Unsupervised Local Feature Hashing for Image Similarity Search

Li Liu, Mengyang Yu, *Student Member, IEEE*, and Ling Shao, *Senior Member, IEEE*

***Abstract*—The potential value of hashing techniques has led to it becoming one of the most active research areas in computer vision and multimedia. However, most existing hashing methods for image search and retrieval are based on global feature representations, which are susceptible to image variations such as viewpoint changes and background cluttering. Traditional global representations gather local features directly to output a single vector without the analysis of the intrinsic geometric property of local features. In this paper, we propose a novel unsupervised hashing method called unsupervised bilinear local hashing (UBLH) for projecting local feature descriptors from a high-dimensional feature space to a lower-dimensional Hamming space via compact bilinear projections rather than a single large projection matrix. UBLH takes the matrix expression of local features as input and preserves the feature-to-feature and image-to-image structures of local features simultaneously. Experimental results on challenging data sets including Caltech-256, SUN397, and Flickr 1M demonstrate the superiority of UBLH compared with state-of-the-art hashing methods.**

***Index Terms*—Hashing, image similarity search, local feature, unsupervised learning.**

I. Introduction

LEARNING to hash has received substantial attention due to its potential in various applications such as data mining, pattern recognition, and information retrieval [1]–[7]. Compact hashing enables significant efficiency gains in both storage and retrieval speed for large-scale databases. Generally speaking, greedy-searching-based retrieval on a data set with N samples is infeasible because linear complexity $O(N)$ is not scalable to realistic applications on large-scale data. Meanwhile, most vision tasks also suffer from the curse of dimensionality, because visual descriptors usually have hundreds or even thousands of dimensions. Due to above reasons, hashing techniques are proposed to effectively embed data from a high-dimensional feature space into a similarity-preserved low-dimensional Hamming space where an approximate nearest neighbor (ANN) of a given query can be found with sublinear time complexity.

Manuscript received June 10, 2015; revised September 14, 2015; accepted September 20, 2015. Date of publication October 13, 2015; date of current version October 13, 2016. This paper was recommended by Associate Editor B. Chaib-draa. *(Corresponding author: Ling Shao.)*

The authors are with the Department of Computer Science and Digital Technologies, Northumbria University, Newcastle upon Tyne NE1 8ST, U.K. (e-mail: li2.liu@northumbria.ac.uk; m.y.yu@ieee.org; ling.shao@northumbria.ac.uk).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCYB.2015.2480966

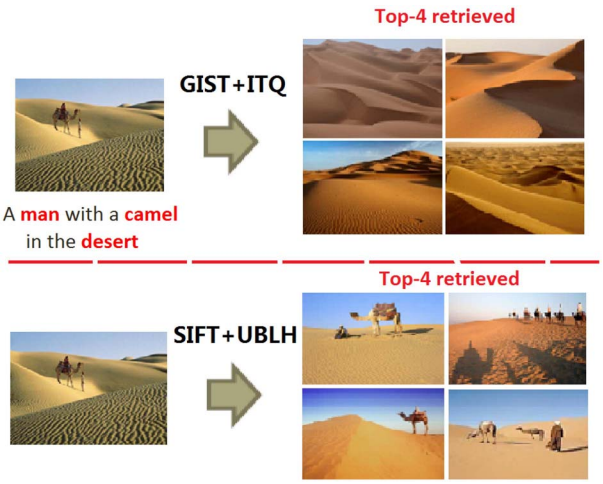


Fig. 1. Comparison of a global representation-based hashing method and our proposed method for relatively complex scene retrieval. The top figure shows the top four retrieved images using GIST features with ITQ approach which is regarded as one of the state-of-the-art hashing methods. The bottom figure shows the retrieved images via SIFT+UBLH. The result illustrates GIST+ITQ cannot return images with detailed information as in the query image. Compared with global representation based hashing, our method is more robust for complex object/scene retrieval tasks.

Currently, both conventional unsupervised and supervised hashing algorithms are primarily designed for global representations, e.g., GIST [8]. For realistic visual retrieval tasks, however, these global hashing techniques cannot cope with different complications appearing in the images such as cluttering, scaling, occlusion, and change of lighting conditions. However, these aspects are more invariant in local features-based representations such as bag-of-features [9], [10], since such representations are statistical distributions of image patches and tend to be more robust in challenging and noisy scenarios. Fig. 1 illustrates the comparison of a global representation-based hashing method (GIST+ITQ) and our proposed local feature-based hashing method [SIFT+unsupervised bilinear local hashing (UBLH)] for relatively complex scene retrieval. We can observe that the top four retrieved images via GIST+ITQ only contain the desert. They are indeed relevant to the query image, but not exactly what we want to search. While, the top four retrieved images via our approach, i.e., SIFT+UBLH, include all the detailed information, i.e., man, camel, and desert, as in the query image. The possible reason of the difference on the above retrieval task is that global representation-based hashing methods are

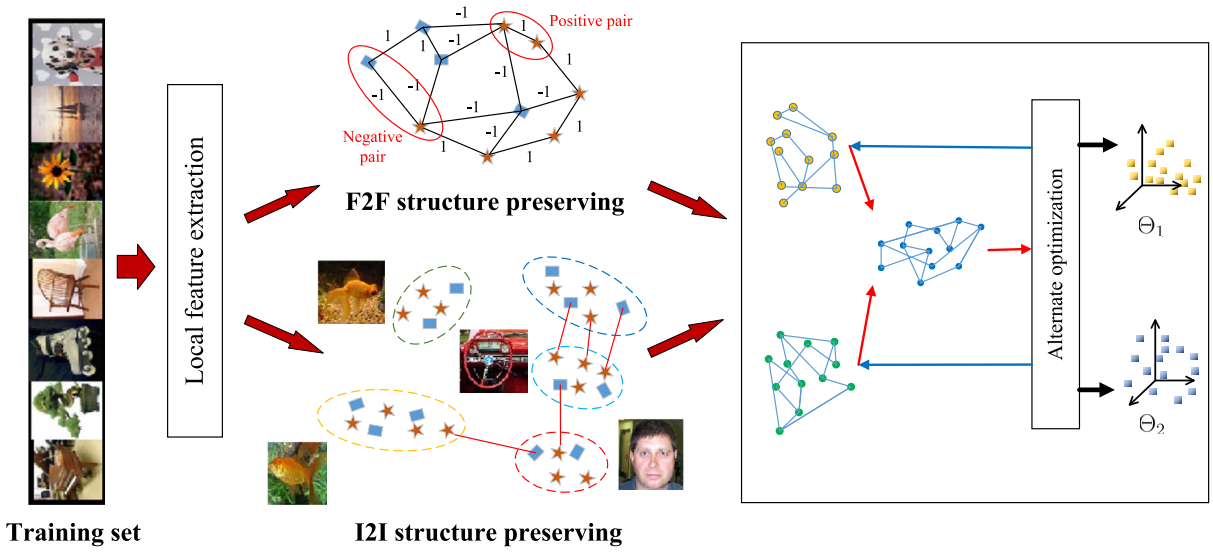


Fig. 2. Working flow of UBLH learning. The algorithm intends to preserve the pairwise structure and the I2I distances and outputs the optimal bilinear orthogonal projection matrices Θ_1 and Θ_2.

good at extracting the global intensity, color, texture, and gradient information of images, but will ignore the detailed information in the query image without analyzing the intrinsic geometric property of local features. This problem may heavily limit the effectiveness on applications that demand more accurate retrieval results for complex scene/object images. Thus, inspired by advantages (e.g., invariance for cluttering, scaling, and occlusion) of local representations, in this paper, we intend to develop a local feature-based hashing method for improving the retrieval results. If keypoints are well detected, local hash codes are able to avoid the limitations such as background variations, occlusions, and shifts in global representations.

In this paper, we propose an UBLH framework for large-scale visual similarity search, in which the feature-to-feature (F2F) and image-to-image (I2I) structures are successfully combined and preserved together. Specifically, the F2F structure considers the pairwise relationship between local features in the original feature space, which is always considered in manifold learning techniques [11]–[16]. From a higher-level aspect, I2I structure reflects the connection between images when each of them is represented by a set of local features. In particular, the I2I distance can provide a feasible way to measure the connection between two images, which is derived from [17]. It measures the distance between two images using the set representation of local features of the image. Since the raised problem of UBLH is regarded as nonconvex and discrete, our objective function is then optimized via an alternate way with relaxation.

Furthermore, motivated by [18]–[20], a bilinear projection is employed to make the algorithm more efficient. To be specific, the bilinear projection applies two projection matrices to local features, which have much smaller sizes than the original single projection matrix. Since the computational complexity of eigen decomposition is cubic degree on the dimension of the matrix, the effect of applying smaller matrices is quite conspicuous. Beyond that, most local features are based on histograms and bins, e.g., an SIFT feature is computed from 16 histograms, each of which has eight bins. Therefore, the bilinear scheme can explore two different kinds of data structure from the views of histograms and bins simultaneously. More crucially, when local features are transformed from the vector form to the matrix form, a factorization of integral for dimensionality is needed. These two different views provide a natural factorization.

The outline of our proposed UBLH is demonstrated in Fig. 2. Considering that our method is specifically designed for local feature-based hashing, the original Hamming ranking and Hamming table cannot be directly applied on local features for visual indexing. Thus, in this paper, we also introduce an image indexing/searching scheme called local hashing voting (LHV), which has been demonstrated to be efficient and accurate for image similarity search in our experiments.

This paper aims at unsupervised linear (bilinear) hashing for local features, which makes UBLH effective and practical for real-world applications without class label information. With the bilinear projection learning, the complexity of the eigen decomposition, which is the cubic form of the dimensionality, will be significantly reduced. Once the projections are learned, they can be efficiently used on the test data. Additionally, UBLH simultaneously preserves the F2F and I2I structures which can be regarded as the local and global structures respectively in the original feature space.

II. Related Work

In terms of bilinear hashing, Gong *et al.* [18] applied the bilinear scheme to the global representations by minimizing the angles between the rotated features and the corresponding binary codes. Although this scheme can effectively solve the high-dimensional hashing problem with less computational complexity, it still has some deficiency in the optimization process. Specifically, the angle between the vector in a continuous

space and that in a discrete space would bring quantization errors in the optimization. Moreover, their scheme lacks considering the relationship among features.

To explore hashing in the early time, random projections are always used to construct randomized hash functions. A most well-known representative is locality-sensitive hashing (LSH) [21], [22], which can preserve similarity information and map data points close in a Euclidean space to similar codes. It is theoretically guaranteed that as the code length increases, the Hamming distance between two codes will asymptotically approach the Euclidean distance between their corresponding data points. Furthermore, a kernel trick, which allows the use of a wide class of similarity functions, was combined with LSH to generalize LSH with arbitrary kernel functions [23]. Beyond that, principled linear projections such as PCA hashing (PCAH) [24] and its rotational variant have been introduced for better quantization rather than random projection hashing. Spectral hashing (SpH) [25] was proposed to preserve the data locality relationship to keep neighbors in the input space as neighbors in the Hamming space. Anchor graphs hashing (AGH) [26] is adopted to obtain tractable low-rank adjacency matrices for efficient similarity search. Kernel reconstructive hashing [27] was proposed to preserve the similarity defined by an arbitrary kernel using compact binary code. Compressed hashing (CH) [28] has been effectively applied for large-scale data retrieval tasks as well. All these hashing techniques mentioned above are regarded as unsupervised methods which may lead to worse retrieval precision for the data sets with noise. To achieve better results, researchers have developed supervised hashing methods which could attain higher search accuracy, since the label information is involved in the learning phase. A simple supervised hashing method is linear discriminant analysis hashing (LDAH) [29] which can tackle supervision via easy optimization but still lacks adequate performance due to the use of orthogonal projection in hash functions. Beyond that, some more complicated methods have been proposed such as binary reconstructive embeddings (BRE) [30], minimal loss hashing (MLH) [31], and kernel supervised hashing (KSH) [32]. Although these supervised methods can achieve promising results, they impose difficult optimization with slow training mechanisms. It is noteworthy that all of methods mentioned above only can be utilized with global representations.

An early work of applying local features to image detection and retrieval was proposed in [33]. Based on LSH, Joly and Buisson [34] proposed a multiprobe LSH for ANN search to improve the local feature-based retrieval tasks [35]. Another ANN algorithm was introduced in [36] to speed up the searching algorithm and find the best algorithm configuration for various data sets. Although a hybrid hashing method for SIFT descriptors was proposed in [37], the relationships between local features are not included in the code learning phase.

The main work for embedding local features to the Hamming space was proposed in [38]. In particular, two schemes are introduced to improve the standard bag-of-words (BoW) model: 1) a Hamming embedding (HE) which provides binary signatures to refine visual words and 2) a weak geometric consistency constraint with the geometrical transformation. Both methods can significantly improve the final performance for retrieval tasks. Furthermore, a coupled multi-index framework was proposed for accurate image retrieval [39]. Beyond that, a selective match kernel approach [40] has also been developed to incorporate matching kernels sharing the best properties of HE and vector of locally aggregated descriptors (VLAD). Another related work based on [38] can be seen in [41], which introduces a color binary descriptor being calculated in either a global or a local form.

However, all the above embedding methods mainly focus on the retrieval techniques rather than the learning procedure of the binary coding for large-scale hashing. Besides, these methods are not fully linear, which limits their efficiency and applicability for large-scale data sets. In fact, one of the most related work using bilinear projection on local feature hashing can be found in [42], which is regarded as a supervised learning method for image similarity search.

III. Unsupervised Bilinear Local Hashing

In this section, we first introduce the bilinear scheme [18] to present our algorithm. Then we illustrate how the F2F and I2I structures are preserved in UBLH. An alternate optimization is used for learning the bilinear projections for hash codes.

A. Notations and Problem Statement

We are given N local features $\mathbf{x}_1, \ldots, \mathbf{x}_N \in \mathbb{R}^D$ from n images. For image i, we use $\mathcal{X}_i = \{\mathbf{x}_{i1}, \ldots, \mathbf{x}_{im_i}\}$ to represent its local feature set. Bilinear projection is to multiply projection matrices on both sides of data. It can explore the matrix structure of features to enhance the effectiveness of projection. First, we factor integer D as $D = D_1 \times D_2$. Then we reorganize vector $\mathbf{x}_i$ into matrix $X_i \in \mathbb{R}^{D_1 \times D_2}$ such that $\text{vec}(X_i) = \mathbf{x}_i$, where $\text{vec}(\cdot)$ represents the vectorization of a matrix. And we also have the inverse map of vectorization $\text{vec}^{-1}(\mathbf{x}_i) = X_i$, since the vectorization is a one-to-one correspondence if D_1 and D_2 are given. To make the transformation more efficient, in this matrix form of local features, we define our hash function using two matrices $\Theta_1 \in \mathbb{R}^{D_1 \times d_1}$ and $\Theta_2 \in \mathbb{R}^{D_2 \times d_2}$

$$H(X_i) = \text{sgn}\big(\text{vec}\big(\Theta_1^T X_i \Theta_2\big)\big) \in \{-1, +1\}^{d_1 d_2}. \tag{1}$$

In fact, we notice that $\text{vec}(\Theta_1^T X_i \Theta_2) = (\Theta_2^T \otimes \Theta_1^T)\text{vec}(X_i) = (\Theta_2^T \otimes \Theta_1^T)\mathbf{x}_i$, where $\otimes$ is the Kronecker product, thus a bilinear projection is simply a special case of the single matrix projection Θ which can be decomposed as $\Theta = \Theta_2 \otimes \Theta_1$. Besides, it is easy to show that if Θ_1 and Θ_2 are orthogonal, i.e., $\Theta_1^T \Theta_1 = I_{d_1 \times d_1}$ and $\Theta_2^T \Theta_2 = I_{d_2 \times d_2}$, then Θ is orthogonal, as well. The bilinear projection leads to a more efficient eigen decomposition on matrices with much smaller sizes $D_1 \times D_1$ and $D_2 \times D_2$ rather than $D_1 D_2 \times D_1 D_2$ for single projection. Additionally, the space complexity for bilinear projections is $O(D_1^2 + D_2^2)$, while the single one needs $O((D_1 \times D_2)^2)$. Besides, since most of the local features are represented as concatenated histogram vectors, they can be intrinsically decomposed by two data structures. For instance, 128-dim SIFT is computed on 4×4 grids and for each grid

a 8-bin histogram is calculated. In this way, a 128-dim SIFT is formed by concatenating 16×8-bin histograms. Thus, for SIFT feature, we can naturally decomposed it via 16×8 in our bilinear codes learning.

Note that during the learning stage, we use $\{-1, +1\}$ to represent the output of hash functions and employ centralized data $\mathbf{x}_i - (1/N)\sum_{j=1}^{N} \mathbf{x}_j$ instead of $\mathbf{x}_i$, $\forall i$. In the indexing phase, we use $\{0, 1\}$ to represent codes for hash lookup.

B. Feature-to-Feature Structure Preserving

To obtain meaningful hash codes for local features, let us first consider the geometric structure of the entire local feature set $\mathcal{F} = \{X_1, \ldots, X_N\}$. We are concerned about the individual relationship between local features in the high-dimensional space, which should also be retained in the lower-dimensional space. Specifically, for similar (dissimilar) pairs, their distance is expected to be minimized (maximized) in the Hamming space. Since the class labels are unavailable for unsupervised method, we first employ K-means clustering on $\mathcal{F}$ to obtain some weak label information. Then the pairwise label of (X_i, X_j) is defined as

$$\ell_{ij} = \begin{cases} +1, & X_i \text{ and } X_j \text{ are in the same cluster} \\ -1, & \text{otherwise.} \end{cases} \tag{2}$$

Since different pairs have different importance in the embedding, for pair (X_i, X_j), we assign a weight which is related to the pairwise distance with parameter σ

$$w_{ij}^F = \exp\left(-\frac{\ell_{ij}+1}{2\sigma^2}\left\|X_i - X_j\right\|^2 + \frac{\ell_{ij}-1}{2\sigma^2\left\|X_i - X_j\right\|^2}\right) \tag{3}$$

where $\|\cdot\|$ is Frobenius norm. We can find that $w_{ij}^F \in (0, 1)$ and for a positive pairwise label, w_{ij}^F is decreasing as the distance $\|X_i - X_j\|$ increases and vice versa. In other words, the positive pair is more important when they are close to each other, and the negative pair is more important when they are far away from each other. We denote $\mathcal{P} = \{(i, j) | X_i, X_j \in \mathcal{F}\}$. Therefore, preserving the F2F structure is to maximize

$$\sum_{(i,j)\in\mathcal{P}} w_{ij}^F \ell_{ij} \langle H(X_i), H(X_j)\rangle. \tag{4}$$

The above function reaches its maximum value when $w_{ij}^F \ell_{ij} H(X_i)$ and $H(X_j)$ are similarly sorted due to the rearrangement inequality [43].

C. Image-to-Image Structure Preserving

Now we take a higher level connection, i.e., the relationship between images, into account since source information is also crucial to local features. For image i, we still use $\mathcal{X}_i$ to represent the local feature set $\{X_{i1}, \ldots, X_{im_i}\}$ in matrix form. Derived from [17], the I2I distance from image i to image j is defined as

$$d_{ij} = \sum_{X\in\mathcal{X}_i} \left\|X - NN_j(X)\right\|^2 \tag{5}$$

where $NN_j(X)$ is the nearest neighbor of the local feature X in image j. Although the number of local features in one image is much smaller than N, the nearest neighbor search (NN-search) for all images is still time-consuming. We hope to use the cluster information in the above F2F section for the reduction of complexity. We denote the clusters of the K-means clustering on $\mathcal{F}$ by $C_1, \ldots, C_K$. Without loss of generality, supposing the local features of image j are in $C_1, \ldots, C_{K_1}$ and the order of distances from corresponding centroids to $X \in \mathcal{X}_i$ is from nearest to farthest, then the range of NN-search in $\mathcal{X}_j$ is reduced to $(C_1 \cup \cdots \cup C_{\lceil (K_1)^\delta \rceil}) \cap \mathcal{X}_j$, where $0 < \delta < 1$ and $\lceil\cdot\rceil$ is the ceiling function. This reduction of range is based on the assumption that the centroid of the cluster where the true neatest neighbor locates is also close to X. In fact, it holds when $K \to N$. After the reduction of searching range, the average complexity is reduced from $O(N^2)$ to $O(NK^{1+\delta})$ and we only need to compute the distances from X to the cluster centroids, which has been done in the K-means.

In a general situation, $d_{ij} \neq d_{ji}$. Thus, to ensure symmetry, we update the I2I distance as

$$D_{ij} = \frac{1}{2}\left(d_{ij} + d_{ji}\right). \tag{6}$$

Via a Gaussian function, we have the following I2I similarity:

$$w_{ij}^I = \exp\left(-\frac{D_{ij}^2}{2\sigma_I^2}\right), \quad i, j = 1, \ldots, n \tag{7}$$

where σ_I is the smooth parameter. After applying UBLH, we have the I2I distance in the Hamming space

$$\widehat{D}_{ij} = \frac{1}{2}\left(\sum_{X\in\mathcal{X}_i} \left\|H(X) - NN_j(H(X))\right\|^2 + \sum_{X\in\mathcal{X}_j} \left\|H(X) - NN_i(H(X))\right\|^2\right). \tag{8}$$

To preserve the I2I structure of the original space, a reasonable objective function is to minimize

$$\sum_{i=1}^{n}\sum_{j=1}^{n} \widehat{D}_{ij} \cdot w_{ij}^I. \tag{9}$$

The above function reaches the minimum value when $\{\widehat{D}_{ij}\}$ and $\{w_{ij}^I\}$ are oppositely sorted due to the rearrangement inequality [43]. With the F2F part in (4) and orthogonal constraints on Θ_1 and Θ_2, i.e., $\Theta_1^T\Theta_1 = I$ and $\Theta_2^T\Theta_2 = I$, we have the final optimization problem

$$\underset{\substack{\Theta_1^T\Theta_1 = I \\ \Theta_2^T\Theta_2 = I}}{\arg\max} \sum_{(i,j)\in\mathcal{P}} w_{ij}^F \ell_{ij} \langle H(X_i), H(X_j)\rangle - \lambda \sum_{i=1}^{n}\sum_{j=1}^{n} \widehat{D}_{ij} \cdot w_{ij}^I \tag{10}$$

where λ is the balance parameter.

D. Alternate Optimization Via Relaxation

In this section, we derive the projections of the optimization problem (10). Motivated by [25] and [32], to gain an optimal solution, we first relax the discrete sign function to a real-valued continuous function by using its signed magnitude,

i.e., $\mathrm{sgn}(x) \approx x$. In this case, the objective function of the F2F part, i.e., (4) becomes

$$\begin{aligned}\sum_{(i,j)\in\mathcal{P}} w_{ij}^F \ell_{ij} \big\langle \mathrm{vec}\big(\Theta_1^T X_i \Theta_2\big), \mathrm{vec}\big(\Theta_1^T X_j \Theta_2\big)\big\rangle \\ = \sum_{(i,j)\in\mathcal{P}} w_{ij}^F \ell_{ij} \mathrm{Tr}\Big(\big(\Theta_1^T X_i \Theta_2\big)\big(\Theta_1^T X_j \Theta_2\big)^T\Big) \\ = \sum_{(i,j)\in\mathcal{P}} w_{ij}^F \ell_{ij} \mathrm{Tr}\Big(\Theta_1^T X_i \Theta_2 \Theta_2^T X_j^T \Theta_1\Big). \end{aligned} \tag{11}$$

Besides, we also make a statistical approximation on the computation of projected I2I distances due to the large number of local features. In other words, we exchange the operation of NN-search and $H(\cdot)$ for all $X \in \mathcal{X}_i$ during the optimization, i.e., $\sum_{X\in\mathcal{X}_i} \|H(X) - NN_j(H(X))\|^2 \approx \sum_{X\in\mathcal{X}_i} \|H(X) - H(NN_j(X))\|^2$. In fact, the pairwise structure has been preserved in the F2F objective function (4), which ensures the correctness of the exchange operation. Hence, the projected d_{ij} in (5) becomes

$$\begin{aligned}\widehat{d}_{ij} &\approx \sum_{X\in\mathcal{X}_i} \big\|\Theta_1^T X \Theta_2 - \Theta_1^T NN_j(X) \Theta_2\big\|^2 \\ &= \sum_{X\in\mathcal{X}_i} \big\|\Theta_1^T \big(X - NN_j(X)\big)\Theta_2\big\|^2 \\ &= \sum_{k=1}^{m_i} \mathrm{Tr}\Big(\Theta_1^T \Delta X_{ik}^j \Theta_2 \Big(\Theta_1^T \Delta X_{ik}^j \Theta_2\Big)^T\Big) \\ &= \sum_{k=1}^{m_i} \mathrm{Tr}\Big(\Theta_1^T \Delta X_{ik}^j \Theta_2 \Theta_2^T \Big(\Delta X_{ik}^j\Big)^T \Theta_1\Big) \end{aligned} \tag{12}$$

where $\Delta X_{ik}^j \coloneqq X_{ik} - NN_j(X_{ik})$, $k = 1, \ldots, m_i$, $i, j = 1, \ldots, n$. And we also have the similar derivation for d_{ji}. Then the projected I2I distance $\widehat{D}_{ij}$ can be written as

$$\begin{aligned}\frac{1}{2}\Bigg(\sum_{k=1}^{m_i} \mathrm{Tr}\Big(\Theta_1^T \Delta X_{ik}^j \Theta_2 \Theta_2^T \Big(\Delta X_{ik}^j\Big)^T \Theta_1\Big) \\ + \sum_{k=1}^{m_j} \mathrm{Tr}\Big(\Theta_1^T \Delta X_{jk}^i \Theta_2 \Theta_2^T \Big(\Delta X_{jk}^i\Big)^T \Theta_1\Big)\Bigg). \end{aligned} \tag{13}$$

Since it is a nonconvex optimization problem, to the best of our knowledge, there is no direct way to output the projections Θ_1 and Θ_2 simultaneously. We derive an alternate iteration algorithm to update one projection when given the other, i.e., we optimize Θ_1 when Θ_2 is fixed and we fix Θ_1 to update Θ_2 iteratively. Combined with (11) and (13), let us denote the objective function by

$$\begin{aligned}&\mathcal{L}(\Theta_1, \Theta_2) \\ &\quad = \sum_{(i,j)\in\mathcal{P}} w_{ij}^F \ell_{ij} \mathrm{Tr}\Big(\Theta_1^T X_i \Theta_2 \Theta_2^T X_j^T \Theta_1\Big) \\ &\qquad - \frac{1}{2}\lambda\Bigg(\sum_{k=1}^{m_i} w_{ij}^I \mathrm{Tr}\Big(\Theta_1^T \Delta X_{ik}^j \Theta_2 \Theta_2^T \Big(\Delta X_{ik}^j\Big)^T \Theta_1\Big) \\ &\qquad\quad + \sum_{k=1}^{m_j} w_{ij}^I \mathrm{Tr}\Big(\Theta_1^T \Delta X_{jk}^i \Theta_2 \Theta_2^T \Big(\Delta X_{jk}^i\Big)^T \Theta_1\Big)\Bigg). \end{aligned} \tag{14}$$

Algorithm 1 Unsupervised Bilinear Local Hashing

Input: The local feature set $\mathcal{F}$ of training images, the number of centroids K in the K-means, the parameter δ for the NN-search in the I2I distance and the balance parameter λ.

Output: The bilinear projection matrices Θ_1 and Θ_2.

1: Preprocessing: centralize $\mathbf{x}_i \leftarrow (\mathbf{x}_i - \frac{1}{N}\sum_{j=1}^N \mathbf{x}_j)$, $i = 1, \ldots, N$;
2: Transform all the local features $\mathbf{x}_i$ into matrix form X_i, $i = 1, \ldots, N$;
3: Employ K-means clustering on $\mathcal{F}$;
4: Construct local feature pairing set $\mathcal{P}$ and their corresponding pairwise labels ℓ_{ij} according to Eq. (2);
5: Compute F2F weight w_{ij}^F and I2I similarity w_{ij}^I by Eqs. (3) and (7), respectively;
6: Initialize Θ_2 randomly;
7: **repeat**
8: $\quad\Theta_1 \leftarrow$ the eigenvectors corresponding to the largest d_1 eigenvalues of $M_2(\Theta_2)$ by Eq. (15);
9: $\quad\Theta_2 \leftarrow$ the eigenvectors corresponding to the largest d_2 eigenvalues of $M_1(\Theta_1)$ by Eq. (16);
10: **until** $\mathcal{L}(\Theta_1, \Theta_2)$ converges.
11: **return** Θ_1 and Θ_2.

By simple algebraic derivation, we have the following form:

$$\mathcal{L}(\Theta_1, \Theta_2) = \mathrm{Tr}\big(\Theta_1^T M_2(\Theta_2)\Theta_1\big) = \mathrm{Tr}\big(\Theta_2^T M_1(\Theta_1)\Theta_2\big)$$

where

$$\begin{aligned}M_2(\Theta_2) = &\sum_{(i,j)\in\mathcal{P}} w_{ij}^F \ell_{ij} X_i \Theta_2 \Theta_2^T X_j^T \\ &- \frac{1}{2}\lambda \sum_{k=1}^{m_i} w_{ij}^I \Delta X_{ik}^j \Theta_2 \Theta_2^T \Big(\Delta X_{ik}^j\Big)^T \\ &- \frac{1}{2}\lambda \sum_{k=1}^{m_j} w_{ij}^I \Delta X_{jk}^i \Theta_2 \Theta_2^T \Big(\Delta X_{jk}^i\Big)^T \end{aligned} \tag{15}$$

and

$$\begin{aligned}M_1(\Theta_1) = &\sum_{(i,j)\in\mathcal{P}} w_{ij}^F \ell_{ij} X_j^T \Theta_1 \Theta_1^T X_i \\ &- \frac{1}{2}\lambda \sum_{k=1}^{m_i} w_{ij}^I \Big(\Delta X_{ik}^j\Big)^T \Theta_1 \Theta_1^T \Delta X_{ik}^j \\ &- \frac{1}{2}\lambda \sum_{k=1}^{m_j} w_{ij}^I \Big(\Delta X_{jk}^i\Big)^T \Theta_1 \Theta_1^T \Delta X_{jk}^i \end{aligned} \tag{16}$$

are two symmetric matrix-valued functions with their codomains $\mathbb{R}^{D_1\times D_1}$ and $\mathbb{R}^{D_2\times D_2}$, respectively. Consequently, for fixed Θ_1, the optimal Θ_2 is given as the eigenvectors corresponding to the largest d_2 eigenvalues of $M_1(\Theta_1)$, and likewise, for fixed Θ_2, the optimal Θ_1 is given as the eigenvectors corresponding to the largest d_1 eigenvalues of $M_2(\Theta_2)$. Although the number of the local features is usually huge, the sizes of our final matrices M_1 and M_2 used for decomposition are small enough (D_1 and D_2 are always less than 100). This property mainly guarantees the efficiency and feasibility.

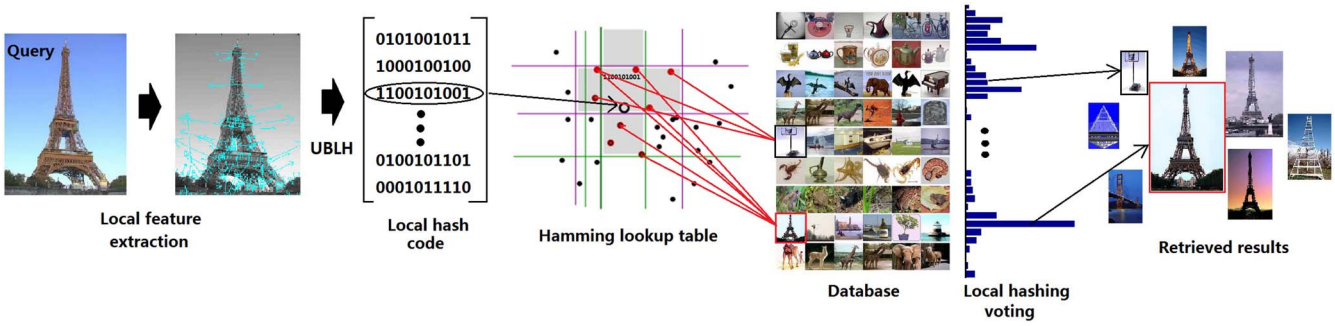


Fig. 3. Illustration for the proposed LHV. Given a query image, its local features are first extracted and embedded into hash codes via UBLH. Then, each hash code (e.g., "1100101001") corresponding to a local feature in the query image is then searched in the Hamming lookup table within the Hamming radius r and the corresponding images' indices are obtained. Finally, we vote and accumulate the times of each image's indices appearing in relevant buckets and rank them to return the retrieved results.

Therefore, for $t = 0$, we randomly initialize $\Theta_2^{(t)}$; for the tth step, we have the update rules

$$\Theta_1^{(t)} \leftarrow \text{the first } d_1 \text{ eigenvectors of } M_2\left(\Theta_2^{(t-1)}\right)$$
$$\Theta_2^{(t)} \leftarrow \text{the first } d_2 \text{ eigenvectors of } M_1\left(\Theta_1^{(t)}\right).$$

For the tth step ($t \geq 1$), we have the following inequality:

$$\mathcal{L}\left(\Theta_1^{(t-1)}, \Theta_2^{(t-1)}\right) \leq \mathcal{L}\left(\Theta_1^{(t)}, \Theta_2^{(t-1)}\right) \leq \mathcal{L}\left(\Theta_1^{(t)}, \Theta_2^{(t)}\right).$$

Thus $\mathcal{L}(\Theta_1^{(t)}, \Theta_2^{(t)})$ is monotonically nondecreasing as $t \to \infty$. And continuous function $\mathcal{L}(\Theta_1, \Theta_2)$ is bounded in the closed district $\{(\Theta_1, \Theta_2)|\Theta_1^T\Theta_1 = I, \Theta_2^T\Theta_2 = I\}$. Then the above alternate iteration converges. In practice, we stop the iteration when the difference $|\mathcal{L}(\Theta_1^{(t)}, \Theta_2^{(t)}) - \mathcal{L}(\Theta_1^{(t-1)}, \Theta_2^{(t-1)})|$ is less than a small threshold or the number of iteration reaches a maximum. We summarize UBLH in Algorithm 1.

IV. Indexing Via Local Hashing Voting

Once the bilinear projection matrices $\{\Theta_1, \Theta_2\}$ are obtained, we can easily embed the training data into binary hash codes by (1). And for a query local feature $\widehat{\mathbf{x}}$, its hash code is obtained by

$$H(\widehat{X}) = \text{sgn}\left(\text{vec}\left(\Theta_1^T\left(\widehat{X} - \frac{1}{N}\sum_{j=1} X_j\right)\Theta_2\right)\right) \quad (17)$$

with the input of centralized data, where $\widehat{X}$ is the matrix form of $\widehat{\mathbf{x}}$. For an upcoming query, a common way to find the similar samples in the training set by using Hamming distance ranking. However, for our local feature hashing scenario, traditional linear search (e.g., Hamming distance ranking) with complexity $O(N)$ is not fast any more, since N denotes the total number (at least 3M for a large-scale database)[1] of local features. To accomplish the local feature-based visual retrieval, in this paper, we introduce a fast indexing scheme via LHV as shown in Fig. 3. We first build the Hamming lookup table (also known as the hashing table) into our LHV scheme. Given a query, we can find the bucket of corresponding hash codes in near constant time $O(1)$, and return all the data in the bucket as the retrieval results.

After construction of the Hamming lookup table over the training set, we store the corresponding indices for the hash codes of all local features. For instance, given a bucket with hash code "1100101001," we store the indices of the images, which contain the same local feature hash code with this bucket. In this way, we search the hash code $H(\text{vec}^{-1}(\mathbf{q}_i))$ for each local feature $\mathbf{q}_k \in \mathcal{Q}$ in the query image $\mathcal{Q} = \{\mathbf{q}_1, \ldots, \mathbf{q}_m\}$ over the Hamming lookup table within Hamming radius r and return the possible images' indices. It is noteworthy that the same bucket in the Hamming lookup table may store the indices from different images. Finally, we vote and accumulate the times of each image's indices appearing in relevant buckets and then rank them in decreasing order. The final retrieved samples are returned according to the relevant ranking generated by LHV, which is depicted in Algorithm 2.

Algorithm 2 Local Hashing Voting

Input: The local feature set $\mathcal{F}$ of training images, the local feature set of query image $\mathcal{Q} = \{\mathbf{q}_1, \ldots, \mathbf{q}_m\}$, where $\mathbf{q}_i \in \mathbb{R}^D$, $\forall i$, Hamming radius r and the learned bilinear projection matrices $\{\Theta_1, \Theta_2\}$.

Output: The retrieved images ranked by similarity.

1: Transform all the local features in $\mathcal{F}$ and $\mathcal{Q}$ into matrix form;
2: Embed all the local features into Hamming space via Eq. (1) with $\{\Theta_1, \Theta_2\}$;
3: Construct Hamming lookup table over the training set;
4: **for** $i = 1$ to m **do**
5: For the query hash code $H(\text{vec}^{-1}(\mathbf{q}_i))$, store all the possible image indices falling into the Hamming lookup table within Hamming radius r;
6: **end for**
7: Vote and accumulate the times of each image's indices appearing and rank them in decreasing order;
8: **return** All the relevant images as the retrieved results.

[1]For large-scale database retrieval, the total number of local features is always very huge. In practice, for a 10K images training set, if each image contains 300 local feature, N would be $300 \times 10\text{K} = 3\text{M}$.

V. Complexity Analysis

The time complexity of UBLH mainly contains three parts. The first part is computing the F2F weight w_{ij}^F, which costs $O(|\mathcal{P}|D+NKTD)$ time, where T is the number of iterations in the K-means. The second part is constructing the I2I similarity w_{ij}^I. Using the reduction strategy in NN-search, the average time complexity of this part is $O(NK^{1+\delta}D)$. The last part is the eigen decomposition for the bilinear projection matrices via alternate optimization. The updates of Θ_1 and Θ_2 have the time complexity of $O(D_1^3)$ and $O(D_2^3)$, respectively. The test phase is image indexing via the LHV. For the hash code of each local feature, the index complexity is $O(1)$ using the Hamming lookup table. Thus, it only needs $O(m)$ time for a query with m local features in the search phase. In total, if we select local feature pairs less densely such that $|\mathcal{P}| \sim O(N)$ and set $\delta = 0.5$ for the reduction of complexity, the time complexity of UBLH is at most $O(ND+NKTD+NK^{1.5}D)+(O(D_1^3)+O(D_2^3))N_T+O(m)$, where N_T is the number of the iteration for alternate optimization. In the experiments, N_T is always less than 10.

VI. Experiments

In this section, the proposed UBLH algorithm is evaluated for the image similarity search problem. Three realistic image data sets are used in our experiments: Caltech-256 [44], SUN397 [45], and Flickr 1M.[2] The Caltech-256 data set consists of 30 607 images associated with 256 object categories. We further randomly choose 1000 images as the query set and the rest of data set is regarded as the training set. The SUN397 data set contains 108 754 scene images in total from 397 well-sampled categories with at least 100 images per category. Seventy samples are randomly selected from each category to construct the training set and the rest of samples are the query set. Thus, there are total numbers of 27 790 and 80 964 in the training set and query set, respectively. For the Flickr 1M data set, it contains one million Web images collected from the Flickr. We take 1K images as the queries by random selection and use the remaining to form the gallery database. Considering the huge cost of computation, in this experiment, only 150 000 randomly selected samples from the gallery database form the training set. Furthermore, for image searching tasks, given an image, we would like to describe it with a set of local features extracted from it. In our experiments, we extract 128-D SIFT as the local feature to describe the images and then learn to hash these local descriptors with all compared methods. Particularly, considering the computational cost, we limit the maximum number of local features extracted from one image with 700.

In the querying phase, using LHV as the retrieval tool, a returned point is regarded as a neighbor if it lies in the top ranked 200, 200, and 1000 points for Caltech-256, SUN397, and Flickr 1M, respectively. Specifically in LHV, we only consider the local hash codes lying in the buckets that fall within a small Hamming radius $r = 2$ (following [25]) in the Hamming lookup table which is constructed using the training codes. We evaluate the retrieval results in terms of the mean average precision (MAP) and the precision-recall curve by changing the number of top ranked points in LHV. Additionally, we also report the training time and the test time (the average searching time used for each query) for all methods. Our experiments are completed using MATLAB 2013a on a server configured with a 12-core processor and 128 GB of RAM running the Linux OS.

[2]http://www.multimedia-computing.de/wiki/Flickr1M

A. Compared Methods and Settings

Since our UBLH is the work of unsupervised linear hashing for local features, for fair comparison, the other hashing techniques originally proposed for global features are also directly applied to local features. In our experiments, we first compare the proposed method against twelve prevailing hashing algorithms, including four supervised methods: 1) LDAH [29]; 2) BRE [30]; 3) MLH [31]; and 4) KSH [32], and eight unsupervised methods: 1) LSH [21]; 2) WTA [46]; 3) PCAH [24]; 4) SpH [25]; 5) AGH [26]; 6) CH [28]; 7) ITQ [47]; and 8) spherical hashing (SpherH) [48]. Besides, HE [38], which is a nonlinear local feature-based hashing method, is also included in our comparison. We use the publicly available codes of BRE, MLH, LDAH, SpH, AGH, ITQ, and SpherH, and implement LSH, PCAH, KSH, CH, HE, and WTA ourselves. All of the above methods are then evaluated on six different lengths of codes (16, 32, 48, 64, 80, and 96). Under the same experimental setting, all the parameters used in the compared methods have been strictly chosen according to their original papers.

For our UBLH, to obtain the weak label information, the parameter K of the K-means in the proposed method for each data set is selected from one of $\{300, 400, \ldots, 1200\}$ with the step of 100 by 10-fold cross-validation on the training sets. The parameter δ is always fixed to 0.5. Besides, we set $D_1 = 16$ and $D_2 = 8$ for the transformation of 128-D SIFT local features (see Section III). Additionally, the optimal balance parameter λ is chosen from one of $\{0.05, 0.1, \ldots, 0.5\}$ with the step of 0.05 via the training sets, as well.

B. Results and Analysis

We demonstrate MAP curves on the Caltech-256, SUN397, and Flickr 1M data sets compared with different methods in Fig. 4. All the results are calculated via the proposed LHV ranking algorithm under the same setting. From the general tendency, accuracies on the SUN397 data set are lower than those on the other two data sets, since more categories and large intraclass variations exist in SUN397. Our UBLH algorithm consistently outperforms all the compared methods in every length of code. Especially on the Caltech256 data set, the improvement is near 5% between UBLH and the top supervised method KSH on each code length. Beyond that, we can observe that due to the available label information in the learning phases, the supervised methods, such as KSH, BRE, and MLH, always achieve better performance than the compared unsupervised methods on all three data sets. Interestingly, the results of LDAH always climb up then go down when the length of code increases. The same tendency also appears with

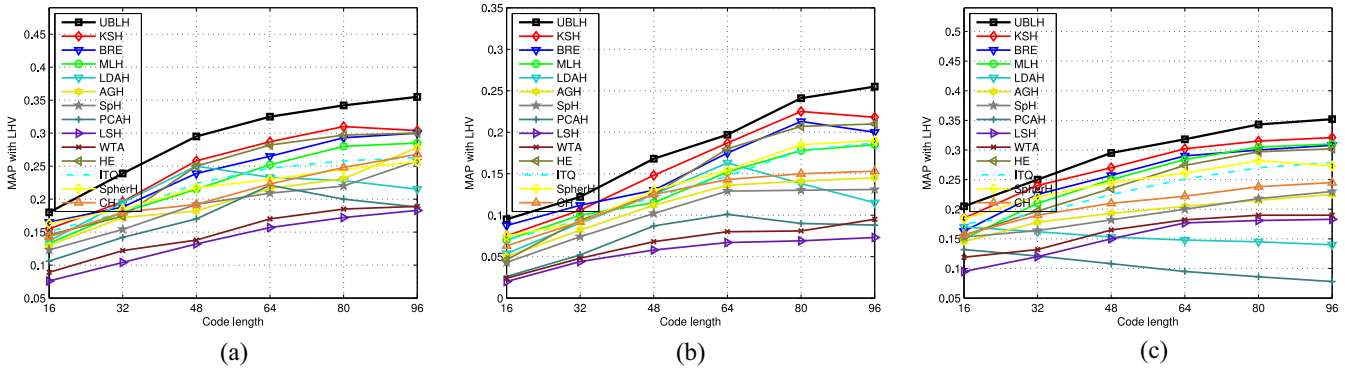


Fig. 4. Performance comparison (MAP) of UBLH and other hashing schemes with different code lengths. (a) Caltech-256. (b) SUN397. (c) Flickr 1M.

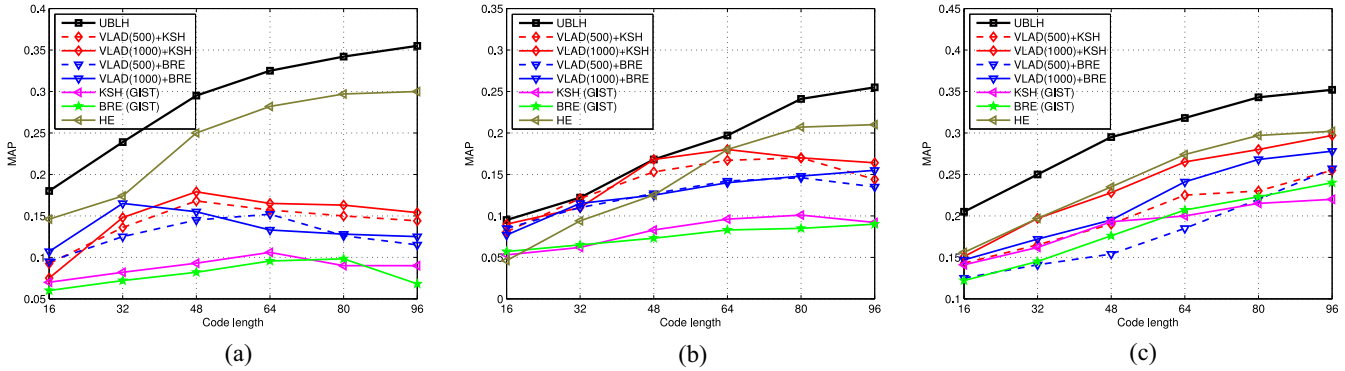


Fig. 5. Performance comparison (MAP) of UBLH with the global representation-based hashing methods and HE. (a) Caltech-256. (b) SUN397. (c) Flickr 1M.

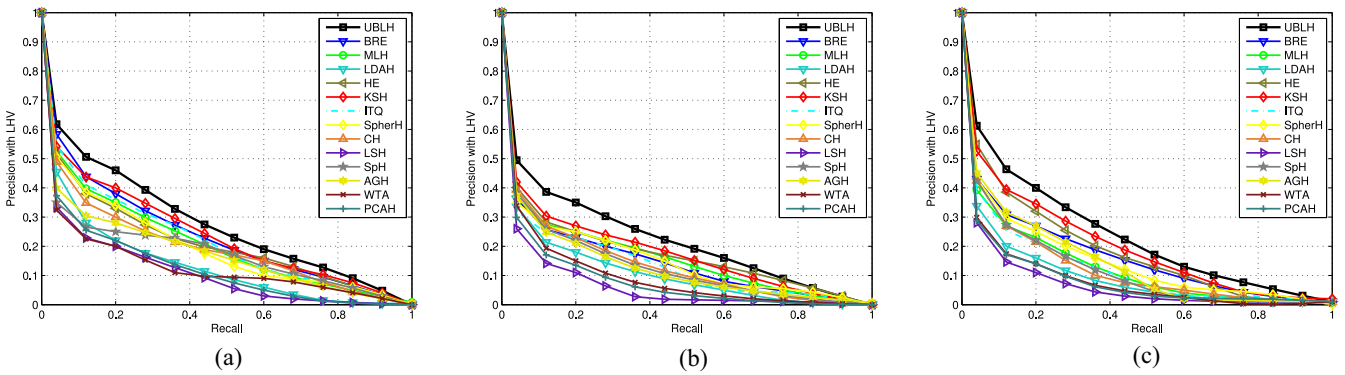


Fig. 6. Precision-recall curves of all compared algorithms on the three data sets with the code length of 96 bits. (a) Caltech-256. (b) SUN397. (c) Flickr 1M.

BRE, KSH, and PCAH. The two unsupervised hashing methods, ITQ and SpherH, generally outperform other compared unsupervised methods, while achieve worse accuracies than the supervised methods. Our UBLH achieves dramatically better performance than all other unsupervised methods and also reaches higher accuracies compared with the supervised ones over three data sets. This is because we consider the geometry structure of local features and the global relationship between images simultaneously. Besides, in Table II we also illustrate the results computed via LFBC under different local feature decomposition. Since SIFT feature is intrinsically composed via 16 grid with 8-bin histograms, the best naturally bilinear decomposition, i.e., $8 \times 16 = 128$ dim, can achieve the better results than other decomposition ways.

To make the comparison more convincing, some hashing schemes based on global representations are also included in our comparison. For all three data sets, we first use the K-means scheme to construct the codebooks with size of 500 and 1000, respectively, and then encode SIFT features into global representations via VLAD [49], which are proved to be more discriminative than original BoW representations. After that, two best performed hashing methods, i.e., KSH and BRE, are used to learn the hash codes on these global representations. Additionally, we also list the search performance via directly using the global feature GIST with KSH and BRE. In Fig. 5, it shows that our local hashing method UBLH with LHV achieves better results than the global representation-based hashing schemes with the ordinary hash table ($r = 2$)

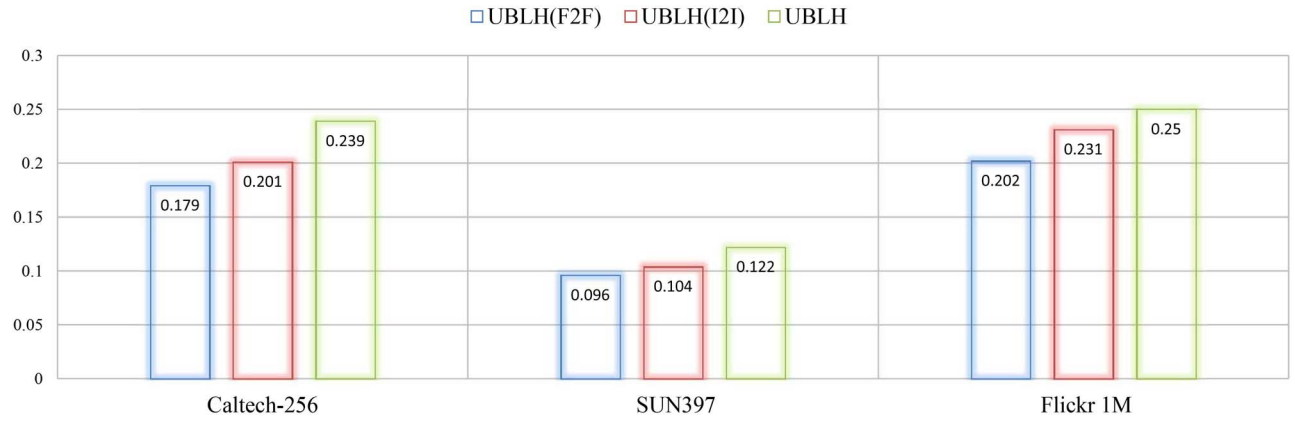
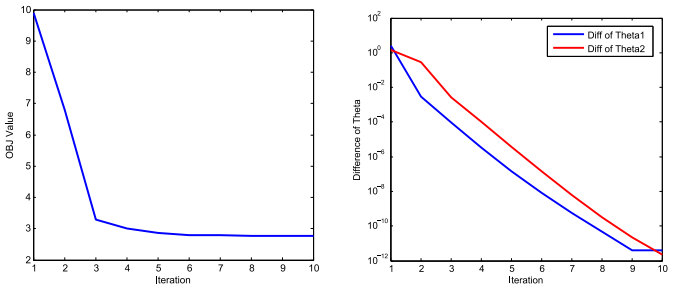


Fig. 7. Performance comparison (MAP) on the three data sets at 32-bit. UBLH(F2F) and UBLH(I2I) represent the proposed method only using the F2F term or the I2I term, respectively.

TABLE I
MAP AT 32-BIT, TRAINING TIME AND TEST TIME (WITH LHV) OF DIFFERENT HASHING METHODS ON THREE DATA SETS

Methods	Caltech-256			SUN397			Flickr 1M		
	MAP (200-NN)	Train time	Test time	MAP (200-NN)	Train time	Test time	MAP (1000-NN)	Train time	Test time
KSH*	0.196	4741.1s	57.4μs	0.106	8384.2s	63.8μs	0.240	1.14×10^4s	73.6μs
BRE*	0.188	2.84×10^4s	32.9μs	0.112	5.17×10^4s	48.8μs	0.225	6.23×10^4s	72.4μs
MLH*	0.180	1.19×10^4s	18.3μs	0.100	3.48×10^4s	25.0μs	0.210	4.77×10^4s	29.9μs
LDAH*	0.195	15.3s	11.2μs	0.091	41.4s	24.8μs	0.162	254.9s	25.0μs
AGH	0.172	764.2s	97.4μs	0.082	1832.8s	105.1μs	0.178	3688.4s	111.7μs
SpH	0.154	282.7s	103.4μs	0.074	883.0s	101.8μs	0.164	1213.3s	114.0μs
PCAH	0.142	17.4s	16.1μs	0.052	42.4s	18.4μs	0.122	277.5s	18.6μs
LSH	0.104	9.1s	8.3μs	0.044	23.6s	11.5μs	0.118	184.2s	13.1μs
WTA	0.122	1004.2s	57.8μs	0.048	3319.0s	42.5μs	0.132	4357.0s	52.8μs
ITQ	0.180	811.6s	93.0μs	0.094	2144.1s	83.2μs	0.192	3521.8s	95.3μs
SpherH	0.185	957.3s	105.6μs	0.091	2356.7s	97.8μs	0.221	3759.4s	101.2μs
CH	0.180	1712.5s	64.3μs	0.092	3721.4s	63.5μs	0.190	6367.9s	79.4μs
HE	0.174	1565.0s	87.2μs	0.094	3920.0	55.5μs	0.197	4771.5s	87.0μs
UBLH	**0.239**	1148.1s	56.1μs	**0.122**	3252.0s	50.7μs	**0.250**	5488.7s	58.4μs

(The test time indicates the average searching time used for each query. "*" denotes the supervised hashing methods.)



(a) (b)

Fig. 8. Illustration of convergence on Caltech-256 with the code length 32. (a) Objective function value versus number of iteration. (b) Differences of Θ_1 and Θ_2 versus number of iteration.

TABLE II
RESULT COMPARISON (32 BITS) VIA LFBC FROM DIFFERENT DECOMPOSITIONS OF SIFT FEATURES

Decomposition methods	Caltech-256	SUN397	Flickr 1M
1×128	0.225	0.115	0.235
2×64	0.229	0.117	0.239
4×32	0.231	0.120	0.241
8×16	**0.239**	**0.122**	**0.250**

for retrieval. Moreover, precision-recall curves of all the compared methods on these data sets with the code length of 96 bits are presented in Fig. 6 as well. From all these figures, we can further discover that, for all three data sets, UBLH achieves significantly better performance than other unsupervised methods and still slightly outperforms supervised ones by comparing the MAP and area under the curve. Fig. 8 illustrates the convergence of the proposed ULBH on Caltech-256 with the code length of 32. We can clearly observe that the objective function value is stable when the number of iteration is larger than 3. Besides, it is easy to show that with the increase in the number of iteration, the differences of Θ_1 and Θ_2, i.e., $\|\Theta_1^{(t)} - \Theta_1^{(t-1)}\|$ and $\|\Theta_2^{(t)} - \Theta_2^{(t-1)}\|$, where t is the number of iteration, dramatically drop down. Additionally, in Fig. 9, we also compare the performance of UBLH with respect to the parameter K in the K-means and balance parameter λ via cross-validation on the training sets.

In addition, to illustrate the effectiveness of F2F and I2I terms in our method, we compare the algorithm only using the F2F term or the I2I term in Fig. 7. The results indicate that preserving the I2I similarity is more effective than preserving the similarity between features. Meanwhile, combining them together could always gain better performance.

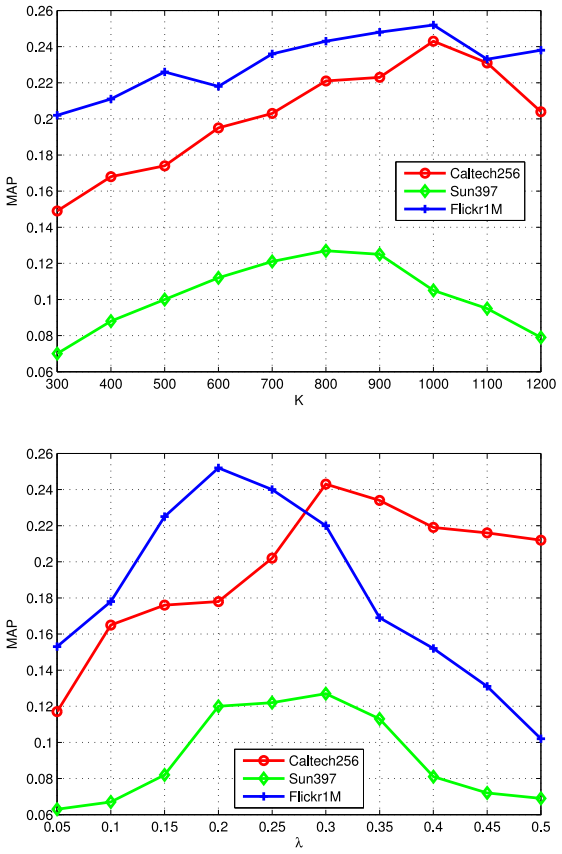


Fig. 9. Parameter sensitivity analysis (K and λ) with 32 bits on training sets. The dashed lines show the best performance for each $K \in \{300, 400, \ldots, 1200\}$ and $\lambda \in \{0.05, 0.1, \ldots, 0.5\}$, respectively, when other parameters are varied.

Finally, the training time and test time for different algorithms on three data sets are also illustrated in Table I. Considering the training time, supervised methods always need more time for the hash learning except for LDAH. In particular, BRE and MLH spend the most time to train hash functions. The random projection-based algorithms are relatively more efficient, especially the LSH. Our UBLH costs significantly less time than KSH, BRE, MLH, and CH but more than other methods for training. In terms of the test phase, LSH, LDAH, and PCAH are the most efficient methods due to the simple matrix multiplication or thresholding in binary code learning, while AGH has the comparable searching time with SpH. Our UBLH costs similar time as WTA. More details can be seen in Table I.

VII. Scalability for Very Large Data Sets

For more realistic retrieval on very large-scale data sets (e.g., the Google database), the proposed approach would become time-consuming for training due to a huge number of local features generated and involved in our computation. To avoid the heavy burden of computation and make our method practical in such cases, we can use anchor point quantization (APQ) to reduce the computational complexity of our UBLH. Inspired by [50], we can first extract the anchor points from all local features via clustering techniques (e.g., K-means) as we have stated in our algorithm. Then, each local feature in the training set can be quantized to one anchor point. In this way, we replace all local features with their corresponding anchor points in the training phase. In particular, F2F preserving can be effectively transferred to anchor point to anchor point preserving. Similarly, we can also use anchor points for I2I preserving. Thus, APQ can be applied on very large-scale data collections to enable more efficient training than directly using a huge number of original local features.

VIII. Conclusion

In this paper, we have presented a novel unsupervised hashing framework, namely UBLH, to learn highly discriminative binary codes on local feature descriptors for large-scale image similarity search. The bilinear property of UBLH lets it explore the matrix representation of local features. Considering the pairwise and source information of local features, as a result, the F2F and I2I structures have been simultaneously preserved in UBLH. We have systematically evaluated our method on the Caltech-256, SUN397, and Flickr 1M data sets and shown promising results compared to state-of-the-art hashing methods.

References

[1] Z. Jin, C. Li, Y. Lin, and D. Cai, "Density sensitive hashing," *IEEE Trans. Cybern.*, vol. 44, no. 8, pp. 1362–1371, Aug. 2014.

[2] Z. Jin *et al.*, "Fast and accurate hashing via iterative nearest neighbors expansion," *IEEE Trans. Cybern.*, vol. 44, no. 11, pp. 2167–2177, Nov. 2014.

[3] L. Chen, D. Xu, I. W.-H. Tsang, and X. Li, "Spectral embedded hashing for scalable image retrieval," *IEEE Trans. Cybern.*, vol. 44, no. 7, pp. 1180–1190, Jul. 2014.

[4] C. Leng, J. Wu, J. Cheng, X. Bai, and H. Lu, "Online sketching hashing," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Boston, MA, USA, 2015, pp. 2503–2511.

[5] L. Liu, M. Yu, and L. Shao, "Multiview alignment hashing for efficient image search," *IEEE Trans. Image Process.*, vol. 24, no. 3, pp. 956–966, Mar. 2015.

[6] X. Bai, H. Yang, J. Zhou, P. Ren, and J. Cheng, "Data-dependent hashing based on p-stable distribution," *IEEE Trans. Image Process.*, vol. 23, no. 12, pp. 5033–5046, Dec. 2014.

[7] Z. Cai, L. Liu, M. Yu, and L. Shao, "Latent structure preserving hashing," in *Proc. Brit. Mach. Vis. Conf.*, Swansea, U.K., Sep. 2015, pp. 172.1–172.11.

[8] A. Oliva and A. Torralba, "Modeling the shape of the scene: A holistic representation of the spatial envelope," *Int. J. Comput. Vis.*, vol. 42, no. 3, pp. 145–175, 2001.

[9] K. Grauman and T. Darrell, "The pyramid match kernel: Discriminative classification with sets of image features," in *Proc. 10th IEEE Int. Conf. Comput. Vis.*, vol. 2. Beijing, China, Oct. 2005, pp. 1458–1465.

[10] J. Qin, L. Liu, M. Yu, Y. Wang, and L. Shao, "Fast action retrieval from videos via feature disaggregation," in *Proc. Brit. Mach. Vis. Conf.*, Swansea, U.K., Sep. 2015, pp. 180.1–180.11.

[11] X. He and P. Niyogi, "Locality preserving projections," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 16. Vancouver, BC, Canada, 2003.

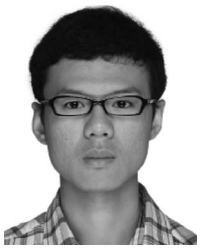
[12] D. Tao, X. Li, X. Wu, and S. J. Maybank, "Geometric mean for subspace selection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 2, pp. 260–274, Feb. 2009.

[13] S. Si, D. Tao, and B. Geng, "Bregman divergence-based regularization for transfer subspace learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 7, pp. 929–942, Jul. 2010.

[14] L. Liu and L. Shao, "Discriminative partition sparsity analysis," in *Proc. Int. Conf. Pattern Recognit.*, Stockholm, Sweden, Aug. 2014, pp. 1597–1602.

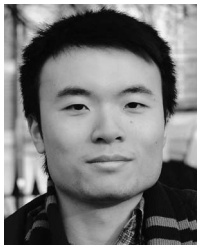
[15] B. Geng, D. Tao, C. Xu, Y. Yang, and X.-S. Hua, "Ensemble manifold regularization," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 6, pp. 1227–1233, Jun. 2012.

[16] D. Tao, X. Tang, X. Li, and X. Wu, "Asymmetric bagging and random subspace for support vector machines-based relevance feedback in image retrieval," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 7, pp. 1088–1099, Jul. 2006.
[17] O. Boiman, E. Shechtman, and M. Irani, "In defense of nearest-neighbor based image classification," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Anchorage, AK, USA, 2008, pp. 1–8.
[18] Y. Gong, S. Kumar, H. A. Rowley, and S. Lazebnik, "Learning binary codes for high-dimensional data using bilinear projections," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Portland, OR, USA, 2013, pp. 484–491.
[19] J. Ye, R. Janardan, and Q. Li, "Two-dimensional linear discriminant analysis," in *Proc. Adv. Neural Inf. Process. Syst.*, 2004, pp. 1569–1576.
[20] D. Tao, X. Li, X. Wu, and S. J. Maybank, "General tensor discriminant analysis and Gabor features for gait recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 10, pp. 1700–1715, Oct. 2007.
[21] A. Gionis, P. Indyk, and R. Motwani, "Similarity search in high dimensions via hashing," in *Proc. 25th Int. Conf. Very Large Data Bases (VLDB)*, vol. 99. Edinburgh, U.K., 1999, pp. 518–529.
[22] A. Andoni and P. Indyk, "Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions," in *Proc. 47th Annu. IEEE Symp. Found. Comput. Sci. (FOCS)*, Berkeley, CA, USA, 2006, pp. 459–468.
[23] B. Kulis and K. Grauman, "Kernelized locality-sensitive hashing for scalable image search," in *Proc. IEEE 12th Int. Conf. Comput. Vis. (ICCV)*, Kyoto, Japan, 2009, pp. 2130–2137.
[24] J. Wang, S. Kumar, and S.-F. Chang, "Semi-supervised hashing for large scale search," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 12, pp. 2393–2406, Dec. 2012.
[25] Y. Weiss, A. Torralba, and R. Fergus, "Spectral hashing," in *Proc. Adv. Neural Inf. Process. Syst.*, 2008.
[26] W. Liu, J. Wang, S. Kumar, and S.-F. Chang, "Hashing with graphs," in *Proc. Int. Conf. Mach. Learn.*, Bellevue, WA, USA, 2011, pp. 1–8.
[27] H. Yang *et al.*, "Adaptive object retrieval with kernel reconstructive hashing," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Columbus, OH, USA, 2014, pp. 1955–1962.
[28] Y. Lin, R. Jin, D. Cai, S. Yan, and X. Li, "Compressed hashing," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Portland, OR, USA, 2013, pp. 446–451.
[29] C. Strecha, A. M. Bronstein, M. M. Bronstein, and P. Fua, "LDAHash: Improved matching with smaller descriptors," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 1, pp. 66–78, Jan. 2012.
[30] B. Kulis and T. Darrell, "Learning to hash with binary reconstructive embeddings," in *Proc. Adv. Neural Inf. Process. Syst.*, 2009, pp. 1042–1050.
[31] M. Norouzi and D. M. Blei, "Minimal loss hashing for compact binary codes," in *Proc. Int. Conf. Mach. Learn.*, Bellevue, WA, USA, 2011, pp. 353–360.
[32] W. Liu, J. Wang, R. Ji, Y.-G. Jiang, and S.-F. Chang, "Supervised hashing with kernels," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Providence, RI, USA, 2012, pp. 2074–2081.
[33] Y. Ke, R. Sukthankar, and L. Huston, "Efficient near-duplicate detection and sub-image retrieval," in *Proc. ACM Multimedia*, New York, NY, USA, 2004, pp. 869–876.
[34] A. Joly and O. Buisson, "A posteriori multi-probe locality sensitive hashing," in *Proc. 16th Int. Conf. Multimedia*, Vancouver, BC, Canada, 2008, pp. 209–218.
[35] A. Joly, O. Buisson, and C. Frelicot, "Content-based copy retrieval using distortion-based probabilistic similarity search," *IEEE Trans. Multimedia*, vol. 9, no. 2, pp. 293–306, Feb. 2007.
[36] M. Muja and D. G. Lowe, "Fast approximate nearest neighbors with automatic algorithm configuration," in *Proc. 4th Int. Conf. Comput. Vis. Theory Appl.*, vol. 1. Lisbon, Portugal, 2009, pp. 331–340.
[37] J. Springer, X. Xin, Z. Li, J. Watt, and A. Katsaggelos, "Forest hashing: Expediting large scale image retrieval," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, Vancouver, BC, Canada, 2013, pp. 1681–1684.
[38] H. Jegou, M. Douze, and C. Schmid, "Hamming embedding and weak geometric consistency for large scale image search," in *Proc. 10th Eur. Conf. Comput. Vis.*, Marseille, France, 2008, pp. 304–317.
[39] L. Zheng, S. Wang, Z. Liu, and Q. Tian, "Packing and padding: Coupled multi-index for accurate image retrieval," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Columbus, OH, USA, 2014, pp. 1947–1954.
[40] G. Tolias, Y. Avrithis, and H. Jegou, "To aggregate or not to aggregate: Selective match kernels for image search," in *Proc. IEEE Int. Conf. Comput. Vis.*, Sydney, NSW, Australia, 2013, pp. 1401–1408.
[41] C. Wengert, M. Douze, and H. Jégou, "Bag-of-colors for improved image search," in *Proc. ACM Multimedia*, Scottsdale, AZ, USA, 2011, pp. 1437–1440.
[42] L. Liu, M. Yu, and L. Shao, "Local feature binary coding for approximate nearest neighbor search," in *Proc. Brit. Mach. Vis. Conf.*, 2015.
[43] G. H. Hardy, J. E. Littlewood, and G. Pólya, *Inequalities*. Cambridge, U.K.: Cambridge Univ. Press, 1952.
[44] G. Griffin, A. Holub, and P. Perona, "Caltech-256 object category dataset," California Inst. Tech., Pasadena, CA, USA, Tech. Rep. CNS-TR-2007-001, 2007.
[45] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba, "SUN database: Large-scale scene recognition from abbey to zoo," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, San Francisco, CA, USA, 2010, pp. 3485–3492.
[46] T. Dean *et al.*, "Fast, accurate detection of 100,000 object classes on a single machine," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Portland, OR, USA, 2013, pp. 1814–1821
[47] Y. Gong and S. Lazebnik, "Iterative quantization: A procrustean approach to learning binary codes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Providence, RI, USA, 2011, pp. 817–824.
[48] J.-P. Heo, Y. Lee, J. He, S.-F. Chang, and S.-E. Yoon, "Spherical hashing," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Providence, RI, USA, 2012, pp. 2957–2964.
[49] H. Jegou, M. Douze, C. Schmid, and P. Perez, "Aggregating local descriptors into a compact image representation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, San Francisco, CA, USA, 2010, pp. 3304–3311.
[50] H. Jegou, M. Douze, and C. Schmid, "Product quantization for nearest neighbor search," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 1, pp. 117–128, Jan. 2011.



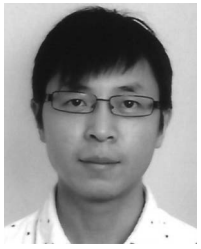
Li Liu received the B.Eng. degree in electronic information engineering from Xi'an Jiaotong University, Xi'an, China, in 2011, and the Ph.D. degree from the Department of Electronic and Electrical Engineering, University of Sheffield, Sheffield, U.K., in 2014.

He is currently a Research Fellow with the Department of Computer Science and Digital Technologies, Northumbria University, Newcastle upon Tyne, U.K. His current research interests include computer vision, machine learning, and data mining.



Mengyang Yu (S'14) received the B.S. and M.S. degrees in mathematics from the School of Mathematical Sciences, Peking University, Beijing, China, in 2010 and 2013, respectively. He is currently pursuing the Ph.D. degree with the Department of Computer Science and Digital Technologies, Northumbria University, Newcastle upon Tyne, U.K.

His current research interests include computer vision, machine learning, and data mining.



Ling Shao (M'09–SM'10) was a Senior Lecturer with the Department of Electronic and Electrical Engineering, University of Sheffield, Sheffield, U.K., from 2009 to 2014, and a Senior Scientist with Philips Research, Eindhoven, The Netherlands, from 2005 to 2009. He is a Professor with the Department of Computer Science and Digital Technologies, Northumbria University, Newcastle upon Tyne, U.K. His current research interests include computer vision, image/video processing, and machine learning.

Dr. Shao is an Associate Editor of the IEEE TRANSACTIONS ON IMAGE PROCESSING, the IEEE TRANSACTIONS ON CYBERNETICS, and several other journals. He is a fellow of the British Computer Society and the Institution of Engineering and Technology.