Signal Processing 93 (2013) 2256–2264

Contents lists available at SciVerse ScienceDirect

Signal Processing

journal homepage: www.elsevier.com/locate/sigpro

Dual local consistency hashing with discriminative projections selection

Peng Li, Jian Cheng*, Hanqing Lu

National Laboratory of Pattern Recognition, Institute of Automation Chinese Academy of Sciences, No. 95, Zhongguancun East Road, Beijing 100190, China

ARTICLE INFO

Article history:
Received 16 December 2011
Received in revised form
23 April 2012
Accepted 30 May 2012
Available online 13 June 2012

Keywords:
Image search
Hashing
Dual local consistency
PCA
Discriminative projection selection

ABSTRACT

Semantic hashing is a promising way to accelerate similarity search, which designs compact binary codes for a large number of images so that semantically similar images are mapped to close codes. Retrieving similar neighbors is then simply accomplished by retrieving images that have codes within a small Hamming distance of the code of the query. However, most of the existing hashing approaches, such as spectral hashing (SH), learn the binary codes by preserving the global similarity, which do not have full discriminative power. In this paper, we propose a dual local consistency hashing method which not only makes the similar images have the same codes but also dissimilar images with different codes. Moreover, we propose a PCA projection selecting scheme that choose the most discriminative projection for each bit of the codes. Therefore, the binary codes learned by our approach are more powerful and discriminative for similarity search. Extensive experiments are conducted on publicly available datasets and the comparison results demonstrate that our approach can outperform the state-of-art methods.

1. Introduction

With the rapid evolution and development of the Internet, the visual contents (e.g. image, video) on web are explosively growing. In order to efficiently exploit such enormous web contents, fast search in large scale has become one of the most critical techniques. It is of great importance to many applications, such as image or video annotation [27,28], social image retrieval [29]. The traditional content-based image retrieval techniques usually adopt exhaustive comparing the query image with pooled database, which is infeasible because the linear complexity is not scalable in practical situations. For example, the photo sharing website Flickr has over 4 billion images. Another visual content sharing website YouTube receives more than 20 h of uploaded videos per minute. Besides, most large-scale content-based image retrieval applications suffer from the curse of dimensionality since visual descriptors usually have hundreds or even thousands of dimensions. Therefore, beyond the infeasibility of exhaustive search, storage of the original data also becomes a challenging problem.

Over the past decades, several Approximate Nearest Neighbor (ANN) search techniques have been developed for large scale applications. Although there exist many tree-based methods [2,6,12,16,22] that can perform similarity search effectively in a low-dimensional feature space, for high-dimensional cases and applications with memory constrains, hashing-based ANN techniques have attracted more attention. Hashing-based methods are promising in accelerating similarity search for their capability of generating compact binary codes for a large number of images in the dataset so that similar images will have close binary codes. Retrieving similar neighbors is then accomplished simply by finding the images that have codes with a small Hamming distance from the query. It is extremely fast to

* Corresponding author.
E-mail address: jcheng@nlpr.ia.ac.cn (J. Cheng).

0165-1684/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.sigpro.2012.05.035

perform similarity search over such binary codes [23], because (1) the encoded data are highly compressed and thus can be loaded into the main memory; (2) the hamming distance between two binary codes can be computed efficiently by using bit XOR operation and computing the number of set bits (an ordinary PC today would be able to do millions of Hamming distance computation in just a few milliseconds).

Many hashing algorithms have been proposed to address the fast retrieval issue in recent years. These hashing-based methods for fast image retrieval can be considered as embedding high dimensional feature vectors to a low dimensional Hamming space, while retaining as much as possible the semantic similarity structure of data. In terms of if labeled data is needed, hashing methods can be roughly divided into two categories: unsupervised methods [1,31,33] and supervised methods [9,21,25,26]. Unsupervised methods, such as Locality Sensitive Hashing (LSH) [1] and spectral hashing (SH) [31], only use the unlabeled data to generate binary codes for given samples, while supervised methods which incorporate the label information, such as Restricted Boltzmann Machines (RBMs) [9] and Sequential Projection Learning for Hashing (SPLH) [26], are able to preserve the semantic similarity and thus facilitate semantic retrieval and classification.

Although the existing hashing methods have shown success in large-scale image search, there are some shortcomings of them. For example, the traditional spectral hashing (SH) [31] and its extension self-taught hashing (STH) [33] learn the binary codes by making the similar images close to each other in the Hamming space. However, the codes generated in this way may not have the full discriminative power for retrieval and classification task, because a good embedding should not only make the similar images close but also make the dissimilar images far away. In addition, a common step in many binary coding methods [25,31] is performing principal component analysis (PCA) to get the projection vectors. However, the projections learned in this way do not have the complementary capability, i.e., the errors generated by the previous projection should be corrected as many as possible by the following projection. Therefore, the learned binary codes also lack discriminative power to some extent.

In order to address the above-mentioned problems, we propose a novel discriminative hashing approach based on dual local consistency, which does not only make the similar images have the same codes but also make the dissimilar images have different codes. Moreover, we adopt a more discriminative projection selecting scheme, in which we select the PCA projection for each bit of the codes sequentially so that the current projection can correct the errors generated by the previous projection as many as possible.

The rest of the paper is organized as follows. Section 2 reviews related work. Section 3 describes our dual local consistency hashing approach with discriminative projections selection. In Section 4, we introduce the experiments on two publicly available datasets. Finally, conclusions are given in Section 5.

2. Related work

2.1. Fast similarity search

Given a set of data points, the objective in nearest neighbor search is to find a subset that is most similar to the query. Exhaustively comparing the query with each sample in the database is computationally infeasible because linear complexity is not acceptable for large-scale database. To avoid excessive computational and memory costs, an Approximate Nearest Neighbor (ANN) search is more appealing than exhaustive comparing with sub-linear query complexity [20].

There has been extensive research on fast similarity search due to its central importance in many applications. For a low-dimensional feature space, similarity search can be carried out efficiently by some tree-based methods (e.g. KD-tree, M-tree, cover tree and metric tree). These methods usually partition the data space recursively to implement an exact similarity search in the low-dimension feature space. For example, the KD-tree method pre-builds space-partitioning index structures, and the R-tree method pre-defines data-partitioning index structures [4]. However, the tree-based methods can degenerate their complexity to a linear scan in the worst case while attempting to speed up the computation of the similarity search. Moreover, in terms of achieving exact results, the tree-based similarity search methods do not perform better than the naive method, that is, a linear scan of the entire dataset, when the number of dimensions of the feature space is slightly high (e.g. >10) [30]. Thus, they will encounter difficulties in practical applications where the number of the dimensions can be hundreds or even thousands. Nevertheless, if the complete exactness of results is not really necessary, similarity search in a high-dimensional space can be dramatically speeded up by using hashing-based methods which are purposefully designed to approximately answer queries in virtually constant time [23]. In addition, the storage is also substantially reduced as they usually store only compact binary codes for each data point.

2.2. Hashing-based image indexing

Many hashing algorithms have been developed in recent years. One of the most popular methods is Locality Sensitive Hashing (LSH) [1]. Given a similarity metric S in a feature space, the LSH algorithm typically guarantees the probability for any two samples x_i and x_j falling into the same bucket to be $S(x_i,x_j)$, known as the "locality sensitive" property. One popular method in LSH is to generate a random vector h from a particular probabilistic distribution, e.g. p-stable distribution [5] for ℓ^p-metric space. However, since the random vector is data-independent, LSH may lead to quite inefficient codes in practice [18,31] as it requires multiple tables with long codes [7]. In order to overcome this problem, several recently proposed hashing techniques attempt to apply machine learning approaches rather than random projections to find good data-aware hash functions. In [18], Salakhutdinov et al. show that stacked Restricted

Boltzmann Machines (RBMs) [8,9] are able to generate much more compact binary codes. The RBMs model is trained with two stages: an unsupervised pre-training stage and a supervised fine-tuning. The greedy pre-training is progressively executed layer by layer from input to output. After achieving the convergence of the parameters of a layer via contrastive divergence, the derived activation probabilities are fixed and treated as input to derive the training of the next layer. In the fine-tuning stage, the labeled data are used to help refine the trained network through back-propagation. Then, the network weights are refined to maximize this objective function through gradient descent. RBMs-based binary encoding involves the estimation of a large number of weights. For example, the RBMs architecture used in [24] has four layers of hidden units, with sizes 512-512-256-32, which requires a total of 663 552 weights to learn. This does not only involve an extremely costly training procedure but also demand sufficient training data for fine-tuning. Researchers have also tried the boosting approach, such as the Similarity Sensitive Coding (SSC) algorithm [21]. They first train AdaBoost [19] classifiers with similar pairs of data points as positive examples (and also non-similar pairs as negative examples in SSC), and then take the output of all weak learners on a given data point as its binary code. In [24], both RBMs and boosting SSC are found to work significantly better than LSH when applied to a database containing tens of millions of images. In [31], a new technique called Spectral Hashing (SH) is proposed based on spectral graph partitioning [3]. SH calculates the bits by thresholding a subset of eigenvectors of the Laplacian of the similarity graph and it has demonstrated significant improvements over many other methods in terms of the number of bits required to find good similar neighbors. As an extension of SH, Self-Taught Hashing (STH) is proposed in [33], which learns the hash functions via SVM for the unseen data points. Liu et al. [13] propose a scalable graph-based unsupervised hashing approach, in which Anchor Graph is used to overcome the computationally prohibitive step of building the graph Laplacian. A Semi-Supervised Hashing (SSH) is proposed in [25], which learns hash functions that minimize the error on the labeled training data while maximally satisfying the desirable properties of hashing. They then further introduce a method to sequentially learn the hash functions [26]. Mu et al. [15] propose a hashing algorithm named LAMP with kernel tricks and weak supervision, which is formulated with a regularized maximum margin framework. In [10], the authors use query-adaptive ranking to address the issue that a large number of images sharing equal Hamming distances to a query. Reconfigurable hashing is proposed in [14], which constructs a large hash pool by one-off data indexing and then selects most effective hashing-bit combination at runtime.

3. The proposed approach

In this section, we will introduce dual local consistency hashing, which do not only make the similar images have similar binary codes but also dissimilar images with different binary codes. In addition, we introduce a more effective projection selecting scheme that can sequentially choose the most discriminative projection for each hash function. Therefore, the binary codes learned by our method perform much better than the other approaches.

3.1. The problem formulation

Assume the database $\mathbf{X}$ consists of N data points $\{\mathbf{x}_i\}_{i=1}^N, \mathbf{x}_i \in R^d$. A hashing method adopts K hash functions to map a data point $\mathbf{x}_i$ to a K-bit hash code $H(\mathbf{x}_i) = [h_1(\mathbf{x}_i), \ldots, h_K(\mathbf{x}_i)]^T$, where each hash function maps the data point to a single bit $h_k(\mathbf{x}_i) \in \{-1, 1\}$. Let $\mathbf{X}$ to be normalized to have zero mean and given a vector $\mathbf{w}_k \in R^d$, the k-th hash function can be defined as $h_k(\mathbf{x}_i) = \mathrm{sgn}(\mathbf{w}_k^T \mathbf{x}_i)$.

As is introduced in [31], spectral hashing (SH) learns the hash functions from the data by minimizing the following objective function:

$$\min \sum_{ij} S_{ij} \|H(\mathbf{x}_i) - H(\mathbf{x}_j)\|^2 \tag{1}$$

s.t.

$$H(\mathbf{x}_i) \in \{-1, 1\}^K,$$

$$\sum_i H(\mathbf{x}_i) = \mathbf{0},$$

$$\frac{1}{N} \sum_i H(\mathbf{x}_i) H(\mathbf{x}_i)^T = \mathbf{I},$$

where S_{ij} is the similarity between data $\mathbf{x}_i$ and $\mathbf{x}_j$. As we can see, SH attempts to make the similar images keep close to each other in the Hamming space. However, a good graph embedding should not only involve a similarity graph which characterizes the favorite relationship among the data points, but also involve a penalty graph which characterizes the unfavorable relationship among the data points [32]. Let D_{ij} denote the dissimilarity between data $\mathbf{x}_i$ and $\mathbf{x}_j$, then a discriminative hashing approach should satisfy the following dual local consistency criteria:

$$\begin{cases} \max \sum_{ij} D_{ij} \|H(\mathbf{x}_i) - H(\mathbf{x}_j)\|^2, \\ \min \sum_{ij} S_{ij} \|H(\mathbf{x}_i) - H(\mathbf{x}_j)\|^2. \end{cases} \tag{2}$$

It means that (1) similar images should be mapped near in the Hamming space; (2) dissimilar images should be mapped far away in the Hamming space. Based on the above principles, we propose our hashing approach by maximizing the following objective function:

$$J(H) = \sum_i \left\{ \sum_{j \in N_1(x_i)} S_{ij} H(\mathbf{x}_i)^T H(\mathbf{x}_j) - \sum_{j \in N_2(x_i)} D_{ij} H(\mathbf{x}_i)^T H(\mathbf{x}_j) \right\}, \tag{3}$$

where $H(\mathbf{x}_i)^T H(\mathbf{x}_j) = \sum_{k=1}^K h_k(\mathbf{x}_i) h_k(\mathbf{x}_j)$ reflects the similarity of the two hash codes, $N_1(x_i)$ and $N_2(x_i)$ are two subsets which contain the images similar and dissimilar to x_i respectively. Unlike the existing approaches, such as SH, that aim to preserve the global similarity structure of all image pairs, we focus on the dual local consistency for each image. Straightforwardly, our objective function

guarantees that if the given images are similar to a certain image, they should be projected to the same hash codes, otherwise they will be mapped to different hash codes.

In our method, the $N_1(x_i)$ and $N_2(x_i)$ are constructed according to the image labels. Given a labeled image set $\mathbf{X}_l$, $N_1(x_i)$ contains the images with the same labels as x_i while $N_2(x_i)$ contains the images with different labels from x_i. Then, the matrix $\mathbf{S}$ and $\mathbf{D}$ can be defined as follows:

$$S_{ij} = 1 \quad \textit{if } j \in N_1(x_i) \textit{ and } 0, \textit{ otherwise};$$

$$D_{ij} = 1 \quad \textit{if } j \in N_2(x_i) \textit{ and } 0, \textit{ otherwise}.$$

In order to make the learned hash functions subject to the constraint in Eq. (1), we propose to maximize the variance of the projected data on the whole training set $\mathbf{X}$ [25]

$$\max_{\mathbf{W}} \mathtt{tr}[\mathbf{W}^T\mathbf{X}\mathbf{X}^T\mathbf{W}], \tag{4}$$

where $\mathbf{W} = [\mathbf{w}_1, \ldots, \mathbf{w}_K]$ is a $d \times K$ matrix. By relaxing $\mathrm{sgn}(\mathbf{w}^T\mathbf{x})$ to the signed magnitude $\mathbf{w}^T\mathbf{x}$ in Eq. (3), and combining the regularization term Eq. (4), we can get the final objective function

$$\begin{aligned}\widehat{J}(W) &= \mathtt{tr}[\mathbf{W}^T\mathbf{X}_l(\mathbf{S}-\mathbf{D})\mathbf{X}_l^T\mathbf{W} + \lambda\mathbf{W}^T\mathbf{X}\mathbf{X}^T\mathbf{W}] \\ &= \mathtt{tr}[\mathbf{W}^T\mathbf{M}\mathbf{W}],\end{aligned} \tag{5}$$

where $\mathbf{M} = \mathbf{X}_l(\mathbf{S}-\mathbf{D})\mathbf{X}_l^T + \lambda\mathbf{X}\mathbf{X}^T$. Parameter λ trades off the effects of the supervised data and the regularizer. The optimal solution $\widehat{\mathbf{W}} = \arg\max_{\mathbf{W}} J(\mathbf{W})$ can be easily obtained by adding the orthogonal constraint $\mathbf{W}^T\mathbf{W} = \mathbf{I}$ which guarantees the bits be uncorrelated, and the projections $\mathbf{W}$ correspond to the top K eigenvectors of $\mathbf{M}$.

3.2. Discriminative projection selection

Although the above problem can be solved in a single shot, such solution does not have the sequential error correcting property. In [26], the authors propose a sequential learning method, in which the projections are learned sequentially and each hash function tries to correct the errors made by the previous one. However, [26] just simply select the first eigenvector as the projection for each bit, which may not be the most effective one in practice. In this section, we introduce a novel projection selecting scheme, so that the most discriminative projection can be chosen for each bit and thus the final learned codes will be more effective.

The idea of our projection selection scheme is intuitive and easy to understand. We learn the hash functions iteratively: first, the similarity matrix $\mathbf{S}$ and dissimilarity matrix $\mathbf{D}$ are updated by imposing higher weights on the image pairs misclassified by the previous hash function; then, we extract the current hash function that corrects the most errors made by the previous one from the top L eigenvectors.

We define $\mathbf{G}^k = \mathbf{X}_l^T\mathbf{w}_k\mathbf{w}_k^T\mathbf{X}_l$, which measures the sign magnitude of pairwise relationships of the k-th projection of $\mathbf{X}_l$. If $G_{ij}^k > 0 (< 0)$, it means that image $\mathbf{x}_i$ and $\mathbf{x}_j$ are considered as similar (dissimilar) pairs by the k-th projection $\mathbf{w}_k$. The magnitude of G_{ij}^k reflects the degree of its confidence. By comparing $\mathbf{G}^k$ with the groundtruth logical relationship, we can get the updating rules for $\mathbf{S}$ and $\mathbf{D}$ as follows:

$$S_{ij}^{k+1} = \begin{cases} S_{ij}^k - \alpha G_{ij}^k & \mathtt{if}\ S_{ij}G_{ij}^k < 0, \\ S_{ij}^k & \mathtt{otherwise}, \end{cases} \tag{6}$$

$$D_{ij}^{k+1} = \begin{cases} D_{ij}^k + \alpha G_{ij}^k & \mathtt{if}\ D_{ij}G_{ij}^k > 0, \\ D_{ij}^k & \mathtt{otherwise}, \end{cases} \tag{7}$$

where α is a step size parameter. $S_{ij}G_{ij}^k < 0$ indicates that the similar image pairs are projected to the dissimilar hash codes while $D_{ij}G_{ij}^k > 0$ indicates that the dissimilar image pairs are projected to the same hash codes. For both of the cases, we increase the weights of the relevant image pairs. From Eqs. (6) and (7) we can see that only the magnitude of S_{ij}^k and D_{ij}^k is changing while the sign remains stable, thus it does not change the logical relationship of the images pairs.

Algorithm 1. Discriminative projections selecting for hashing based on dual local consistency.

Input: data $\mathbf{X}$, labeled data $\mathbf{X}_l$, local semantic matrix $\mathbf{S}$,$\mathbf{D}$, length of hash codes K, number of hash projections in each iteration L, parameters λ,α.
Output: hash projections $\mathbf{W} = [\mathbf{w}_1, \ldots, \mathbf{w}_K]$
Initialize the weight matrix $\mathbf{S}_1 = \mathbf{S}$, $\mathbf{D}_1 = \mathbf{D}$, $\mathbf{G}^0 = \mathbf{0}$
for $k = 1$ **to** K **do**
  Compute $\mathbf{M}_k$ as follows:
    $\mathbf{M}_k = \mathbf{X}_l(\mathbf{S}_k - \mathbf{D}_k)\mathbf{X}_l^T + \lambda\mathbf{X}\mathbf{X}^T$
  Extract the first L eigenvectors of $\mathbf{M}_k$: $\{\mathbf{w}_{k,m}\}_{m=1}^L$
  for $m = 1$ **to** L **do**
    Compute the score of $\mathbf{w}_{k,m}$ as Eq. (8)
  end for
  Sort the L scores and set the eigenvector with the highest score as $\mathbf{w}_k$
  Update the wight matrix by Eqs. (6) and (7)
  Compute the residual:
    $\mathbf{X} = \mathbf{X} - \mathbf{w}_k\mathbf{w}_k^T\mathbf{X}$
end for

With the updated weight matrix, we hope that the new projection can decrease the errors generated by the previous one. Unlike [26], which just extracts the first eigenvector in each iteration, our method attempts to select the most discriminative projection for each bit, because the first eigenvector may not be the most suitable one for generating a discriminative code. As is introduced above, $SG^k < 0$ and $DG^k > 0$ indicate the wrongly predicted relationships of the image pairs by the k-th projection, in contrast, $SG^k > 0$ and $DG^k < 0$ indicate the correctly classified image pairs. In the $k+1$-th iteration, we compute $\mathbf{M}^{k+1}$ with the new weight matrix and extract its first L eigenvectors $\mathbf{w}_{k+1,m}$ $(m = 1, \ldots, L)$. Then we calculate the scores of the L eigenvectors as follows:

$$\begin{aligned} score(\mathbf{w}_{k+1,m}) = \sum\sum((SG^k < 0) \odot (SG^{k+1,m} > 0) \\ + (DG^k > 0) \odot (DG^{k+1,m} < 0)), \end{aligned} \tag{8}$$

where $\odot$ denotes the element-wise multiplication. The score in fact reflects how many wrongly predicted image pairs by the k-th projection are corrected by the projection $\mathbf{w}_{k+1,m}$. We sort the scores of the L projections and select the one with highest score as $\mathbf{w}_{k+1}$. The detail of

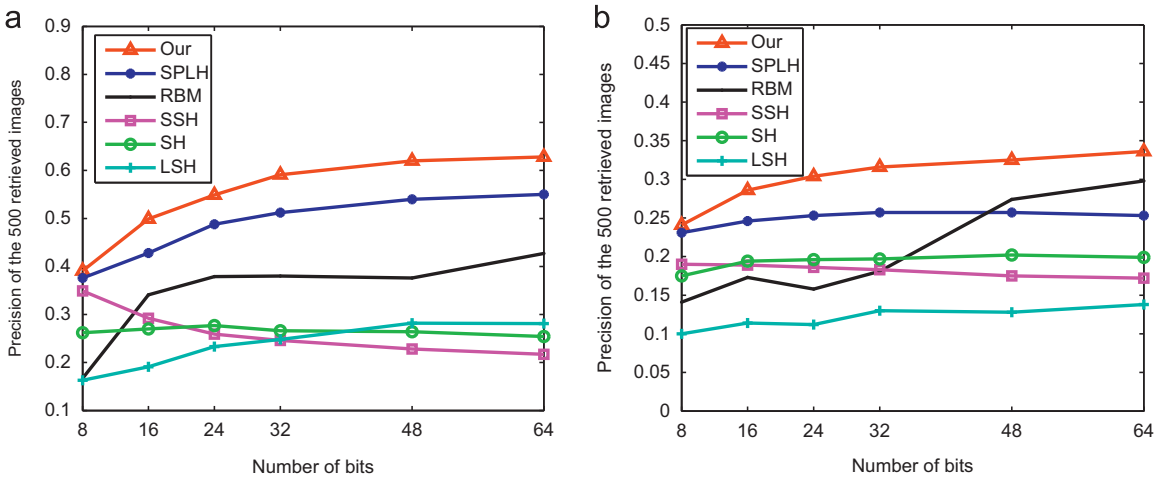

Fig. 1. Precision of image search on (a) USPS and (b) CIFAR datasets with different numbers of bits.

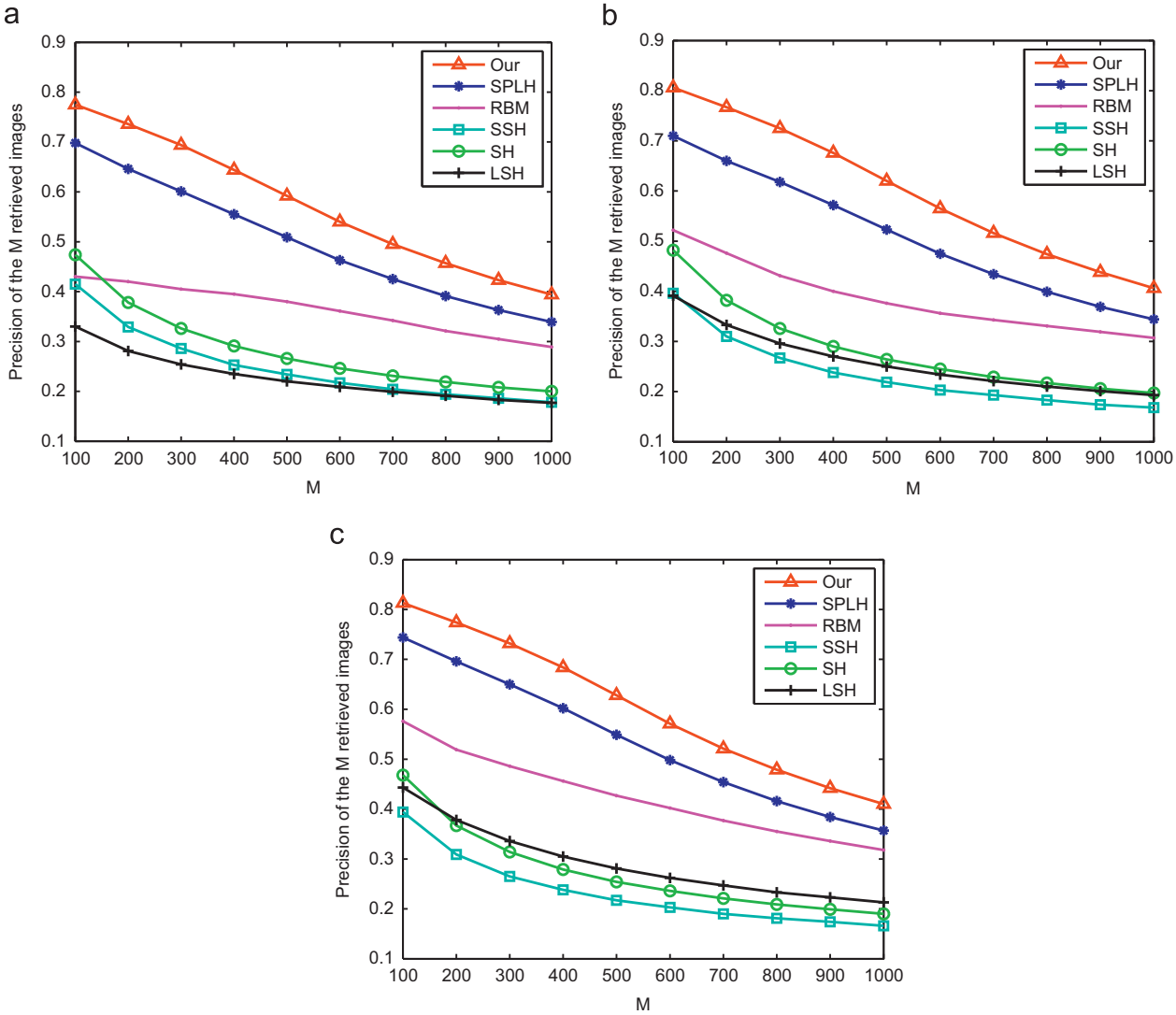

Fig. 2. Precision of the first *M* neighbors searched by different algorithms on the USPS dataset: (a) 32-bits, (b) 48-bits, (c) 64-bits.

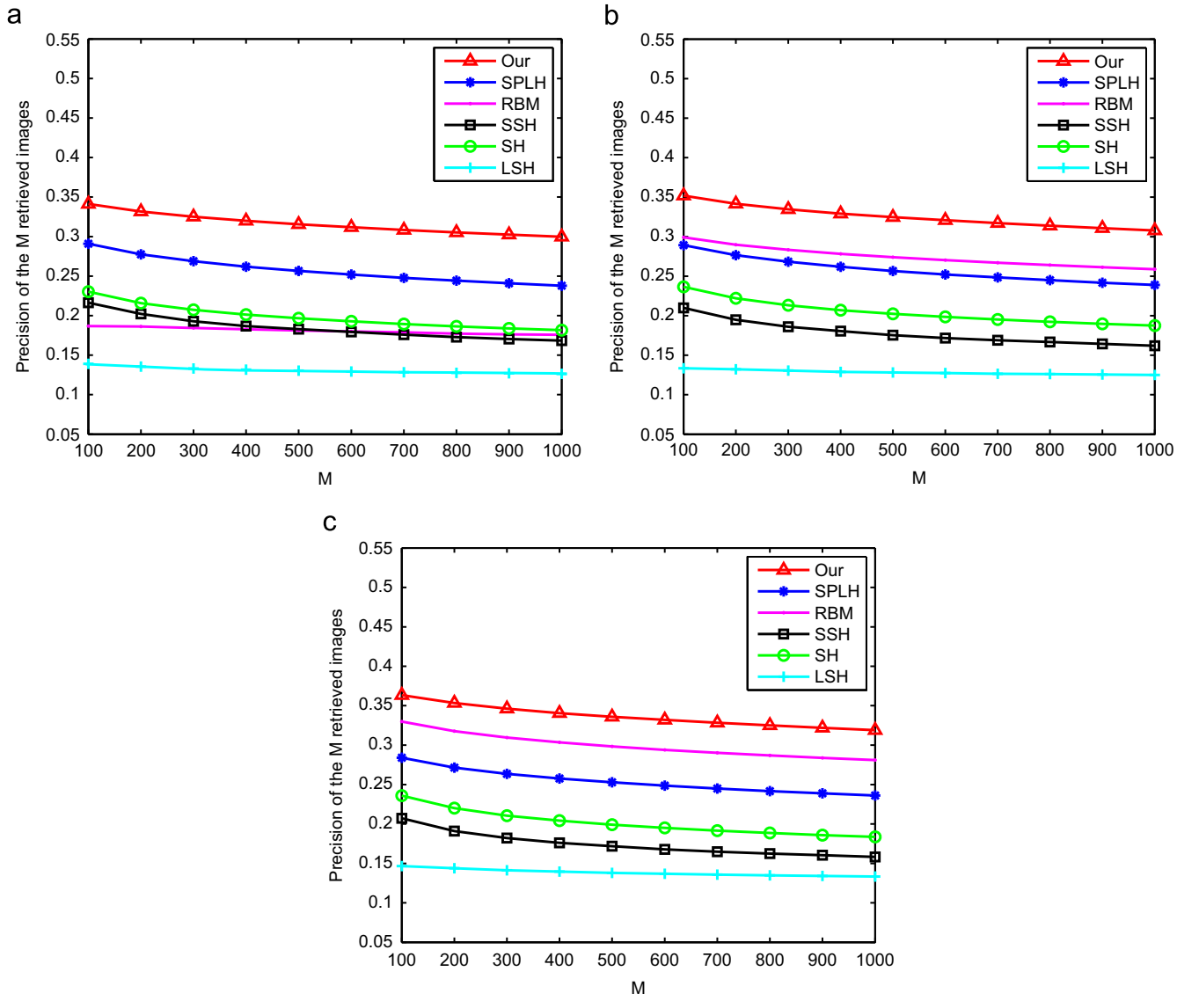

Fig. 3. Precision of the first M neighbors searched by different algorithms on the CIFAR dataset: (a) 32-bits, (b) 48-bits, (c) 64-bits.

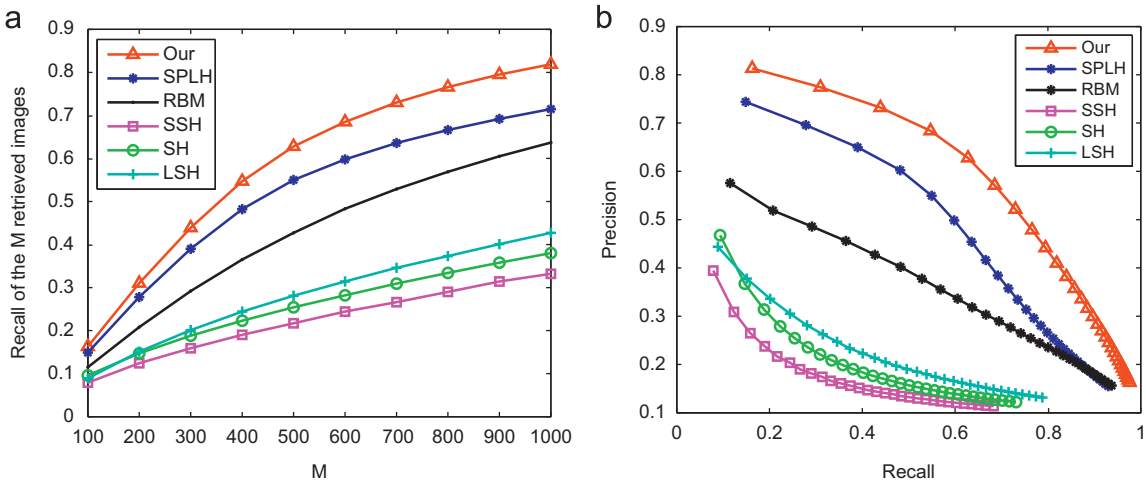

Fig. 4. (a) Recall of different numbers of retrieved images and (b) precision–recall curve on the USPS dataset (64-bits).

our algorithm is shown in Algorithm 1. Since the useful information is only contained in a few top eigenvectors, L is set to 10 in our experiments.

4. Experiments

In this section, we evaluate the proposed approach on the USPS digit and CIFAR-10 datasets. We compare its performance with five state-of-the-art binary coding methods:

1. Locality Sensitive Hashing (LSH) [1];
2. Restricted Boltzmann Machines (RBMs) [9];
3. Spectral Hashing (SH) [31];
4. Semi-Supervised Hashing (SSH) [25];
5. Sequential Projection Learning for Hashing (SPLH) [26].

4.1. Datasets and parameter settings

The USPS handwritten digit database[1] contains 10 classes (0–9 digit characters) and each class has 1100 images. Each image is 16×16 in resolution, which results in a 256-D feature vector. In our experiments, we randomly select 500 images from each class for training, which results in a training set with 5000 images. The left 6000 images are used as queries for testing. The training set is used to learn the hash functions of different hashing methods. The testing set is used to evaluate the performance of different approaches through the nearest neighbor retrieval based on the binary codes.

The CIFAR-10 dataset [11] is a subset of the Tiny Images dataset and it consists of 60 000 color images from 10 classes, with 6000 images per class. The training set contains 50 000 images, including 5000 randomly selected images from each class. The remaining 10 000 images are used as the testing set. The original Tiny images are 32×32 pixels. We represent them with GIST [17] descriptors computed at eight orientations and four different scales, resulting in a 512-D vector for each image.

About 5000 images are used labeled images in all the experiments. The sizes of the similar images set $N_1(\bullet)$ and the dissimilar images set $N_2(\bullet)$ are both set to 90. In order to make the dissimilar images set diverse and balanced, 10 images are randomly selected from each of the other nine image classes different from the given image. The parameter λ is set to 0.1 and α is set to 1.

4.2. The comparison results

Fig. 1 illustrates the performance of different algorithms for the nearest neighbor search with different numbers of the bits. We can see that our method achieves the best performance with different numbers of bits on both the USPS and CIFAR datasets. Among the different methods, the unsupervised methods LSH and SH get relatively poor performance than the other supervised methods. RBM performs well at large bits and it even outperforms SPLH on the CIFAR dataset at 48-bits and 64-bits. SSH gets good results at low bits and the performance decreases when the number of bits grows, which may be attributed to the orthogonality constraints that force one to pick those directions that have very low variance. This problem is more obvious when the number of hash bit grows. That is why SSH performs even worse than LSH and SH in the case of long codes. SPLH learns the directions sequentially which can relax the orthogonality constraint to some extent and have property of error correcting, thus it performs much better than SSH. However, it still gets lower performance than our method which is based on dual local consistency and adopts a more discriminative projection selection scheme.

The performance of all the methods, in terms of precision versus the number of retrieved images on the two datasets, is illustrated in Figs. 2 and 3, from which we can see that our method is consistently better than the other methods. SPLH also gets very good results in many cases due to its error correcting property. But RBM

[1] http://www.cs.nyu.edu/~roweis/data.html

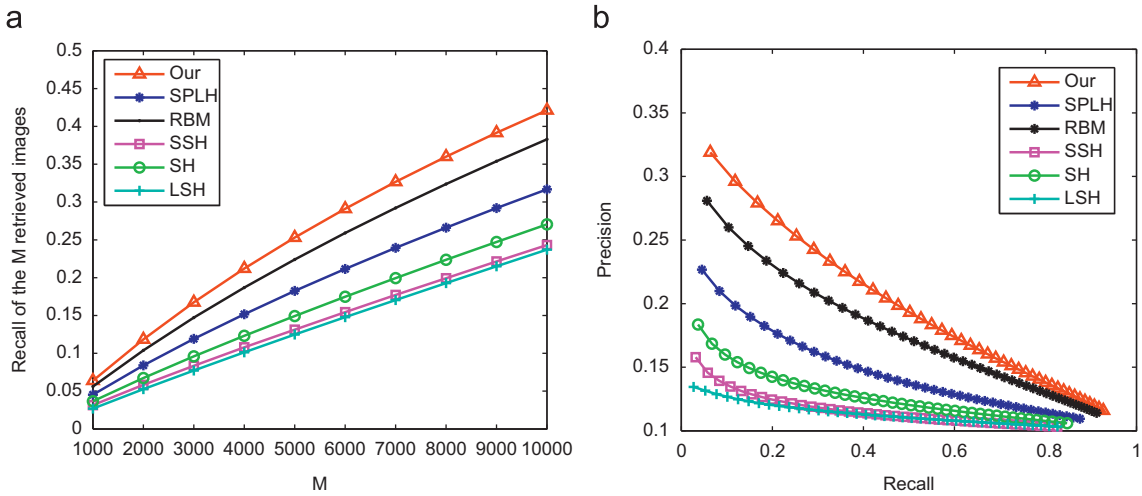

Fig. 5. (a) Recall of different numbers of retrieved images and (b) precision–recall curve on the CIFAR dataset (64-bits).

performs better than SPLH on the CIFAR dataset with large bits, because a good network can be trained with enough training images on this dataset. LSH, which randomly selects the hash hyperplanes, gets the lowest precision in most cases. Because of the data-dependent functions, SH is able to produce better hash codes than LSH and thus gets better results. Figs. 4 and 5 show the recall performance and precision–recall curve respectively on different datasets. It demonstrates that our approach has the highest score which means that our approach can return more relevant images in the fast image retrieval task than the other methods.

5. Conclusion

In this paper, we propose a novel hashing approach for fast image retrieval. Unlike many traditional hashing methods which only preserve the similarity structure of images in a global manner, our method is based on dual local consistency. In our approach, not only the similar images are projected to the same hash codes, but also the dissimilar images are projected to different hash codes. Moreover, our approach adopts a more discriminative projection selecting scheme, which can choose more discriminative projection for each hash function. Therefore, the binary codes learned by our approach are much more effective than the other methods. The extensive experimental results have shown that our approach can outperform the state-of-the-art hashing approaches.

Acknowledgement

This work was supported in part by the 973 Programm under Project 2010CB327905, by the National Natural Science Foundation of China under Grant 61170127, 60975010, 60833006, and 61070104.

References

[1] A. Andoni, P. Indyk, Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions, in: Proceedings of 47th Annual IEEE Symposium on Foundations of Computer Science, 2006, pp. 459–468.
[2] S. Arya, D.M. Mount, N.S. Netanyahu, R. Silverman, A.Y. Wu, An optimal algorithm for approximate nearest neighbor searching, Journal of ACM 45 (6) (1998) 891–923.
[3] F.R.K. Chung, Spectral Graph Theory, American Mathematical Society, 1997.
[4] T.H. Cormen, C.E. Leiserson, R.L. Rivest, C. Stein, Introduction to Algorithms, 2nd ed. MIT Press, McGraw-Hill, 2001.
[5] M. Datar, N. Immorlica, P. Indyk, V. Mirrokni, Locality-sensitive hashing scheme based on p-stable distributions, in: Proceedings of the 20th Annual Symposium on Computational Geometry, 2004, pp. 253–262.
[6] J.H. Friedman, J.L. Bentley, R.A. Finkel, An algorithm for finding best matches in logarithmic expected time, ACM Transactions on Mathematical Software (TOMS) 3 (3) (1977) 209–226.
[7] A. Gionis, P. Indyk, R. Motwani, Similarity search in high dimensions via hashing, in: Proceedings of the 25th International Conference on Very Large Data Bases (VLDB), 1999, pp. 518–529.
[8] G. Hinton, S. Osindero, Y.W. Teh, A fast learning algorithm for deep belief nets, Neural Computation 18 (7) (2006) 1527–1554.
[9] G. Hinton, R. Salakhutdinov, Reducing the dimensionality of data with neural networks, Science 313 (5786) (2006) 504–507.
[10] Y. Jiang, J. Wang, S. Chang, Lost in binarization: query-adaptive ranking for similar image search with compact codes, in: Proceedings of the 1st ACM International Conference on Multimedia Retrieval, 2011.
[11] A. Krizhevsky, Learning Multiple Layers of Features from Tiny Images, Technical Report, University of Toronto, 2009.
[12] N. Kumar, L. Zhang, S. Nayar, What is a good nearest neighbors algorithm for finding similar pathes in images? in: Proceedings of the 10th European Conference on Computer Vision, 2008, pp. 364–378.
[13] W. Liu, J. Wang, S. Kumar, S. Chang, Hashing with graphs, in: Proceedings of the 28th International Conference on Machine Learning, 2011.
[14] Y. Mu, X. Chen, T. Chua, S. Yan, Learning reconfigurable hashing for diverse semantics, in: Proceedings of the 1st ACM International Conference on Multimedia Retrieval, 2011.
[15] Y. Mu, J. Shen, S. Yan, Weakly-supervised hashing in kernel space, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2010, pp. 3344–3351.
[16] M. Muja, D.G. Lowe, Fast approximate nearest neighbors with automatic algorithm configuration, in: VISAPP International Conference on Computer Vision Theory and Applications, 2009, pp. 331–340.
[17] A. Oliva, A. Torralba, Modeling the shape of the scene: a holistic representation of spatial envelope, in: IJCV, 2001.

[18] R. Salakhutdinov, G. Hinton, Semantic hashing, International Journal of Approximate Reasoning 50 (7) (2009) 969–978.
[19] R.E. Schapire, The boosting approach to machine learning: an overview, in: Proceedings of MSRI Workshop on Nonlinear Estimation and Classification, 2002.
[20] G. Shakhnarovich, T. Darrel, P. Indyk, Nearest Neighbor Methods in Learning and Vision: Theory and Practice (Neural Information Processing), The MIT Press, 2006.
[21] G. Shakhnarovich, P.A. Viola, T. Darrel, Fast pose estimation with parameter-sensitive hashing, in: Proceedings of the 9th IEEE International Conference on Computer Vision, 2003, pp. 750–759.
[22] C. Silpa-Anan, R. Hartley, Optimised KD-trees for fast image descriptor matching, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2008, pp. 1–8.
[23] B. Stein, Principles of hash-based text retrieval, in: Proceedings of 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 2007, pp. 527–534.
[24] A.B. Torralba, R. Fergus, Y. Weiss, Small codes and large image databases for recognition, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2008, pp. 1–8.
[25] J. Wang, S. Kumar, S. Chang, Semi-supervised hashing for scalable image retrieval, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2010, pp. 3424–3431.
[26] J. Wang, S. Kumar, S. Chang, Sequential projection learning for hashing with compact codes, in: Proceedings of International Conference on Machine Learning, 2010.
[27] M. Wang, X. Hua, J. Tang, R. Hong, Beyond distance measurement: constructing neighborhood similarity for video annotation, IEEE Transactions on Multimedia 11 (3) (2009) 465–476.
[28] M. Wang, X. Hua, R. Hong, J. Tang, G. Qi, Y. Song, Unified video annotation via multi-graph learning, IEEE Transactions on Circuits and Systems for Video Technology 19 (5) (2009) 733–746.
[29] M. Wang, K. Yang, X. Hua, H. Zhang, Towards a relevant diverse search of social images, IEEE Transactions on Multimedia 12 (8) (2010) 829–842.
[30] R. Weber, H.-J. Schek, S. Blott, A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces, in: Proceedings of the 24th International Conference on Very Large Data Bases (VLDB), 1998, pp. 194–205.
[31] Y. Weiss, A.B. Torralba, R. Fergus, Spectral hashing, in: Proceedings of the Conference on Advances in Neural Information Processing Systems, 2008, pp. 1753–1760.
[32] J. Yang, S. Yan, Y. Fu, X. Li, T. Huang, Non-negative graph embedding, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2008, pp. 1–8.
[33] D. Zhang, J. Wang, D. Cai, J. Lu, Self-taught hashing for fast similarity search, in: Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2010.