DISTANCE PRESERVING MARGINAL HASHING FOR IMAGE RETRIEVAL

Li Wu, Kang Zhao, Hongtao Lu, Zhen Wei, Baoliang Lu*

Key Laboratory of Shanghai Education Commission for Intelligent Interaction and Cognitive Engineering
Department of Computer Science and Engineering, Shanghai Jiao Tong University, P.R.China
{beautifulgirl, sjtuzk, htlu, zhenni94, bllu}@sjtu.edu.cn

ABSTRACT

Hashing for image retrieval has attracted lots of attentions in recent years due to its fast computational speed and storage efficiency. Many existing hashing methods obtain the hashing functions through mapping neighbor items to similar codes, while ignoring the non-neighbor items. One exception is the Local Linear Spectral Hashing (LLSH), which introduces negative values into the local affinity matrix to map non-neighbor images to non-similar codes. However, setting $10th$ percentile distance in affinity matrix as a threshold, which is used to judge neighbors and non-neighbors, is not reasonable. In this paper, we propose a novel unsupervised hashing method called $Distance\ Preserving\ Marginal\ Hashing$ (DPMH) which not only makes the average Hamming distance minimized for the intra-cluster pairs and maximized for the inter-cluster pairs, but also preserves the distance of non-neighbor points. Furthermore, we adopt an efficient sequential procedure to learn the hashing functions. The experimental results on two large-scale benchmark datasets demonstrate the effectiveness and efficiency of our method over other state-of-the-art unsupervised methods.

***Index Terms*—** hashing, image retrieval, margin, non-neighbor images, sequential procedure

1. INTRODUCTION

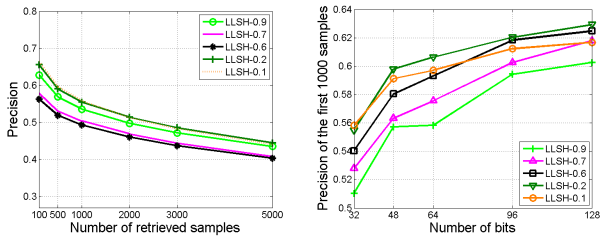
Thanks to the rapid advances of the Internet, we can easily share our pictures and images on the website for various purposes, such as **Flickr** and **YouTube**. This makes hundreds of millions of images available online. Therefore there is an emerging need of searching visually relevant images from very huge databases. Content-Based Image Retrieval (CBIR) methods have attracted substantial attentions over the past decades. In a traditional CBIR system, given one query image, it exhaustively searches the nearest neighbors over all images in the database, which has a linear time complexity and is thus not scalable when the number of images is very large.

* Corresponding author

Over the past decades, numerous techniques have been proposed to accelerate the nearest neighbors search. For example, many tree-based approximate nearest neighbor (ANN) search techniques, such as kd-tree [1], metric tree [2], have been developed. These techniques aim to speed up nearest neighbors search and achieve promising performance in real applications. Unfortunately, these methods don't perform very well in the condition of high-dimensional data. Recently, hashing methods have attracted considerable attentions in computer vision community and has been successfully applied to computer vision, information retrieval, and data mining. In these problems, hashing methods aim to encode high-dimensional images into compact binary codes, while maintaining aspects of the structure of the original data as much as possible, so that the search in the Hamming space becomes efficient and effective. The early exploration of hashing focuses on using random projections to construct randomized hash functions. One of the most popular method is Locality Sensitive Hashing (LSH) [3].

Following LSH, its several variations, including kernelized LSH [4], mahalanobis distance [5], p-stable distributions [6], which accommodate more distance metrics, have been proposed. Another effective method called Spectral Hashing (SH) [7] formulates the hashing problem as a particular form of graph partition to seek a code with balanced partitioned and uncorrelated bits. However, its assumption of uniform data distribution is usually not true and it doesn't take the non-neighbor points into consideration. Due to these drawbacks of SH, LLSH [8] has been put forward which minimizes the average Hamming distances and introduces negative values into new local neighbor matrix to map neighbor items into similar codes, and non-neighbor items to non-similar codes. LLSH judges the neighbor items and non-neighbor items by comparing their distances with a predefined threshold. But this threshold is not reasonable and has no theoretical basis.

In this paper, we propose a $Distance\ Preserving\ Marginal\ Hashing$ for image retrieval to solve the problems of SH and LLSH mentioned above. Similar to LLSH, we encode neighbor points into similar binary codes and map non-neighbor points to non-similar codes. More specifical-



(a) (b)

Fig. 1. The experimental results on CIFAR10 dataset: (a) Precision curves at different thresholds with the $10th$, $20th$, $60th$, $70th$, $90th$ percentile distance in $\mathbf{A}$. (b) Hamming ranking precision of the first 1000 returned samples with different thresholds.

ly, we not only make the average Hamming distance minimized for the intra-cluster pairs, and maximized for the inter-cluster pairs, but also preserve the distance of non-neighbor points. With the spectral relaxation, we apply a sequential procedure to relax the orthogonality constraints and solve the proposed model efficiently. The experimental results on two large datasets demonstrate that our method achieves a promising performance.

The remainder of this paper is organized as follows. We briefly introduce the related work in Section 2. Section 3 presents the formulation as well as our solving algorithm for DPMH. Section 4 provides the experimental evaluation on real image datasets. The conclusions are given in Section 5.

2. RELATED WORK

2.1. Spectral Hashing (SH)

Spectral Hashing (SH) is proposed by Weiss $et\ al.$. The goal of this method is to learn binary codes by mapping neighbors in the input space to similar codes in the Hamming space, as well as enforcing the codes to be balanced and uncorrelated to each other. The objective function that SH codes $H(x) = \{h_k(x)\}$, k = 1,...,K can be written as:

$$\begin{aligned} \min \quad & \sum_{i,j} w(x_i, x_j)||H(x_i) - H(x_j)||^2 \qquad (1) \\ s.t. \quad & h_k(x_i) \in \{-1, 1\} \\ & \sum_i h_k(x_i) = 0 \\ & \sum_i h_k(x_i)h_l(x_i) = 0, k \neq l \end{aligned}$$

Here $\mathbf{W} = (w_{ij})_{n\times n}$ with $w_{ij} = \exp(-||x_i - x_j||^2/\sigma^2)$ is the graph affinity matrix, where σ is the bandwidth parameter. For a single bit (K = 1), the solution for above optimization (1) is equivalent to balanced graph partitioning and is NP hard. The combination of K-bit balanced partition makes it more difficult. The above optimization can be solved by spectral graph analysis [9] with relaxation of the constraints. Furthermore, after assuming a uniform data distribution, the out-of-sample extension problem can be efficiently overcome by a closed-form solution [7]. Along the principal directions of the data, the final SH algorithm can produce very compact hash codes by thresholding with a nonlinear function.

2.2. Local Linear Spectral Hashing (LLSH)

In order to overcome the limitation of SH, LLSH has recently been proposed to solve this problem. The goal of LLSH is to minimize the following objective function:

$$\begin{aligned} \min \quad & \sum_{i,j} A_{ij}||\mathbf{y}_i - \mathbf{y}_j||^2 \qquad (2) \\ s.t. \quad & \mathbf{Y} \in \{-1, 1\}^{n\times r} \\ & \mathbf{1}^T\mathbf{Y} = 0 \\ & \mathbf{Y}^T\mathbf{Y} = n\mathbf{I} \end{aligned}$$

Here, $\mathbf{Y}$ is the r-bit codes of n points, and $\mathbf{y}_i \in \{-1, 1\}^r$ is the ith row in $\mathbf{Y}$. The difference between LLSH and SH is the affinity matrix. The size of the affinity matrix in SH is $n\times n$. It is intractable when n is very large. However, LLSH uses a local affinity matrix $\mathbf{A}$ constructed from m samples, whose size is $m \times m (m \ll n)$. LLSH introduces negative values into the local affinity matrix and judges neighbors and non-neighbors based on a threshold which is the $10th$ percentile distance in $\mathbf{A}$ calculated with L_2 norm. Thus, the point pair whose distance is close to the threshold will be inevitably classified to either the neighbor points or non-neighbors points. Fig.1 illustrates that the experimental results significantly fluctuate with small change of thresholds. Note that when the number of retrieved samples is 100, the range of precision varied at different thresholds is more than 0.1. Hence, it is not reasonable to set the $10th$ percentile distance in affinity matrix as a threshold.

3. DISTANCE PRESERVING MARGINAL HASHING

This section presents the proposed unsupervised method Distance Preserving Marginal Hashing. For the first step, we apply the Principal Component Analysis (PCA) [10] to reduce the original data to appropriate dimensions. The core part of our method is in the second step detailed in Section 3.2, where we try to preserve the neighbor and non-neighbor relationships in the projection data. In Section 3.3, a sequential procedure is applied to solve the proposed model and learn the projection directions.

Let us first introduce some notations. Given a data set $\mathbf{X}$ = $[x_1, x_2, ..., x_n]^T \in R^{n\times d}$ and assuming $\mathbf{X}$ is zero-centered, i.e.,$\sum_{i=1}^n x_i = 0$, our goal is to map each point x_i to a K-dimensional binary code $y_i \in \{-1, +1\}^K$. In this paper,

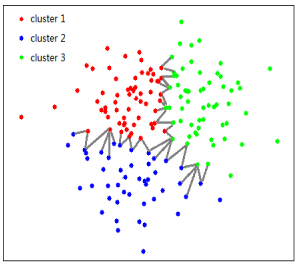


Fig. 2. The adjacency relationships between the marginal data pairs from different clusters.

we restrict the transformation to a linear one. Thus, we can obtain the hash code y_i of the sample x_i by a sign function as follows:

$$y_i = sgn(x_i^T \mathbf{W}) \tag{3}$$

where $\mathbf{W} = [w_1, ..., w_K] \in R^{d \times K}$ is the projection matrix that we need to learn.

3.1. Dimensionality Reduction by PCA

In our method, we first project the data set into m-dimensional subspace by PCA:

$$\begin{aligned} \max \quad & tr(W_{pca}^T \mathbf{X}^T \mathbf{X} W_{pca}) \\ s.t. \quad & W_{pca}^T W_{pca} = \mathbf{I}_{m \times m} \end{aligned} \tag{4}$$

where $W_{pca} = [w_1, ..., w_m] \in R^{d \times m}$ denotes the transformation matrix of PCA, $tr(\cdot)$ denotes the matrix trace. Here m is larger than K so that we can remove some redundant information while without losing too much information. This step can increase the robustness and accelerate the method. We denote the data matrix after PCA projection as $X = \mathbf{X} W_{pca}$. The subsequent processing will be made on the projected data X.

3.2. DPMH Formulation

As mentioned above, we intend to map the points from the same cluster to binary codes that are as close as possible, while those from different clusters to binary codes that are far away from each other, and at the same time we preserve the Hamming distance of non-neighbor points. Since we have no label information, we first obtain u clusters via clustering algorithm such as K-means. Let π_c and n_c denote the index set and number of the samples belonging to the c^{th} cluster respectively.

Inspired by Marginal Fisher Analysis [11], we make use of neighbor relationships of the points in the same cluster and marginal point pairs from different clusters to achieve intra-cluster aggregation and inter-cluster separability.

Apparently, the data pairs in the same cluster have the neighbor relationships, we map these similar points to similar binary codes within a small Hamming distance. We construct an adjacency matrix $\mathbf{B}$, whose entry is defined as:

$$b_{ij} = \begin{cases} 1, & \text{if } i \in N_{k_1}^+(j) \text{ or } j \in N_{k_1}^+(i) \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

here, $N_{k_1}^+(i)$ indicates the index set of the k_1 nearest neighbors of the sample x_i in the same cluster. To realize intra-cluster aggregation, we minimize the following objection function [12]:

$$\min \sum_{i,j} b_{ij} ||y_i - y_j||^2 \tag{6}$$

Empirically, if x_i and x_j are similar, then y_i and y_j will be close as well. By substituting Y with $sgn(XW)$, we apply the spectral relaxation trick to drop the sign functions, then Eq.(6) becomes an optimization problem for solving W:

$$\min \ tr(W^T X^T L_b X W) \tag{7}$$

where $L_b = D_b - \mathbf{B}$ is the graph Laplacian [13], $D_b = diag(\mathbf{B}\mathbf{1}_{n \times 1})$.

Furthermore, considering the neighbor relationships between marginal data pairs from different clusters, we define a similarity matrix $\mathbf{S}$ as below:

$$s_{ij} = \begin{cases} 1, & \text{if } (i,j) \in E_{k_2}(c_i) \text{ or } (i,j) \in E_{k_2}(c_j) \\ 0, & \text{otherwise} \end{cases} \tag{8}$$

Here, $E_{k_2}(c)$ is a set of data pairs that are the k_2 nearest pairs among the set $\{(i,j), i \in \pi_c, j \notin \pi_c\}$. For each cluster c, if the pair (i,j) is one of the k_2 shortest pairs among the set $\{(i,j), i \in \pi_c, j \notin \pi_c\}$, we set s_{ij} = 1, otherwise, s_{ij} = 0. As Fig.2 shows, the adjacency relationships between the marginal data pairs are taken into account for separating the clusters. The problem of solving W for inter-cluster separability is formulated as:

$$\max \sum_{i,j} s_{ij} ||y_i - y_j||^2 \tag{9}$$

Applying similar spectral relaxation to problem (9), we have another optimization problem :

$$\max \ tr(W^T X^T L_s X W) \tag{10}$$

where $L_s = D_s - \mathbf{S}$, $D_s = diag(\mathbf{S}\mathbf{1}_{n \times 1})$.

Having considered the neighbor relationships of the data pairs in the same cluster and marginal pair points from different clusters, we also need to pay close attention to the non-neighbor relationships to preserve the Hamming distance of non-neighbor points. Suppose p_{ij} is the Euclidean distance

Algorithm 1 Distance Preserved Marginal Hashing (DPMH)

Input: data $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n]^\top$; parameters , η and γ; number of hash bits K.

Output: projection matrix $\mathbf{W}$.

1: Use PCA to project the data set into m dimensions, the project matrix $W_{pca} \in R^{d\times m}$ is generated by (4), then $X = \mathbf{X}W_{pca}$
2: Constructing the matrix $\mathbf{S}_w$ by $X^T L_b X$, and $\mathbf{S}_t$ by $X^T(L_s + L_p)X$, where $L_b,\ L_s,\ L_p$ are the graph Laplacian.
3: $w_1 \leftarrow$ the largest eigenvector of $(\mathbf{S}_w)^{-1}\mathbf{S}_t$.
4: **for** $i = 2, ..., K$ **do**
5: $w_i \leftarrow$ the largest eigenvector of matrix $(\mathbf{S}_w + \eta\sum_{j=1}^{i-1}\gamma^{i-j}w_j w_j^\top)^{-1}\mathbf{S}_t$.
6: **end for**
7: Get $W^* = [w_1, ..., w_K] \in R^{m\times K}$.
8: The final linear projection direction is $\mathbf{W} = W_{pca}W^*$

between x_i, x_j in the original data space, we define a distance matrix $\mathbf{P} = (p_{ij})_{n\times n}$. Then, we maximize the following objective function:

$$\max \sum_{i,j} p_{ij}||y_i - y_j||^2 \tag{11}$$

Intuitively, Eq.(11) implies that the farther data pair is, the larger the hamming distance $||y_i - y_j||$ is. Hence, we can preserve the Hamming distance of non-neighbor points. With the relaxation, problem (11) becomes:

$$\max\ tr(W^T X^T L_p X W) \tag{12}$$

where $L_p = D_p - \mathbf{P}$, $D_p = diag(\mathbf{P}\mathbf{1}_{n\times 1})$.

In summary, getting hashing project W, we should not only minimize (7), but simultaneously maximize (10), (12). The formulation of our hashing is:

$$\max_W \quad \frac{tr(W^T\mathbf{S}_t W)}{tr(W^T\mathbf{S}_w W)} \tag{13}$$

where $\mathbf{S}_t = X^T(L_s + L_p)X,\ \mathbf{S}_w = X^T L_b X$. Such formulation forces the learned hash functions to preserve the Hamming distance of point pairs and separate the points in different clusters, while mapping similar points into close binary codes according to the adjacency relationship.

3.3. An Efficient Algorithm for Solving DPMH

Problem (13) can be easily solved by solving the following generalized eigendecomposition problem:

$$\mathbf{S}_t W = \mathbf{S}_w W \Lambda \tag{14}$$

However, such a method doesn't take the independence of the hash bits into consideration. Motivated by SH, we would like the hash bits to be independent and then relax the pairwise decorrelation of bits $Y^T Y = n\mathbf{I}$ by imposing the constraints $W^T W = \mathbf{I}$ as in [14], which requires all the projection directions to be unite vectors and mutually orthogonal. Thus, the relaxed problem can be expressed as :

$$\begin{aligned} \max_W \quad & \frac{tr(W^T\mathbf{S}_t W)}{tr(W^T\mathbf{S}_w W)} \\ s.t. \quad & W^T W = \mathbf{I} \end{aligned} \tag{15}$$

Actually the strict orthogonality constrains don't perform well, so we learn the hash functions by adopting a sequential procedure inspired by [15] to relax the strict orthogonality constrains. We learn one projection w_i by gradually reducing the correlation with previously learned projections $w_{i-1}, ..., w_1$. Thus, we add it as a penalty term to the objective function and determine w_i by solving:

$$\begin{aligned} \max_{w_i} \quad & \frac{w_i^T\mathbf{S}_t w_i}{w_i^T\mathbf{S}_w w_i + \eta\sum_{j=1}^{i-1}\gamma^{i-j}(w_i^T w_j)^2} \\ s.t. \quad & w_i^\top w_i = 1, \end{aligned} \tag{16}$$

where $\gamma \in (0, 1)$ is a weight parameter which is used for presenting the correlation between w_i and the earlier-computed projection vectors, and η is a tradeoff parameter. Finally, our sequential objective function in Eq.(16) can be rewritten as:

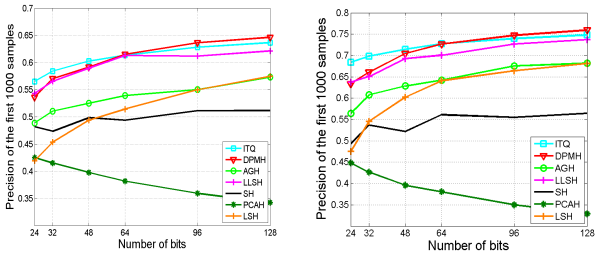
$$\begin{aligned} \max_{w_i} \quad & \frac{w_i^T\mathbf{S}_t w_i}{w_i^T\mathbf{S}_w^{(i)} w_i} \\ s.t. \quad & w_i^\top w_i = 1, \end{aligned} \tag{17}$$

where $\mathbf{S}_w^{(i)} = \mathbf{S}_w + \eta\sum_{j=1}^{i-1}\gamma^{i-j}w_j w_j^T$.

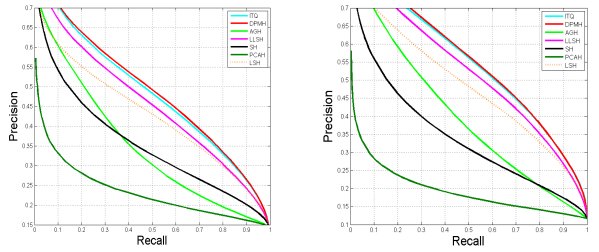
By solving (17), we can get the projection directions $W^* = [w_1, ..., w_K] \in R^{m\times K}$ which map the dataset $X \in R^{n\times m}$ to K-dimensional dataset XW^*. By combining the first step, $\mathbf{W} = W_{pca}W^* \in R^{d\times K}$ is the final projection matrix we want. The whole procedure of DPMH is outlined in Algorithm 1.

3.4. Complexity Analysis

The computational cost of our DPMH contains two parts, the offline and online part. The offline calculation mainly contains three phases : the first phase is K-means, the second phase is PCA and the last phase is solving DPMH. The time complexity of K-means and PCA are $O(unt)$ (t is the number of iterations) and $O(min(n^2K, nK^2))$. In order to derive matrices $\mathbf{B}$, $\mathbf{S}$, $\mathbf{P}$, the time complexity of the matrices calculations is $O(n^2)$. Thus, the time complexity of the offline part is $O(unt + min(n^2K, nK^2) + n^2)$. Although it is time-consuming to obtain, the calculation of matrices $\mathbf{B}$, $\mathbf{P}$ can be parallelized. The online part is the learning procedure. In this sequential learning procedure, matrix inversion and eigenvalue decomposition both cost $O(m^3)$ ($m < d$), then the total

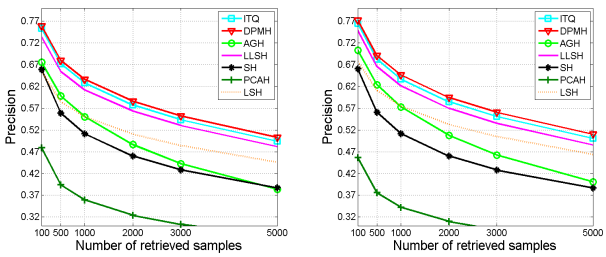


(a) CIFAR-10 (b) STL-10

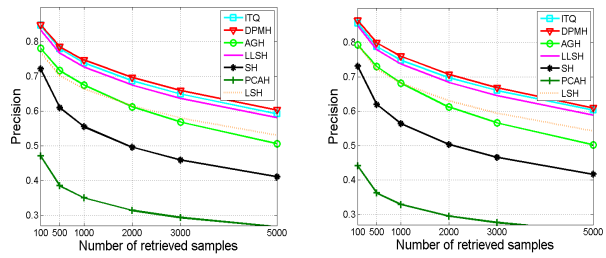


(c) CIFAR-10 (d) STL-10

Fig. 3. Experimental results on on CIFAR-10 datasets and STL-10 datasets. a) and b): Precision of the first 1000 retrieved samples; c) and d): The precision-recall results at 128 bits on CIFAR-10 datasets and STL-10 datasets.



(a) Precision (96 bits) (b) Precision (128 bits)



(c) Precision (96 bits) (d) Precision (128 bits)

Fig. 4. Precision curves on CIFAR-10 dataset and STL-10 dataset at different number of retrieved samples with 96, 128 bits respectively.

time complexity of this online part is $O(K(m^3 + Km^2))$. Hence, this algorithm is fairly scalable since its time complexity only relies on the dimensionality m and the number of bits K.

4. EXPERIMENTS

In this section, we evaluate our method on two large scale datasets in our experiment.

CIFAR-10: It consists of 60,000 32×32 color images in 10 classes which are manually labeled with 6,000 images per class.[1] The entire dataset is partitioned into two parts: 50,000 images for training and 10,000 for testing. This dataset has been used in [8,18]. We choose 50,000 images as our training set and 1,000 from 10,000 test images as our test set, then convert original images to 512-dim GIST feature [16].

STL-10: It is an image recognition dataset, which has 10 classes, 5,000 training images, 8,000 test images and 100,000 unlabeled images.[2] In our experiment, images are represented as 384-dimensional grayscale GIST descriptors [16]. This dataset has been used in [8]. We choose 100,000 images for training and 1,000 for testing.

Hashing methods can be divided into three categories: supervised, semi-supervised and unsupervised. For fair comparison, we compare the proposed DPMH with the following state-of-the-art methods : PCAH [14], LSH, SH, and LLSH (we use the variant of sequential projection learning hashing), AGH [17] (we use its two-layer hash functions to generate hash bits), ITQ [18]. As in dimension reduction algorithms such as ISOMAP, LLE, and Laplacian Eigenmap, how to set parameters k_1 and k_2 is still an open problem. In our experiments, we set the parameters k_1 and k_2 just as the method MFA [11] did. For our method, we empirically sample five values of k_1 between two and ($min_c\{n_c\}$-1) and choose the best k_2 between 200 and $5n_c$ at sampled intervals of 20. The other parameters are given as t = 10, u = 400, $\eta = 10^3$, and γ = 0.9.

For quantitative evaluation, the retrieval results are assessed with Hamming ranking. The precision of the first 1000 retrieved samples with different hash bits is recorded. Fig.3 (a) and (b) show the results in details. Fig.3 (c) and (d) show the precision-recall results at 128 bits on two datesets. Fig.4 illustrates the precision curves with different number of retrieved samples at 96, 128 bits respectively. From these figures, we find that our method achieves accuracy comparable to ITQ. Our method performs better especially for higher bits because of relaxation of the orthogonality constraints, while the improvement of performance of other methods including ITQ is limited with higher bits. LLSH, which adopts a sequential procedure to relax the orthogonality constrains, is always a better method than other methods except ITQ. But in most cases, DPMH is superior to LLSH. We also can see that our method significantly outperforms SH and AGH which on-

[1] http://www.cs.toronto.edu/ kriz/cifar.html
[2] http://www.stanford.edu/ acoates/stl10/

ly preserve the neighborhood relationships. PCAH has almost the worst performance, since the maximum variance projection can hardly preserve locality structure. Our method, which not only preserves local similarity but also forces dissimilar data apart in the low-dimensional space, achieves the best results in most cases.

5. CONCLUSIONS

In order to capture meaningful local information, most existing hashing methods have been developed to preserve the neighbor relationships while ignoring the non-neighbor relationships. In this paper, we propose a novel hashing method called DPMH which takes advantage of the neighbor and non-neighbor points to learn hash codes. The experimental results on two large-scale benchmark datasets show better performance of DPMH over the state-of-the-art methods. Our approach which uses linear hash functions can be generalized to other hash functions, such as kernel hash functions or multi-layers neural nets. On the other hand, although our method is entirely unsupervised in this work, it can be easily extended to semi-supervised or supervised scenarios for semantic neighbor search.

6. ACKNOWLEDGEMENTS

This work is supported by NSFC (No.61272247, 60873133), 863 (No.2008AA02Z310, SS2015AA020501) in China and the Science and Technology Commission of Shanghai Municipality (Grant No.13511500200).

7. REFERENCES

[1] Jon Louis Bentley, "Multidimensional binary search trees used for associative searching," *Communications of the ACM*, vol. 18, no. 9, pp. 509–517, 1975.

[2] Jeffrey K Uhlmann, "Satisfying general proximity/similarity queries with metric trees," *Information processing letters*, vol. 40, no. 4, pp. 175–179, 1991.

[3] Aristides Gionis, Piotr Indyk, Rajeev Motwani, et al., "Similarity search in high dimensions via hashing," in *VLDB*, 1999, vol. 99, pp. 518–529.

[4] Brian Kulis and Kristen Grauman, "Kernelized locality-sensitive hashing for scalable image search," in *IEEE 12th International Conference on Computer Vision*. IEEE, 2009, pp. 2130–2137.

[5] Roy De Maesschalck, Delphine Jouan-Rimbaud, and Désiré L Massart, "The mahalanobis distance," *Chemometrics and intelligent laboratory systems*, vol. 50, no. 1, pp. 1–18, 2000.

[6] Mayur Datar, Nicole Immorlica, Piotr Indyk, and Vahab S Mirrokni, "Locality-sensitive hashing scheme based on p-stable distributions," in *Proceedings of the twentieth annual symposium on Computational geometry*. ACM, 2004, pp. 253–262.

[7] Yair Weiss, Antonio Torralba, and Robert Fergus, "Spectral hashing.," in *NIPS*, 2008, vol. 9, p. 6.

[8] Kang Zhao, Dengxiang Liu, and Hongtao Lu, "Local linear spectral hashing," in *Neural Information Processing*. Springer, 2013, pp. 283–290.

[9] Charless Fowlkes, Serge Belongie, Fan Chung, and Jitendra Malik, "Spectral grouping using the nystrom method," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 2, pp. 214–225, 2004.

[10] Ian Jolliffe, *Principal component analysis*, Wiley Online Library, 2005.

[11] Shuicheng Yan, Dong Xu, Benyu Zhang, Hong-Jiang Zhang, Qiang Yang, and Stephen Lin, "Graph embedding and extensions: a general framework for dimensionality reduction," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 1, pp. 40–51, 2007.

[12] X Niyogi, "Locality preserving projections," in *Neural information processing systems*, 2004, vol. 16, p. 153.

[13] Fan RK Chung, *Spectral graph theory*, vol. 92, American Mathematical Soc., 1997.

[14] Jun Wang, Sanjiv Kumar, and Shih-Fu Chang, "Semi-supervised hashing for scalable image retrieval," in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2010, pp. 3424–3431.

[15] Saehoon Kim, Yoonseop Kang, and Seungjin Choi, "Sequential spectral learning to hash with multiple representations," in *Computer Vision–ECCV*, pp. 538–551. Springer, 2012.

[16] Aude Oliva and Antonio Torralba, "Modeling the shape of the scene: A holistic representation of the spatial envelope," *International journal of computer vision*, vol. 42, no. 3, pp. 145–175, 2001.

[17] Wei Liu, Jun Wang, Sanjiv Kumar, and Shih-Fu Chang, "Hashing with graphs," in *Proceedings of the 28th International Conference on Machine Learning*, 2011, pp. 1–8.

[18] Yunchao Gong and Svetlana Lazebnik, "Iterative quantization: A procrustean approach to learning binary codes," in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2011, pp. 817–824.