in the same group. Therefore, MSLH proposes to learn the initial cluster centers by maximum the average distance values as defined in Eq. (1), which can avoid the initial centers converging to the same local optimal solution. Thanks to this measure, the clustering results are consistent with the samples' distribution in the Euclidean space.

$$\sum_{i=1}^{K}\sum_{j=i}^{K} d(c_i, c_j) \tag{1}$$

$K$ represents the number of clustering groups. $c_i$ is the $i$-th clustering group, and $d(c_i, c_j)$ returns the Euclidean distance.

The process of generating initial cluster centers which satisfy the restriction defined in Eq. (1) is shown in Fig. 2. Firstly, the mean value of all samples is considered as the first cluster center $c_1$ as in Fig. 2(a). Then, the sample which has the largest distance to the first center is chosen as the second cluster center $c_2$, and the samples are divided into two groups based on the obtained cluster centers. For the obtained sub-group in which the samples' distribution are scatter, MSLH will iteratively execute the above two steps until enough cluster centers are obtained. In Fig. 2(c), the center $c_1$ is updated based on the mean value of the red samples in the same group. Similarly, the center $c_3$ with maximum average distance value to $c_1$ and $c_2$ is chosen as in Fig. 2(d).

The process of generating the initial cluster centers with maximum average distance is shown in algorithm 1.

**Algorithm 1** Generating initial centers by optimized self-taught method

**Input:** Training data $X=x_1,\cdots, x_n$.

**Output:** The initial centers $C=c_1,\cdots, c_n$.

1: Put X into the null dataset G.

2: **repeat**

3: Select the group in which the samples' distribution are scatter from G.

4: Consider the mean value of the samples as the first center

5: Choose the sample with the largest distance to the first center as the second center

6: Divide the samples into two groups

7: Store the group which has a scatter distribution into G

8: Store the center of the group in which the samples are distributed centralized into C

9: **until K** cluster centers are obtained

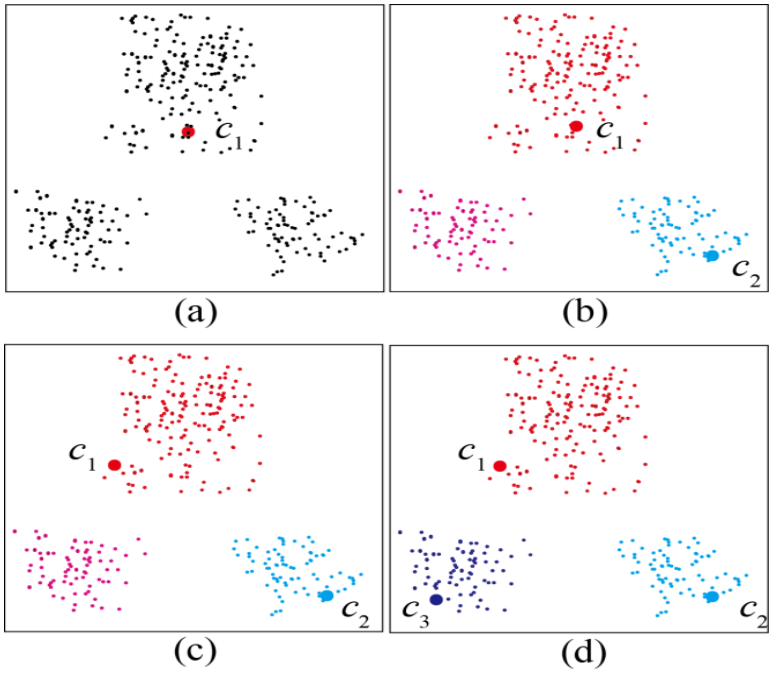


**Fig. 2. The process of generating initial centers by self-taught mechanism**

## 2.2 The Similarity Preserving Objective Function

Hashing algorithms map the floating point data into binary codes, and achieve the approximate nearest neighbors (ANN) task in the Hamming space. During the training process, MSLH demands the similarity relationship among data points in the Hamming space should be consistent with that in the original Euclidean space. To achieve the above goal, MSLH requires the samples' binary codes should minimize the value of the similarity loss function $L_s$ as defined in Eq. (2).

$$L_s = \sum_{i=0}^{K-1}\sum_{j=1}^{K-1}\frac{n_i n_j}{n^2}(d(c_i,c_j) - d_h(b_i,b_j))^2 \quad (2)$$

$c_i$ and $c_j$ separately represent the $i$-th and $j$-th clustering center, and $b_i$ and $b_j$ are their binary codes. $K$ is the number of clustering centers. $n_i$ and $n_j$ separately represent the number of samples in the $i$-th and $j$-th group. $d(c_i, c_j)$ represents the Euclidean distance between the $i$-th and $j$-th clustering center, and $d_h(b_i, b_j)$ is the Hamming distance.

By minimizing the value of the objective function $L_s$, MSLH can achieve the goal of preserving the Euclidean similarity relationship among different clusters in the Hamming space. So, the next task is how to map the similar data points into the same binary code.

In this paper, MSLH maps the data into the same binary code as its cluster center. Thus, the hashing algorithms should cluster the nearest neighbors into the same group. To fulfill the above goal, the quantization error $L_q$ defined in Eq. (3) should be minimized.

$$L_q = \frac{1}{n}\sum \| x - c(x) \|^2 \quad (3)$$

$n$ is the number of samples and $c(x)$ returns the clustering center of $x$. $L_q$ can guarantee the samples with the same binary code are nearest neighbors.

As described above, the objective function for learning clustering centers and its corresponding binary codes can be defined as in Eq. (4)

$$L = L_q + L_s \quad (4)$$

In this paper, MSLH adopts an iterative optimization mechanism to learn the clustering centers and their binary codes by simultaneously minimizing the similarity loss and the quantization loss. The optimization procedure mainly includes two steps as shown below.

(1) Fix the binary codes and update the clustering centers.

Assign each sample to its nearest center, and update the clustering center by Eq. (5).

$$c_i = \frac{1}{n_i}\sum_{j=1}^{n_i} x_j \quad (5)$$

$x_j$ is the sample which belongs to the $i$-th clustering group, and $n_i$ is the number of the samples in the $i$-th clustering group.

(2) Fix the clustering centers, and update their binary code in sequence.

When updating the $i$-th clustering center's binary code $b_i$, the other clustering centers' binary codes are considered as constant values. Then $b_i$ can be computed by Eq. (6).

$$b_i = \arg\min \sum_{j=0}^{K} \| d(c_i,c_j) - d_h(b_i,b_j) \|^2 \quad (6)$$

MSLH iteratively updates the clustering centers and binary codes by repeat executing the steps (1) and (2), until the algorithm converges. Then, the obtained clustering centers can well adaptive to the data distribution, and the Hamming distances computed based on their binary codes can approximate their Euclidean distance.

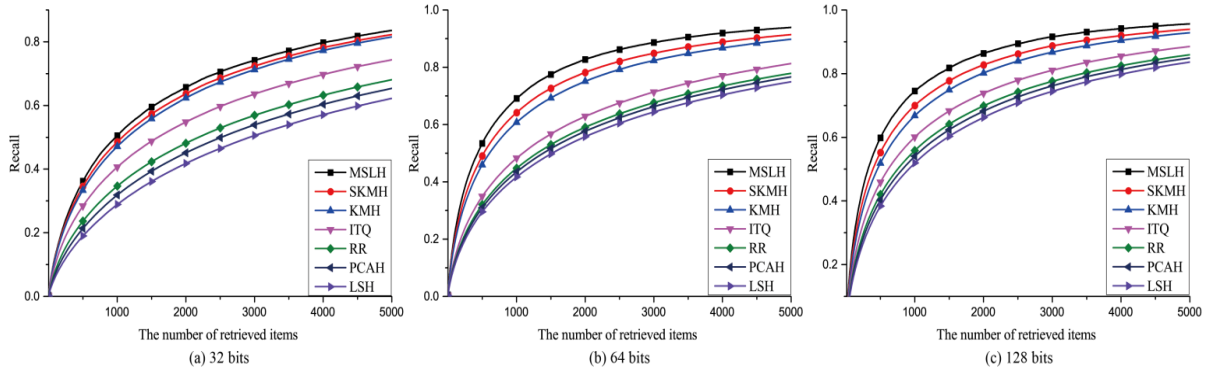


**Fig. 3. The recall curves of ANN search results in 22K LabelMe dataset**.

## 2.3 Parallel Computing

If directly computing the integrated binary codes, the training time complexity would be unacceptable. To decrease the training time complexity, the product quantization[11, 12] method is adopted to divide the dimensions of training data into different sub-spaces, and parallel learn the sub-centers and sub-binary codes in each sub-space.

Given the length of binary code is $B$, $2^B$ centers and corresponding binary codes need to be computed and store. After dividing the dimensions of original data into $M$ spaces, $b=B/M$ bit code need to be learn in each sub-space. Then, only $M \cdot 2^b$ centers and binary codes need to be computed and stored.

To guarantee the dimensions are independent with each other and the variances in each sub-space are balance. The process of dividing the dimensions of data into sub-spaces is described as below:

(1) Compute the dimensions' eigenvalues by principal component analysis algorithm, and sort the dimensions according to their eigenvalues.

(2) Iteratively put the dimension with maximum eigenvalue into the sub-space which has minimal value of the sum of dimensions' eigenvalues, until no dimension is left.

Thanks to the above division procedure, the dimensions of the data are evenly distributed in each sub-space, and the neighbor information contained in each sub-space is equal. As a result, the number of binary bits need to be learnt in each sub-space is identical during the parallel computing procedure.

# 3. EXPERIMENTS AND EXPERIMENTAL RESULTS

## 3.1 Datasets and Experimental Setting

In this paper, the comparative experiments are conducted on three widely used data sets including NUS-WIDE [13], 22K LabelME [14] and ImageNet 100. For the ANN search comparative experiments, each dataset is divided into three parts to train the hashing functions and test its ANN search performance.

NUS-WIDE dataset includes 270 thousand images which are selected from the Flickr dataset, and 190 thousand and 50 thousand images are separately chosen as test dataset and query samples. For NUS-WIDE dataset, the number of query samples is 50 thousand. In 22K LabelME dataset, 20 thousand images are considered as the test dataset, and 2 thousand images are utilized as the query samples. 5 thousand images are randomly selected from 22K LabelME dataset to learn hashing functions. ImageNet 100 including 100 kinds of images is the sub-set of ImageNet database. In ImageNet 100 dataset, the testing database includes 130 thousand images, and 30 thousand images are stored in the training dataset. To compare the ANN search performance, 10 thousand images are randomly selected from ImageNet 100 as the test dataset.

In this paper, the recall curves are adopted to show the experimental results. The recall curves represent the fraction of the true nearest neighbors returned in the Hamming space as defined in Eq. (7).

$$recall = \frac{\#(returned)}{\#(all)}$$

*#*(*returned*) is the number of the true nearest neighbors returned in the Hamming space. *#*(*all*) means the total amount of the true nearest neighbors in the original Euclidean space.

## 3.2 Experimental Results

For the ANN search experiments, the feature descriptors GIST [15] are computed according to the image content in NUS-WIDE, 22K LabelME and ImageNet 100 datasets, and the true nearest neighbors of query samples are defined according to their Euclidean distances. During ANN search procedure, the GIST descriptors are mapped into binary codes, and their nearest neighbors are returned according to Hamming distances. the length of binary code are separately set as 32, 64, and 128. The experimental results are shown in Figs. 3, 4, 5.

The experimental results show that the proposed method has a superior performance. The classical method, local sensitivity hashing (LSH) [4], randomly generates hashing functions which makes the training procedure independent from the training samples. As a result, the ANN search performance of LSH cannot improve obviously as the binary bits increasing. MSLH, SKMH [10], KMH [9], ITQ [8], RR [7] and PCAH [6] adopt machine learning mechanisms to learn the compact binary codes which can preserve the Euclidean similarity relationship among data points. So, the above machine learning based hashing methods can achieve a better performance than LSH. In PCAH [6] method, the data are projected by the principal component functions, and they are encoded as binary codes according to the projected results. Unfortunately, PCAH [6] would separate many nearest neighbors into axis' different sides, and the data are assigned different binary codes. To fix this problem, RR [7] method randomly rotates the PCA-projected data, and its performance is better than PCAH. In contrast, ITQ [8] method iteratively learns the rotation matrix to avoid separating the nearest neighbors into the axis' different sides. However, RR [7] and ITQ [8] do not take the data distribution into consideration,

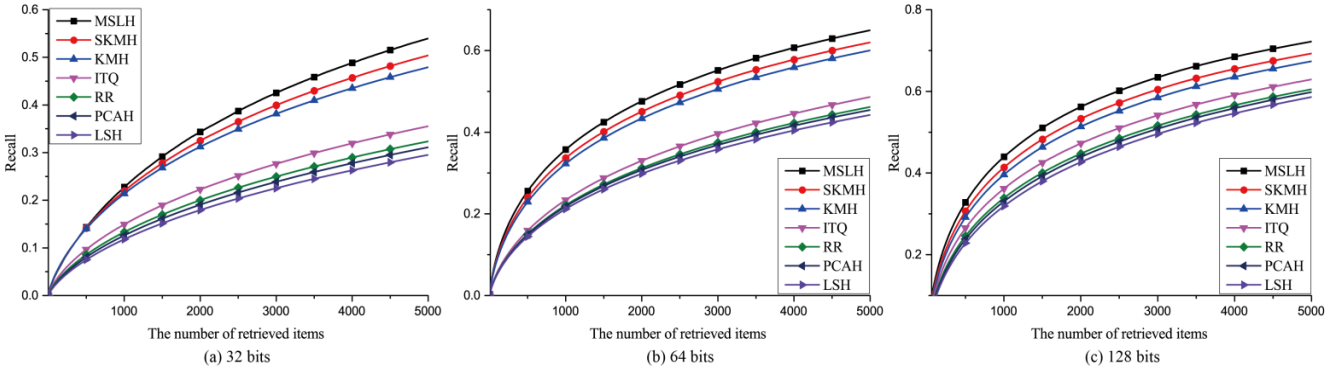


**Fig. 4. The recall curves of ANN search results in NUS-WIDE dataset.**

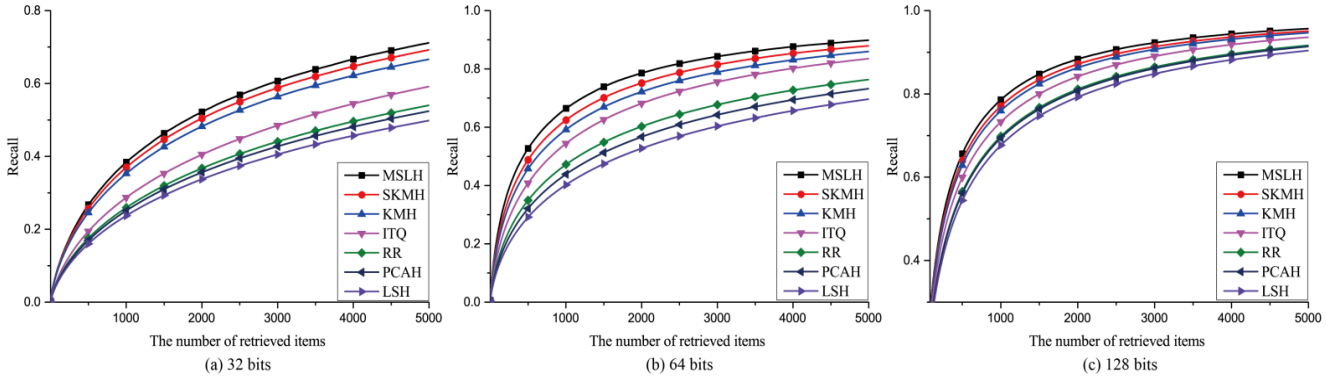


**Fig. 5. The recall curves of ANN search results in ImageNet 100 dataset**.

and the encoding results may conflict with the original similarity relationship among data points. To fix the above problem, KMH [9] and SKMH [10] adopt the k-means clustering method to learn the binary code. So, KMH and SKMH can achieve a better performance than ITQ and RR. SKMH [10] proposes a coarse-to-fine multi-layer lower-dimensional cube to instead the high dimensional cube in KMH, which can effectively reduce the training time complexity. During the training process, both KMH [9] and SKMH [10] randomly generate the initialize centers, which would lead an inferior clustering result. In this paper, MSLH learns the initial centers with maximum average distance by a self-taught mechanism. When the algorithm converges, MSLH can find the extreme points as much as possible. Furthermore, MSLH demands the encoding results minimize both the similarity loss and the quantization loss. As a result, the binary codes can preserve the data points' original Euclidean similarity relationship and the encoding results well adaptive to data distribution. The experimental results have also shown that MSLH achieves the best ANN search performance.

## 4. CONCLUSION

In this paper, a novel hashing method termed as minimal similarity loss hashing (MSLH) is proposed. To guarantee the obtained compact binary code can achieve an excellent ANN search performance, MSLH adopts an iterative optimization mechanism to learn the hashing functions with minimal similarity loss and quantization loss. During the training process, the k-means like mechanism is utilized to learn the encoding centers. But, the random initialize centers would lead an inferior solution. To fix this problem, MSLH devise a self-taught learning procedure to generate the initialize centers which have maximum average distances among themselves. With the assistance of the learned initialize centers, the algorithm can converge to an optimal solution. The experimental results on two large-scale datasets have shown that MSLH can obtain a superior ANN search performance.

## 5. REFERENCES

[1] Wang Z., Sun F., Zhang L., Liu P.. 2020 Minimal residual ordinal loss hashing with an adaptive optimization mechanism. EuRASIP Journal on Image and Video Processing, 10.

[2] Wang Z., Sun F., Zhang L., Wang L.. 2020 Top position sensitive ordinal relation preserving bitwise weight for image retrieval [J]. Algorithms, 13(1), 18.

[3] Wang Z., Zhang L., Sun F., Wang L., Liu S. Relative similarity preserving bitwise weights generated by an adaptive mechanism [J]. 2019 Multimedia Tools and Applications, 78 (17), pp. 24453-24472.

[4] Datar M., Immorlica N., Indyk P., Mirrokni V. S. 2004 Locality-sensitive hashing scheme based on p-stable distributions [C]. In Proceedings of twentieth Annual Symposium on Computational Geometry. Brooklyn, New York, USA, 253-262.

[5] Weiss Y., Torralba A., Fergus R.. 2008 Spectral hashing [C]. In Proceedings of the Advances in Neural Information Processing Systems. British Columbia, Canada, 1753-1760.

[6] Yusuke M., Toshikazu W.. 2009 Principal component hashing: An accelerated approximate nearest neighbor search [C]. In proceedings of Pacific Rim Symposium on Advances in Image and Video Technology. Springer-

Verlag, 374-385.

[7] Jegou H., Douze M., Schmid C., Perez P. 2010 Aggregating local descriptors into a compact image representation [C]. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. San Francisco, California, USA, 3304-3311.

[8] Gong Y., Lazebnik S.. 2011 Iterative Quantization: A procrustean approach to learning binary codes [C]. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Colorado Springs, CO, USA, 817-824.

[9] He K., Wen F., Sun J.. 2013 K-means hashing: an affinity-preserving quantization method for learning binary compact codes [C]. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Portland, Oregon, 2938-2945.

[10] Chen Y., Li Z., Shi J., Liu Z. and Qu W.. 2018 Stacked K-Means Hashing Quantization for Nearest Neighbor Search [C]. In Proceedings of the IEEE Fourth International Conference on Multimedia Big Data. 1-4.

[11] Ge T., He K., Ke Q., Sun J.. 2013 Optimized Product Quantization for Approximate Nearest Neighbor Search [C]. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Portland, Oregon, 2946-2953.

[12] Ge T., He K., Ke Q., Sun J.. 2014 Optimized Product Quantization [J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 36(4): 744-755.

[13] Wang X., Shi Y., Kitani K.. 2016 Deep Supervised Hashing with Triplet Labels [C]. In Proceedings of Asian Conference on Computer Vision. Taipei, Taiwan, 70-84.

[14] Cakir F., Sclaroff S.. 2015 Adaptive Hashing for Fast Similarity Search [C]. In Proceedings of the 2015 IEEE International Conference on Computer Vision (CVPR). Santiago, Chile, 1044-1052.

[15] Oliva A., Torralba A.. 2001 Modeling the shape of the scene: A holistic representation of the spatial envelope [J]. International journal of computer vision, 42(3): 145-175.