Locality Preserving Hashing

Kang Zhao, Hongtao Lu and **Jincheng Mei**
Key Laboratory of Shanghai Education Commission for Intelligent Interaction and Cognitive Engineering
Department of Computer Science and Engineering, Shanghai Jiao Tong University, China
{sjtuzk, htlu, jcmei}@sjtu.edu.cn

Abstract

Hashing has recently attracted considerable attention for large scale similarity search. However, learning compact codes with good performance is still a challenge. In many cases, the real-world data lies on a low-dimensional manifold embedded in high-dimensional ambient space. To capture meaningful neighbors, a compact hashing representation should be able to uncover the intrinsic geometric structure of the manifold, e.g., the neighborhood relationships between subregions. Most existing hashing methods only consider this issue during mapping data points into certain projected dimensions. When getting the binary codes, they either directly quantize the projected values with a threshold, or use an orthogonal matrix to refine the initial projection matrix, which both consider projection and quantization separately, and will not well preserve the locality structure in the whole learning process. In this paper, we propose a novel hashing algorithm called *Locality Preserving Hashing* to effectively solve the above problems. Specifically, we learn a set of locality preserving projections with a joint optimization framework, which minimizes the average projection distance and quantization loss simultaneously. Experimental comparisons with other state-of-the-art methods on two large scale datasets demonstrate the effectiveness and efficiency of our method.

Introduction

The explosive growth of the vision data on the internet has posed a great challenge to many applications in terms of fast similarity search. To handle this problem, hashing based approximate nearest neighbor (ANN) search techniques have recently become more and more popular because of their improvements in computational speed and storage reduction.

Given a dataset, hashing methods convert each dataset item into a binary code so as to accelerate search. In many cases, the real-world data lies on a low-dimensional manifold (Niyogi 2004). To depict such a manifold, the neighborhood relationships between data points are essential. Hence, a compact hashing representation should preserve the neighborhood structure as much as possible after Hamming embedding. So far, a lot of hashing methods (Weiss, Torralba, and Fergus 2008; Wang, Kumar, and Chang 2010a; 2012; Liu et al. 2012; Zhao, Liu, and Lu 2013) have been proposed from this perspective. In these methods, a typical two-stage strategy is adopted in hashing function learning, as mentioned in (Kong and Li 2012a). In the first stage, several projected dimensions are generated in certain ways. Then in the second stage, the projected values will be quantized into binary codes. One problem with these methods is that they preserve locality structure only in the first stage. In the second stage, directly thresholding the projected values is chosen to get binary codes, which means a number of neighbor points close to the threshold will be inevitably hashed to distinct bits. Another case is shown in (Gong and Lazebnik 2011; Kong and Li 2012b; Xu et al. 2013), where an orthogonal matrix is used to refine the initial projection matrix in the second stage. However, the orthogonal transformation is just a rotation operation, which will not really "change" the projection matrix. The step-by-step learning procedure adopted by these methods considers the two stages separately during Hamming embedding.

In this paper, we propose a novel hashing algorithm named *Locality Preserving Hashing* (LPH) to solve the above problems. The main contributions of our work are outlined as follows:

(1) We take advantage of a more general method of minimizing the quantization loss. Our approach can maintain the neighborhood structure preserved in projection stage as much as possible.

(2) A joint optimization framework, which minimizes the average projection distance and the quantization loss simultaneously, is provided. To the best of our knowledge, this is one of the first work that learns hash codes with a joint optimization of projection and quantization stages together to preserve the locality structure of a dataset in Hamming space.

(3) Relaxation and an iterative algorithm along with state-of-the-art optimization techniques are proposed to solve the joint optimization problem efficiently. Experimental results show that our approach is superior to other state-of-the-art methods.

Related Work

Given a data set $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n]^\top \in \mathrm{R}^{n \times d}$, the basic idea of hashing is to map each point $\mathbf{x}_i$ to a suitable K-dimensional binary code $\mathbf{y}_i \in \{-1, +1\}^K$ with K denoting the code size. Linear projection-based hashing methods have been widely used due to their simplicity and efficiency. In linear projection-based hashing, after learning a projection matrix $\mathbf{W} = [\mathbf{w}_1, ..., \mathbf{w}_K] \in \mathrm{R}^{d\times K}$, the k^{th} hash bit y_{ik} of $\mathbf{x}_i$ can be expressed as the following form:

$$y_{ik} = sgn(f(\mathbf{w}_k^\top \mathbf{x}_i + b_k)), \tag{1}$$

where b_k is a bias, $f(\cdot)$ is an arbitrary function and $sgn(\cdot)$ is the sign function. Then the corresponding $\{0, 1\}$ code can be given by $\frac{1}{2}(1 + y_{ik})$.

Broadly, hashing methods can be roughly divided into two main categories (Gong and Lazebnik 2011; Liu et al. 2011): data-independent methods and data-dependent methods. Locality Sensitive Hashing (LSH) (Gionis et al. 1999) and its variants (Datar et al. 2004; Kulis, Jain, and Grauman 2009; Kulis and Grauman 2009) are one representative kind of data-independent methods. They use random projections to get binary codes of data. Although, there exists an approximate theoretical proof for these methods that the locality structure is asymptotically preserved in Hamming space, it requires numerous tables with long codes for high accuracy in practice (Gionis et al. 1999). Especially for large scale applications, the numerous hashing tables will cost considerable storage and query time. Besides, long codes will decrease the collision probability of similar samples, consequently resulting in low recall.

Due to the shortcomings of data-independent methods, many data-dependent methods have been developed to learn more compact hash codes from dataset. Semantic Hashing (Salakhutdinov and Hinton 2009) employs a deep generative model combined with restricted Boltzmann machine to generate hash functions. In PCA-Hashing (PCAH) (Wang et al. 2006; Gong and Lazebnik 2011), the eigenvectors of the data covariance matrix with maximum eigenvalues are used as hashing projection matrix. Spectral Hashing (SH) (Weiss, Torralba, and Fergus 2008) formulates the hashing problem as a particular form of graph partition to seek a code with balanced partitioned and uncorrelated bits. After that, some state-of-the-art hashing methods inspired by SH are put forward, including Hashing with Graphs (AGH) (Liu et al. 2011) and Harmonious Hashing (HamH) (Xu et al. 2013). AGH introduces anchor graphs to accelerate the computation of graph Laplacian, while HamH adopts a linear relaxation of the neighborhood graph and tries to maintain equivalent variance on each dimension. Isotropic Hashing (IsoH) (Kong and Li 2012b) is another hashing method proposed to seek a projection with equal variances for different dimensions by rotating the PCA-projected matrix. In (Gong and Lazebnik 2011), Iterative Quantization (ITQ) is proposed to learn an orthogonal matrix by minimizing the quantization loss of mapping the data generated by PCA projection to binary codes.

Recently, some hashing algorithms exploiting label information have been developed. By introducing semantic pairs, Semi-Supervised Hashing (SSH) (Wang, Kumar, and Chang 2010a; 2012) minimizes the empirical error on the labeled pairs and maximizes the information theoretic regularization on all data to learn hash functions. LDA-Hash (Strecha et al. 2012) learns hash codes by making the Hamming distance minimized between positive pairs and maximized between negative pairs. In (Liu et al. 2012), Kernel-Based Supervised Hashing (KSH) uses the equivalence between optimizing the code inner products and the Hamming distances to map the data to compact codes whose Hamming distances are minimized on similar pairs and simultaneously maximized on dissimilar pairs.

Locality Preserving Hashing

This section presents the formulation of our $Locality\ Preserving\ Hashing$ (LPH) method. First, we introduce the motivation of our method. Then, we discribe the deduction of LPH and formulate it as a joint optimization problem. Finally, we give an iterative algorithm along with state-of-the-art optimization techniques to solve the proposed problem. To facilitate our discussion, some notations are given below.

We aim to map the data $\mathbf{X} \in \mathrm{R}^{n\times d}$ to a Hamming space to get compact hash representations. Let $\mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_n]^\top \in \mathrm{R}^{n\times K}$ be the K-bit Hamming embedding of $\mathbf{X}$. In our work, linear projections along with thresholding are used to obtain hash bits. For every data point $\mathbf{x}_i$, the k^{th} hash bit is defined as

$$y_{ik} = sgn(\mathbf{w}_k^\top \mathbf{x}_i + b_k), \tag{2}$$

where b_k is the negative mean value of the projected data. Without loss of generality, we assume $\mathbf{X}$ is zero centered, i.e., $\sum_{i=1}^n \mathbf{x}_i = \mathbf{0}$. Thus $b_k = -\frac{1}{n}\sum_{i=1}^n \mathbf{w}_k^\top \mathbf{x}_i = 0$. One can get the corresponding $\{0, 1\}$ code as

$$h_{ik} = \frac{1}{2}(1 + y_{ik}) = \frac{1}{2}(1 + sgn(\mathbf{w}_k^\top \mathbf{x}_i)). \tag{3}$$

Motivation

As the real-world data usually lies on a low-dimensional manifold, the neighborhood structure of the manifold should be preserved to capture meaningful neighbors with hashing. Mainstream hashing methods adopt a two-stage strategy step by step. In order to make the locality preserving projection matrix learned in projection stage be well preserved in quantization stage, we adopt a more general quantization approach and formulate the two-stage learning procedure as a joint optimization problem. Apparently, if the locality structure of the dataset is well preserved in Hamming space, the true positive rate of the returned samples will be high. As a result, our goal is to learn such codes that the locality structure is well preserved after the dataset has been embedded into Hamming space.

Projection Stage

As aforementioned, for most existing hashing methods, the two-stage strategy is usually adopted to learn hash codes. In the first stage, one wants to learn a projection matrix in which the neighborhood structure is well preserved. We construct an affinity matrix $\mathbf{A}$ first, whose entry A_{ij}, representing the similarity of data $\mathbf{x}_i$ and $\mathbf{x}_j$, is defined as follows:

$$A_{ij} = \begin{cases} \exp(\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{\sigma}), & \mathbf{x}_i \in \mathcal{N}_k(\mathbf{x}_j)\ or\ \mathbf{x}_j \in \mathcal{N}_k(\mathbf{x}_i), \\ 0, & otherwise, \end{cases}$$

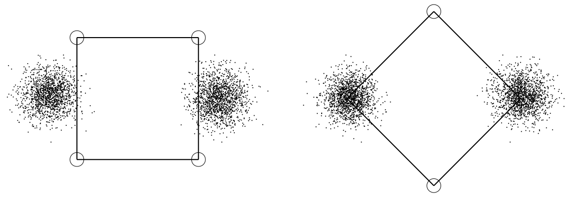


Figure 1: 2-D illustration of quantization stage. A circle denotes a vertex of the hypercube $\{-1, 1\}^2$, and a dot denotes a data point. (a) shows the bad circumstance of traditional quantization condition after projection stage; (b) presents the optimized quantization situation by minimizing the quantization loss.

where $\mathcal{N}_k(\mathbf{x})$ denotes the k-nearest neighbors of $\mathbf{x}$. Let $\{\mathbf{p}_i\}_{i=1}^n$ be the list of projection vectors, i.e., $\mathbf{p}_i = \mathbf{W}^\top \mathbf{x}_i$. Then, we minimize the following objective function (Niyogi 2004):

$$\sum_{ij} A_{ij} \|\mathbf{p}_i - \mathbf{p}_j\|^2, \tag{4}$$

Empirically, if $\mathbf{x}_i$ and $\mathbf{x}_j$ are close then $\mathbf{p}_i$ and $\mathbf{p}_j$ will be close as well (Niyogi 2004). So, the locality structure is preserved. With some algebra manipulations, we have

$$\sum_{i,j} A_{ij} \|\mathbf{p}_i - \mathbf{p}_j\|^2 = 2tr\{\mathbf{W}^\top \mathbf{X}^\top \mathbf{L} \mathbf{X} \mathbf{W}\}, \tag{5}$$

where $\mathbf{L} = \mathbf{D} - \mathbf{A}$, $\mathbf{D} = diag(\mathbf{A1})$ with $\mathbf{1} = [1, \cdots, 1]^\top \in \mathrm{R}^n$ and $tr\{\cdot\}$ is the trace of a matrix.

Quantization Stage

In the second stage, we want to obtain the hash codes from projection vector $\mathbf{p}$ while maintaining the locality preserving property of the projected data as much as possible. We found that $sgn(\mathbf{p})$ can be seen as the vertex of the hypercube $\{-1, 1\}^K$ corresponding to $\mathbf{p}$ in terms of Euclidean distance. The closer $sgn(\mathbf{p})$ and $\mathbf{p}$ are, the better the locality structure will be preserved (see Figure 1), which can be formulated as the following optimization problem:

$$\min_{\mathbf{p}_i} \sum_{i=1}^n \|sgn(\mathbf{p}_i) - \mathbf{p}_i\|^2. \tag{6}$$

The objective function can be rewritten in a compact matrix form:

$$\sum_{i=1}^n \|sgn(\mathbf{p}_i) - \mathbf{p}_i\|^2 = \|\mathbf{Y} - \mathbf{XW}\|_F^2, \tag{7}$$

where $\|\cdot\|_F$ denotes the Frobenius norm. Note that our quantization method is different from those in ITQ and HamH, where they aim to rotate the projection matrix to minimize the loss function by introducing an orthogonal matrix. By contrast, we adopt a more general way to minimize the quantization loss (7) without extra rotational operations. Besides, our method does not require a pre-learned projection matrix.

A Joint Optimization Framework

In contrast to the mainstream two-stage strategy of hashing conducted step by step, we propose a joint optimization framework to hold the locality preserving property in the two stages simultaneously. By incorporating the graph Laplacian regularization term and the quantization loss, we minimize the following joint optimization function:

$$H(\mathbf{Y}, \mathbf{W}) = tr\{\mathbf{W}^\top \mathbf{X}^\top \mathbf{L} \mathbf{X} \mathbf{W}\} + \rho \|\mathbf{Y} - \mathbf{XW}\|_F^2, \tag{8}$$

where ρ is a positive parameter controlling the tradeoff between projection and quantization stages.

By minimizing Eq.(8), the neighborhood structure of the dataset can be well preserved in just one step, which is always better than step-by-step learning since it hardly ensures the locality preserving projection matrix learned in projection stage not to be destroyed in quantization stage. In addition, as discussed above, ITQ and HamH learn an orthogonal matrix to refine the initial projection matrix to minimize the quantization loss. The orthogonal transformation just rotates the projection matrix, it can not change the intrinsic structure of the matrix. In other words, if we treat the projection matrix $\mathbf{W} \in \mathrm{R}^{d \times K}$ as K hyperplanes, the relative relationship between each hyperplane has been decided in projection stage. Consequently, it makes the quantization stage not really “work” in a certain sense. In contrast, our joint optimization framework will make $\mathbf{W}$ determined by both projection and quantization stages. It does not need a pre-learned $\mathbf{W}$ as well. As a result, the projection matrix $\mathbf{W}$ learned by our method will hold the locality preserving property in two stages simultaneously.

Motivated by SH, we would like the hash bits to be independent and generate a balanced partition of the dataset. Further, we relax the independence assumption to pairwise decorrelation of bits, then we have the following problem:

$$(\mathbf{Y}^*, \mathbf{W}^*) = \arg\min_{\mathbf{Y},\mathbf{W}} H(\mathbf{Y}, \mathbf{W}) \tag{9}$$
$$\text{subject to} : \mathbf{Y}^\top \mathbf{1} = \mathbf{0}$$
$$\mathbf{Y}^\top \mathbf{Y} = n\mathbf{I}_{K \times K},$$

where $\mathbf{I}_{K \times K}$ is the identity matrix with size of $K \times K$.

The above problem is difficult to solve since it is a typical combinatorial optimization problem which is usually NP hard. In the next section, we relax the constraints and adopt a coordinate-descent iterative algorithm to get an approximate solution.

Relaxation and Optimization

The constraints $\mathbf{Y}^\top \mathbf{1} = \mathbf{0}$ mean that each bit takes 50% probability to be 1 or -1. However, for real-world data, it is not always the case, as illustrated in Figure 2. The constraints will force one to select the red hyperplane instead of the green one. Hence, we will discard these strong constraints.

Moreover, we relax the pairwise decorrelation of bits $\mathbf{Y}^\top \mathbf{Y} = n\mathbf{I}_{K \times K}$ by imposing the constraints $\mathbf{W}^\top \mathbf{W} = \mathbf{I}_{K \times K}$ instead as in (Wang, Kumar, and Chang 2012; 2010a), which request the projection directions to be unit-norm and

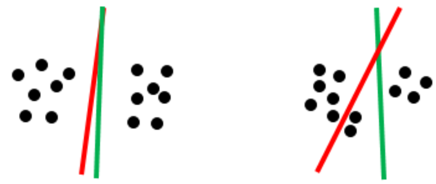


Figure 2: An illustration of splitting the data points with hashing hyperplanes. On the left panel, the red hyperplane and the green one are equivalent. However, on the right, the green hyperplane is more reasonable.

orthogonal to each other. The relaxed problem is thus expressed as

$$(\mathbf{Y}^*, \mathbf{W}^*) = \arg\min_{\mathbf{Y},\mathbf{W}} H(\mathbf{Y}, \mathbf{W}) \quad (10)$$
$$\text{subject to} : \mathbf{W}^\top \mathbf{W} = \mathbf{I}.$$

It is still not easy to be solved because of the orthogonality constraints. Fortunately, $H(\mathbf{Y}, \mathbf{W})$ is lower-bounded as Eq.(8) is always nonnegative. To seek a local minimum of Eq.(10), we employ a coordinate-descent iterative procedure. We begin the iterative algorithm with a random orthogonal initialization of $\mathbf{W}$, which provides better performance than arbitrary random initialization since it corresponds to the orthogonality constraints. In each iteration, we first fix $\mathbf{W}$ and optimize $\mathbf{Y}$, then fix $\mathbf{Y}$ and optimize $\mathbf{W}$. The details of the two alternating steps are described below.

Quantization step: fix W and optimize Y. Expanding Eq.(8), we obtain

$$\begin{aligned} H(\mathbf{Y}, \mathbf{W}) &= C_1 + \rho(\|\mathbf{Y}\|_F^2 + C_2 - 2tr\{\mathbf{Y}\mathbf{W}^\top\mathbf{X}^\top\}) \\ &= C_1 + \rho n K + \rho C_2 - 2\rho tr\{\mathbf{Y}\mathbf{W}^\top\mathbf{X}^\top\}, \end{aligned}$$

where $C_1 = tr\{\mathbf{W}^\top\mathbf{X}^\top\mathbf{L}\mathbf{X}\mathbf{W}\}$ and $C_2 = tr\{\mathbf{X}\mathbf{W}\mathbf{W}^\top \mathbf{X}^\top\}$. Because $\mathbf{W}$ and $\mathbf{X}$ are fixed, minimizing $H(\mathbf{Y}, \mathbf{W})$ with respect to $\mathbf{Y}$ is equivalent to maximizing

$$tr\{\mathbf{Y}\mathbf{W}^\top\mathbf{X}^\top\} = \sum_{i=1}^{n}\sum_{j=1}^{K} y_{ij}(XW)_{ij}, \quad (11)$$

where $(XW)_{ij}$ denotes the $(i, j)^{th}$ element in $\mathbf{XW}$. To maximize the above expression, we should have $y_{ij} = 1$ when $(XW)_{ij} \geq 0$ and $y_{ij} = -1$ otherwise. That is,

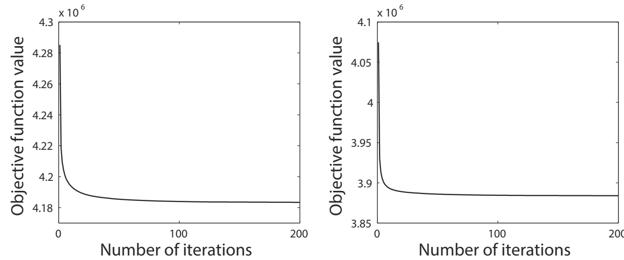
$$\mathbf{Y} = sgn(\mathbf{XW}). \quad (12)$$

Projection step: fix Y and optimize W. For a fixed $\mathbf{Y}$, our problem corresponds to a typical minimization with orthogonality constraints. A popular solution to the optimization problem is the gradient flow method on the orthogonal constraints (Helmke and Moore 1996; Ng, Liao, and Zhang 2011).

We denote $\mathcal{M}_\mu^\nu = \{\mathbf{W} \in \mathrm{R}^{\mu\times\nu} : \mathbf{W}^\top\mathbf{W} = \mathbf{I}\}$, $\mathcal{T}_\mathbf{W}\mathcal{M}_\mu^\nu = \{\mathbf{T} \in \mathrm{R}^{\mu\times\nu} : \mathbf{T}^\top\mathbf{W} + \mathbf{W}^\top\mathbf{T} = 0\}$. The $\mathcal{M}_\mu^\nu$ is usually referred to as the compact $\mu \times \nu$ Stiefel manifold (Kreyszig 1968) and $\mathcal{T}_\mathbf{W}\mathcal{M}_\mu^\nu$ is its tangent space. Generally, there are two known metrics for $\mathcal{T}_\mathbf{W}\mathcal{M}_\mu^\nu$: the Euclidean metric $\langle\mathbf{T}_1, \mathbf{T}_2\rangle_e = tr\{\mathbf{T}_1^\top\mathbf{T}_2\}$ and the canonical metric

$$\langle\mathbf{T}_1, \mathbf{T}_2\rangle_c = tr\{\mathbf{T}_1^\top(\mathbf{I} - \frac{1}{2}\mathbf{W}\mathbf{W}^\top)\mathbf{T}_2\},$$

where $\mathbf{T}_1, \mathbf{T}_2 \in \mathcal{T}_\mathbf{W}\mathcal{M}_\mu^\nu$.



(a) STL-10 (b) GIST1M

Figure 3: The values of objective function (8) for learning a 48-bit code on (a) STL-10 and (b) GIST1M datasets.

In our work, we use the the canonical metric and apply the Cayley transformation to overcome the non-convex constraints and the expensive cost of preserving orthogonality, as mentioned in (Wen and Yin 2013). The gradient under canonical metric is given below:

$$\nabla_c\mathcal{F} = \mathbf{G} - \mathbf{W}\mathbf{G}^\top\mathbf{W}, \quad (13)$$

where $\mathbf{G}$ is the gradient of our objective function $H(\mathbf{Y}, \mathbf{W})$ with respect to $\mathbf{W}$:

$$\mathbf{G} = 2(\mathbf{X}^\top\mathbf{L}\mathbf{X}\mathbf{W} + \rho\mathbf{X}^\top\mathbf{X}\mathbf{W} - \rho\mathbf{X}^\top\mathbf{Y}). \quad (14)$$

With the Cayley transformation: $\mathbf{P}(\tau) = \mathbf{QW}$, where $\mathbf{Q} = (\mathbf{I}+\frac{\tau}{2}\mathbf{M})^{-1}(\mathbf{I}-\frac{\tau}{2}\mathbf{M})$ and $\mathbf{M}$ is a skew-symmetric matrix defined as $\mathbf{M} = \mathbf{G}\mathbf{W}^\top - \mathbf{W}\mathbf{G}^\top$, we employ the new trial point $\mathbf{P}(\tau)$ to replace $\mathbf{W}$. Moreover, $\mathbf{P}(\tau)^\top\mathbf{P}(\tau) = \mathbf{W}^\top\mathbf{W}$ for all $\tau \in \mathrm{R}$ and $\{\mathbf{P}(\tau)\}_{\tau\geq 0}$ is a descent path (Wen and Yin 2013). Hence, orthogonality is preserved and a gradient descent method can be adopted to minimize $H(\mathbf{Y}, \mathbf{W})$. In practice, the Crank-Nicolson-like update scheme (another form of Cayley transformation) is used as the iteration formulation:

$$\mathbf{P}(\tau) = \mathbf{W} - \tau\mathbf{M}(\frac{\mathbf{W} + \mathbf{P}(\tau)}{2}), \quad (15)$$

where τ denotes a step size satisfying the Armijo-Wolfe conditions (Nocedal and Wright 1999). And we accelerate it by Barzilai-Borwein (BB) step size as in (Wen and Yin 2013).

We alternate between projection step and quantization step for several iterations to seek a locally optimal solution. Since in each step we minimize the objective function, we have

$$H(\mathbf{Y}^{(t)}, \mathbf{W}^{(t)}) \geq H(\mathbf{Y}^{(t+1)}, \mathbf{W}^{(t)}) \geq H(\mathbf{Y}^{(t+1)}, \mathbf{W}^{(t+1)}),$$

where $\mathbf{Y}^{(t)}$ denotes the t^{th} iteration results of $\mathbf{Y}$, the same as $\mathbf{W}^{(t)}$. The typical behavior of the values of Eq.(8) is presented in Figure 3. We do not have to iterate until convergence. In practice, we use 50 iterations for all experiments,

Algorithm 1 Locality Preserving Hashing (LPH)

Input: Data $\mathbf{X}$ (zero-centered); random initialization matrix $\mathbf{W}^{(0)}$; positive parameter ρ; number of hash bits K; iteration counter $t \leftarrow 1$.

repeat

Update binary codes $\mathbf{Y}^{(t)}$ from $\mathbf{W}^{(t-1)}$ by Eq.(12);

Update projection matrix $\mathbf{W}^{(t)}$ from $\mathbf{Y}^{(t)}$ by Eq.(15);

$t \leftarrow t + 1$;

until convergence;

Output: Hash codes $\mathbf{Y}$ and projection matrix $\mathbf{W}$.

which has already achieved good performance. The whole procedure of LPH is outlined in Algorithm 1.

In particular, if we remove quantization stage from Eq.(8) by setting $\rho = 0$, our method will reduce to the classical SH problem. On the other hand, if we set $\rho \to \infty$ in Eq.(8), our method is equivalent to only minimizing the quantization loss (7), which is denoted as LPH-q in the experiments. These two paradigms can be seen as the special cases of our model. We experimentally present that our method (with ρ equal to 1) has the best performance.

Experiments

Datasets

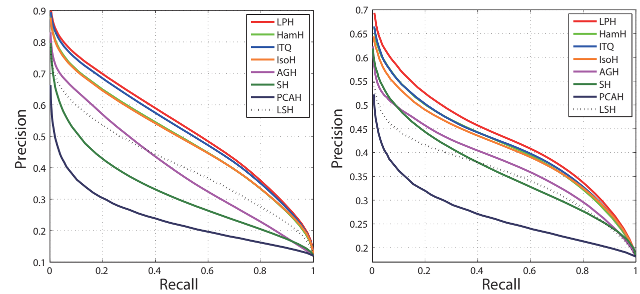
We evaluate our LPH method on the two benchmark datasets: STL-10 and ANN-GIST1M.

- STL-10 dataset is an image recognition dataset with higher resolution (96×96). It has 10 classes, 5000 training images, 8000 test images and 100000 unlabeled images. In our experiments, images are represented as 384-dimensional grayscale GIST descriptors (Oliva and Torralba 2001). The training set consists of 100000 unlabeled images, and the test set consists of 1000 test images.
- ANN-GIST1M dataset is a subset of the largest set provided for the task of ANN search. We randomly sample 100000 images from its 1 million 960-dimensional GIST features as the training set, and 1000 query images as the test set.

Evaluation Protocols and Baselines

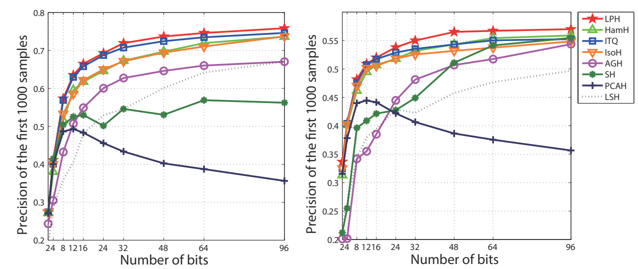
We evaluate the performance of nearest neighbor search by using Euclidean neighbors as ground truth. More specifically, we randomly pick 10000 samples from the training set to construct a pair-wise distance matrix $\mathbf{D}^*$ with l_2 norm and set the 10^{th} percentile distance in $\mathbf{D}^*$ as the threshold, which is used to judge whether a returned point is a true positive or not. And we adopt $Hamming\ ranking$ to report the averaged precision as in (Wang, Kumar, and Chang 2012; 2010a; 2010b). Then, we compute the Precision-Recall curves and retrieving accuracy.

The existing hashing methods can be divided into three categories: supervised, semi-supervised and unsupervised. Our LPH is essentially unsupervised, for fair comparison, we compare our method with the following representative unsupervised ones: LSH (Gionis et al. 1999), PCAH (Gong and Lazebnik 2011), SH (Weiss, Torralba, and Fergus 2008), AGH (Liu et al. 2011) (we use its two-layer hash functions to generate hash bits), ITQ (Gong and Lazebnik 2011), IsoH (Kong and Li 2012b) (we implement the Lift and Projection algorithm there to solve the rotating problem), HamH (Xu et al. 2013).



(a) STL-10 (b) GIST1M

Figure 6: Precision-recall curves with 48 bits on (a) STL-10 and (b) GIST1M datasets.



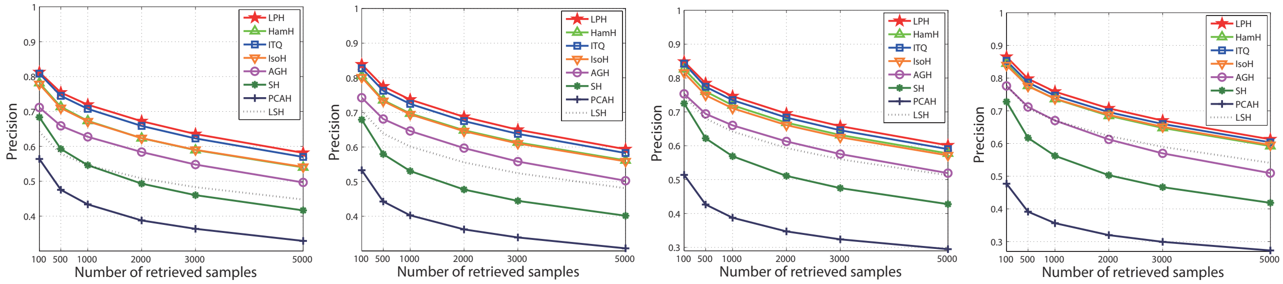
(a) STL-10 (b) GIST1M

Figure 7: Precision of first 1000 samples with different number of bits on (a) STL-10 and (b) GIST1M datasets.

Results

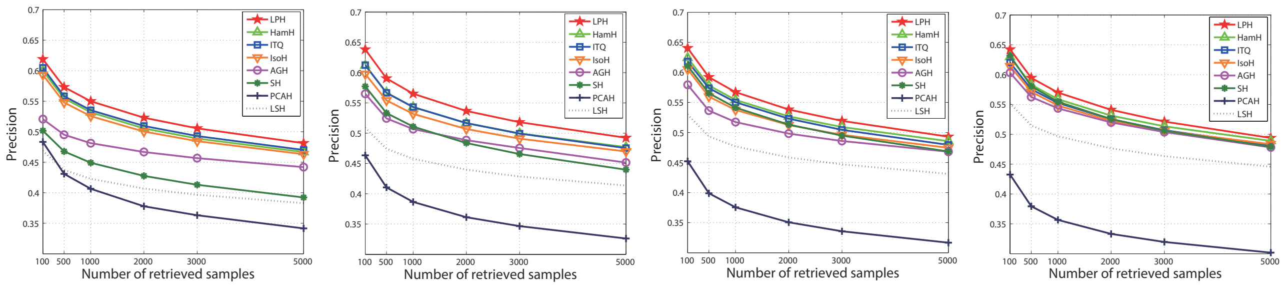
We first present the precision curves with different number of retrieved samples in Figures 4 and 5. We vary the length of hash bits from 32 to 96 to see the performance of all methods. It is clear that our LPH method outperforms all other methods on each dataset and each code size, proving its efficiency and stability. ITQ, IsoH and HamH perform better than other methods since they rotate the initial projection matrix to minimize their loss functions in quantization stage. Another case is AGH, which employs a hierarchical threshold learning procedure. Both are always better than directly thresholding. By contrast, LPH preserves the locality structure in projection stage and tries to maintain the property in quantization stage with a joint optimization technique. Consequently, the experimental results demonstrate that our locality preserving strategy is more reasonable.

In Figure 6, we plot the precision-recall curves with 48 bits, and Figure 7 illustrates the precision of first 1000 returned samples with different length of hash bits. It can be seen again, the performance of LPH is superior to other state-of-the-art methods. From these figures, We also note that PCAH and LSH have almost the worst performance,



(a) 32 bits (b) 48 bits (c) 64 bits (d) 96 bits

Figure 4: Precision curves on STL-10 dataset with different number of retrieved samples at 32, 48, 64, 96 bits respectively.



(a) 32 bits (b) 48 bits (c) 64 bits (d) 96 bits

Figure 5: Precision curves on GIST1M dataset at different number of retrieved samples with 32, 48, 64, 96 bits respectively.

Table 1: Precision of first 1000 samples with different number of bits on (a) STL-10 and (b) GIST1M datasets.

	STL-10			GIST1M		
#bits	32	48	96	32	48	96
LPH	**0.7194**	**0.7368**	**0.7586**	**0.5499**	**0.5651**	**0.5700**
LPH-q	0.6956	0.7087	0.7342	0.5377	0.5485	0.5514
SH	0.5460	0.5305	0.5623	0.4491	0.5109	0.5546

since the maximum variance projection can hardly preserve locality structure and the random projection is weak in preserving neighborhood relationships.

Table 1 shows the precision of first 1000 returned samples with different code sizes of LPH, LPH-q and SH. Obviously, our LPH method outperforms these two methods on each code size, which once again verifies that the joint learning framework of LPH is effective and efficient.

Finally, we record the training time on the two datasets in Table 2. Although LPH needs more time than the most hashing methods, it is still faster than AGH. Considering the complicated optimization procedure and the learning process running totally off-line, the time of our method is acceptable. We skip the test time comparisons, since their query processes are similar and efficient.

Conclusions

To capture meaningful neighbors, a lot of hashing algorithms have been proposed to preserve the neighbor relationships. However, they either guarantee the locality structure only in the projection stage, or do this separately between the two stages. In this paper, we have proposed a new method named Locality Preserving Hashing to hold the locality preserving property in two stages simultaneously, by combining minimizing the average projection distance and the quantization loss with a joint learning technique. Experimental results present significant performance gains over or comparable to other state-of-the-art methods in large scale similarity search on high-dimensional datasets.

Table 2: Training time (in seconds) of all methods on (a) STL-10 and (b) GIST1M datasets.

	STL-10			GIST1M		
#bits	24	32	48	24	32	48
LSH	0.61	0.79	0.88	1.49	1.64	2.08
PCAH	0.04	0.03	0.05	0.06	0.07	0.09
SH	0.74	0.84	0.94	2.99	3.08	3.17
AGH	177.65	203.08	287.84	333.32	428.56	572.60
IsoH	0.12	0.15	0.24	0.14	0.18	0.27
ITQ	3.13	4.33	6.48	3.10	4.31	6.44
HamH	1.60	1.61	1.87	3.26	3.33	3.67
LPH	123.58	173.86	229.47	297.63	350.48	449.25

Acknowledgements

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

Seventh Frame work Programme (Grant No.247619).

References

Datar, M.; Immorlica, N.; Indyk, P.; and Mirrokni, V. S. 2004. Locality-sensitive hashing scheme based on p-stable distributions. In *Proceedings of the twentieth annual symposium on Computational geometry*, 253–262. ACM.

Gionis, A.; Indyk, P.; Motwani, R.; et al. 1999. Similarity search in high dimensions via hashing. In *Proceedings of the international conference on very large data bases*, 518–529.

Gong, Y., and Lazebnik, S. 2011. Iterative quantization: A procrustean approach to learning binary codes. In *Computer Vision and Pattern Recognition*, 817–824. IEEE.

Helmke, U., and Moore, J. 1996. Optimization and dynamical systems. *Proceedings of the IEEE* 84(6):907.

Kong, W., and Li, W.-J. 2012a. Double-bit quantization for hashing. In *AAAI*.

Kong, W., and Li, W.-J. 2012b. Isotropic hashing. In *Advances in Neural Information Processing Systems*, 1655–1663.

Kreyszig, E. 1968. *Introduction to differential geometry and Riemannian geometry*, volume 16. University of Toronto Press.

Kulis, B., and Grauman, K. 2009. Kernelized locality-sensitive hashing for scalable image search. In *IEEE 12th International Conference on Computer Vision*, 2130–2137.

Kulis, B.; Jain, P.; and Grauman, K. 2009. Fast similarity search for learned metrics. *IEEE Trans. on Pattern Analysis and Machine Intelligence,* 31(12):2143–2157.

Liu, W.; Wang, J.; Kumar, S.; and Chang, S.-F. 2011. Hashing with graphs. In *Proceedings of the 28th International Conference on Machine Learning*, 1–8.

Liu, W.; Wang, J.; Ji, R.; Jiang, Y.-G.; and Chang, S.-F. 2012. Supervised hashing with kernels. In *Computer Vision and Pattern Recognition*, 2074–2081. IEEE.

Ng, M. K.; Liao, L.-Z.; and Zhang, L. 2011. On sparse linear discriminant analysis algorithm for high-dimensional data classification. *Numerical Linear Algebra with Applications* 18(2):223–235.

Niyogi, X. 2004. Locality preserving projections. In *Neural information processing systems*, volume 16, 153.

Nocedal, J., and Wright, S. J. 1999. *Numerical optimization*, volume 2. Springer New York.

Oliva, A., and Torralba, A. 2001. Modeling the shape of the scene: A holistic representation of the spatial envelope. *International journal of computer vision,* 42(3):145–175.

Salakhutdinov, R., and Hinton, G. 2009. Semantic hashing. *International Journal of Approximate Reasoning* 50(7):969–978.

Strecha, C.; Bronstein, A. M.; Bronstein, M. M.; and Fua, P. 2012. Ldahash: Improved matching with smaller descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34(1):66–78.

Wang, X.-J.; Zhang, L.; Jing, F.; and Ma, W.-Y. 2006. Annosearch: Image auto-annotation by search. In *Computer Vision and Pattern Recognition*, volume 2, 1483–1490. IEEE.

Wang, J.; Kumar, S.; and Chang, S.-F. 2010a. Semi-supervised hashing for scalable image retrieval. In *Computer Vision and Pattern Recognition*, 3424–3431. IEEE.

Wang, J.; Kumar, S.; and Chang, S.-F. 2010b. Sequential projection learning for hashing with compact codes. In *Proceedings of the 27th International Conference on Machine Learning*, 1127–1134.

Wang, J.; Kumar, S.; and Chang, S.-F. 2012. Semi-supervised hashing for large-scale search. *IEEE Trans. on Pattern Analysis and Machine Intelligence,* 34(12):2393–2406.

Weiss, Y.; Torralba, A.; and Fergus, R. 2008. Spectral hashing. In *Advances in Neural Information Processing Systems*, 1753–1760.

Wen, Z., and Yin, W. 2013. A feasible method for optimization with orthogonality constraints. *Mathematical Programming* 1–38.

Xu, B.; Bu, J.; Lin, Y.; Chen, C.; He, X.; and Cai, D. 2013. Harmonious hashing. In *Proceedings of the Twenty-Third international joint conference on Artificial Intelligence*, 1820–1826. AAAI.

Zhao, K.; Liu, D.; and Lu, H. 2013. Local linear spectral hashing. In *Neural Information Processing*, 283–290. Springer.