IET Image Processing

Research Article

Comprehensive analysis and evaluation to unsupervised binary hashing method in image similarity measurement

ISSN 1751-9659
Received on 10th November 2016
Revised 29th March 2017
Accepted on 30th April 2017
E-First on 12th July 2017
doi: 10.1049/iet-ipr.2016.0935
www.ietdl.org

Jin-Bum Kim[1], *Rae-Hong Park*[1] ✉, *Hong-In Kim*[1]

[1]*Department of Electronic Engineering, School of Engineering, Sogang University, 35 Baekbeom-ro (Sinsu-dong), Mapo-gu, Seoul 04107, Korea*

✉ *E-mail: rhpark@sogang.ac.kr*

Abstract: This study proposes an unsupervised binary hashing (UBH) method and provides a comprehensive analysis and evaluation to the UBH method in image similarity. Using the orthogonal locality preserving projection, the proposed UBH method performs the dimensionality reduction (DR). To reduce the quantisation error between low-dimensional vectors and binary hash codes generated in the DR, the proposed UBH method calculates the optimal parameters of rotation and offset. The authors use two test beds to evaluate the preservation of original feature space and the semantic consistency. Experimental results with two test beds show that the proposed UBH method is state of the art.

1 Introduction

Recently, image similarity analysis is getting an attention in the image processing field [1–3]. To measure the image similarity, the nearest neighbour (NN) search calculates distances between a query feature vector with the every feature vectors based on some distance metric such as the Euclidean [4], Hamming [5], or Mahalanobis [6]. Since the NN search uses the full search, the computational load of it is high. Binary hash can reduce the computational load. Various methods have been proposed to reduce the computational load of the NN search. Using the inner product operation, the locality sensitive hashing (LSH) [7, 8] calculates the similarity score between two input feature vectors. The LSH measures the similarity between the raw feature vector and random hyper-plane assumed the multivariate normal distribution. The distance of the binary hash code is equal to +1, if the similarity is positive, otherwise, −1. The shift-invariant kernel LSH (SKLSH) [9] measures the inner product of output vectors of the shift-invariant kernel. The spectral hashing (SH) [10] solves the optimisation problem, where the sum of weighted Hamming distances between binary hash codes is minimised. The spectral relaxation is assumed to solve this NP hard optimisation. The random rotation (RR) [11, 12] and iterative quantisation (ITQ) [13, 14] simply rotate the feature vector to minimise the quantisation error. The dimensionality reduction (DR) step is not included in [11], and elements of hash codes generated using [12] are real numbers. According to Jegou *et al.* [11], the RR matrix or random orthogonal transformation is used to minimise the quantisation error. On the other hand, the ITQ [13, 14] performs the DR using the principal component analysis (PCA), followed by the quantisation error minimisation. Then, the rotation used to minimise the quantisation error is determined by solving the optimisation problem. Existing methods that include the LSH, SKLSH, SH, RR, and ITQ assume that the features vectors are zero-centred in feature space. The reason of this requirement is that the sign function is used for quantisation of feature vectors. However, this assumption cannot be maintained when the feature vectors are not zero-centred inherently.

Using the orthogonal locality preserving projection (OLPP) [15–17], the proposed unsupervised binary hashing (UBH) method preserves the Euclidean metric and the local structures. To find the optimal projection matrix that preserves structures, the OLPP solves the optimisation problem which is represented by the sum of weighted distances between projected raw feature vectors. The weight is determined according to the Euclidean distance between feature vectors. Then, the proposed UBH method minimises the quantisation error between the low-dimensional vector and the binary hash code produced by the OLPP using the optimal rotation and offset.

This paper is an extended version of the work initially published [18]. This paper has four contributions with respect to our previous work [18]. First, the detailed mathematical analysis of the proposed and other existing methods is described. Second, experimental results corresponding to two different features are illustrated to show the generality of the proposed UBH method. Moreover, the big dataset is used in our experiments because the state-of-the-art classification methods are focused on big data. Finally, the performance of the proposed and existing UBH methods is compared using three datasets, two testbeds, and four evaluation measures. The additional experimental results and mathematical analysis also support that the proposed UBH method gives better performance than the existing methods.

The rest of the paper is structured as follows. In Section 2, brief reviews of other existing methods are provided. In Section 3, the proposed UBH method is presented in detail. In Section 4, experimental results and discussions are shown. Finally, conclusion and future work are given in Section 5.

2 Related work

2.1 Locality sensitive hashing

The LSH [7, 8] is one of the basic and widely used binary hashing methods. In the LSH, the binary hashing function $h_m(\boldsymbol{x})$ satisfies two properties [7, 8]. $h_m(\boldsymbol{x})$ satisfying these two properties is defined using the inner product operation between $\boldsymbol{w}_m$ and $\boldsymbol{x}$, where $\boldsymbol{w}_m$ denotes the coefficient vector of the hyper-plane, with each coefficient assumed the multivariate normal distribution [7, 8]. Therefore, the process of the LSH can be summarised as $\boldsymbol{B} = \text{sgn}(\boldsymbol{W}^{\text{T}}\boldsymbol{X})$, where $\boldsymbol{W}$ and $\boldsymbol{X}$ are a collection of $\boldsymbol{w}_m$ and $\boldsymbol{x}_i$, respectively (i.e. $\boldsymbol{W} = [\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_c] \in \boldsymbol{R}^{d \times c}$, $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n] \in \boldsymbol{R}^{d \times n}$). sgn($A$) is the matrix of which the (i, j)-element is $+1$, if the (i, j)-element is greater than or equal to zero, otherwise -1. The LSH works well for long binary hash codes that well preserve the distance between raw feature vectors [19]. Various extended LSH methods were proposed. For example, the SKLSH [9] applies a shift-invariant kernel which has two random values to a binary hashing function $h_m(\boldsymbol{x})$. Each of random values is drawn from uniform distribution with the range of $[0, 2\pi]$ and $[-1, 1]$, respectively [20].

2.2 Spectral hashing

To obtain the best code for the given raw feature vector matrix $\boldsymbol{X}$, the SH [10] solves the optimisation problem, which is expressed as

$$\begin{aligned} \tilde{\boldsymbol{B}} &= \arg\min_{B} \sum_{i,j} A_{ij} \| \boldsymbol{b}_i - \boldsymbol{b}_j \|_B^2, \\ \text{s.t.}\ & \boldsymbol{b}_i \in \{-1, \ 1\}^c, \\ & \sum_i \boldsymbol{b}_i = \boldsymbol{0}, \\ & \frac{1}{n} \boldsymbol{B}\boldsymbol{B}^{\mathrm{T}} = I_c, \end{aligned} \tag{1}$$

where $\boldsymbol{b}_i$ and $\boldsymbol{b}_j$ are binary hash code vectors corresponding to $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, respectively. c represents the length of the binary hash code. $\boldsymbol{I}_c$ signifies the $c \times c$ identity matrix and $\| \cdot \|_B$ denotes the Hamming distance. The affinity coefficient A_{ij} between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ is defined as

$$A_{ij} = \exp\left[-\frac{\| \boldsymbol{x}_i - \boldsymbol{x}_j \|_2^2}{\sigma} \right] \quad (1 \le i, j \le n), \tag{2}$$

where σ signifies a kernel parameter. If $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are close to each other, the Hamming distance between $\boldsymbol{b}_i$ and $\boldsymbol{b}_j$ that are generated by solving (2) is small. The second constraint in (1) represents that the rate of +1 or −1 of the binary hash code matrix is 0.5. The third constraint in (1) signifies that binary hash codes are decorrelated. However, the optimisation problem in (1) is the NP hard problem due to the second constraint [10]. To solve this optimisation problem, the spectral relaxation is used [10]. The limitation of the SH is that it does not work well when raw feature vectors $\boldsymbol{x}_i$ are not separable. In addition, the SH has a poor performance when feature vectors are located far apart from each other. In contrast, the SH is an efficient method in terms of the computational cost, because computation of the covariance matrix is not required.

2.3 Iterative quantisation

The key idea of the RR [11, 12] and ITQ [13, 14] is rotating the feature vectors to minimise the quantisation error that is the squared difference between the low-dimensional vector $\boldsymbol{v}_i$ and the binary hash code $\boldsymbol{b}_i$. To reduce the computational load with high-dimensional raw feature vectors, the ITQ solves the following optimisation problem:

$$\begin{aligned} \tilde{\boldsymbol{W}} &= \arg\max_{W} \sum_{m=1}^{c} \mathrm{var}[h_m(\boldsymbol{x})] \\ \infty &= \sum_{m=1}^{c} \mathrm{var}[\mathrm{sgn}(\boldsymbol{w}_m^{\mathrm{T}}\boldsymbol{x})], \\ \text{s.t.}\ & \frac{1}{n} BB^{\mathrm{T}} = I_c. \end{aligned} \tag{3}$$

The optimal projection matrix $\tilde{\boldsymbol{W}}$ maps the raw feature vector $\boldsymbol{x}_i$ into $\boldsymbol{v}_i$ so that binary hash codes generated using $\boldsymbol{v}_i$ have the maximum variance. In addition, binary hash codes are decorrelated because of the constraint in (3). However, the optimisation problem is intractable, because the objective function is not differentiable [3, 21]. Using the zero-centred assumption and the magnitude relaxation [3, 21], it is modified as

$$\begin{aligned} \tilde{\boldsymbol{W}} &= \arg\max_{W} \sum_{m=1}^{c} E\left[\left| \boldsymbol{w}_m^{\mathrm{T}} \boldsymbol{x} \right|^2 \right] \\ &= \arg\max_{W} \frac{1}{n} \mathrm{tr}\left[W^{\mathrm{T}} \boldsymbol{X}\boldsymbol{X}^{\mathrm{T}} \boldsymbol{W} \right], \\ \text{s.t.}\ & W^{\mathrm{T}}\boldsymbol{W} = I_c, \end{aligned} \tag{4}$$

where $E[\cdot]$ and $\mathrm{tr}[\cdot]$ represent the expectation and trace operations, respectively. The optimisation problem in (4) is exactly equal to the PCA, i.e. called the PCA embedding [13, 14]. The optimal projection matrix $\tilde{\boldsymbol{W}}$ is determined using the top c eigenvectors of the covariance matrix $\boldsymbol{X}\boldsymbol{X}^{\mathrm{T}}$ of the input feature vector matrix. Next, the ITQ minimises the quantisation error that is defined as $\| \mathrm{sgn}(\boldsymbol{v}) - \boldsymbol{v} \|_2^2$. To minimise the quantisation error by rotating the feature vector, the ITQ solves the optimisation problem, which is represented by

$$\begin{aligned} (\tilde{\boldsymbol{B}}, \tilde{\boldsymbol{R}}) &= \arg\min_{B,R} \| \boldsymbol{B} - \boldsymbol{R}\boldsymbol{V} \|_F^2, \\ \text{s.t.}\ & \boldsymbol{R}^{\mathrm{T}}\boldsymbol{R} = I_c \end{aligned} \tag{5}$$

where $\boldsymbol{R}$ is the $c \times c$ rotation matrix, $\| \cdot \|_{\mathrm{F}}$ represents the Frobenius norm. In [13, 14], the alternative algorithm that consists of two steps can solve the optimisation problem in (5). The whole process of the ITQ is summarised as $\tilde{\boldsymbol{B}} = \mathrm{sgn}(\tilde{\boldsymbol{R}}\tilde{\boldsymbol{W}}^{\mathrm{T}}X)$, where $\tilde{\boldsymbol{B}}$, $\tilde{\boldsymbol{R}}$, and $\tilde{\boldsymbol{W}}$ denote the optimal binary hash code, rotation, and projection matrices, respectively [13]. The ITQ has a better performance than the LSH, SH, and RR in terms of the structure preservation of the original feature space and the sematic consistency, when the length of binary hash code c is short [13, 14].

3 Proposed unsupervised binary hashing method

The proposed UBH method consists of DR and quantisation steps. In the DR step, the raw feature vector $\boldsymbol{x}_i$ is projected into the low-dimensional vector $\boldsymbol{v}_i$ using the projection matrix $\boldsymbol{W}$ that has orthogonality and local structure preservation properties. Since $\boldsymbol{v}_i$ should preserve the Euclidean metric structure, the projection matrix $\boldsymbol{W}$ satisfies the orthogonality and maintains the local structure [8, 9]. In the quantisation step, the quantisation error between $\boldsymbol{v}_i$ and the binary hash code $\boldsymbol{b}_i$ is iteratively minimised.

3.1 Dimensionality reduction using OLPP

To generate the low-dimensional vector $\boldsymbol{v}_i$, the concept of the OLPP is used [15–17]. First, the affinity matrix $\boldsymbol{A}$ is constructed. (i, j)th element of the affinity matrix $\boldsymbol{A}$ is defined as [19]

$$A_{ij} = \begin{cases} \exp\left[-\dfrac{\| \boldsymbol{x}_i - \boldsymbol{x}_j \|_2^2}{\sigma} \right] & \text{if } \boldsymbol{x}_i \in N_k(\boldsymbol{x}_j) \text{or} \boldsymbol{x}_j \in N_k(\boldsymbol{x}_i) \\ 0 & \text{otherwise} \quad (1 \le i, j \le n), \end{cases} \tag{6}$$

where σ signifies a kernel parameter. σ determined by the user is to be sufficiently small. The reason is the following. According to Cai *et al.*'s method [15, 16], σ is determined by Belkin and Niyogi's [22] method. In [22], the authors mentioned that the affinity coefficient A_{ij} becomes increasingly localised as the kernel parameter tends to zero, which was proven in [23]. Therefore, we used a sufficiently small value that gives optimal performance in experiments. $N_k(\boldsymbol{x})$ represents the set of the k-NN method of which input is $\boldsymbol{x}$. If $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are neighbours in the k-NN method, A_{ij} is determined by a function of the Euclidean distance between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, otherwise, A_{ij} is zero. Compared with (2), the definition of the affinity coefficient A_{ij} of (6) has an additional NN condition, i.e. the affinity matrix $\boldsymbol{A}$ in (6) reflects the local similarity. Note that

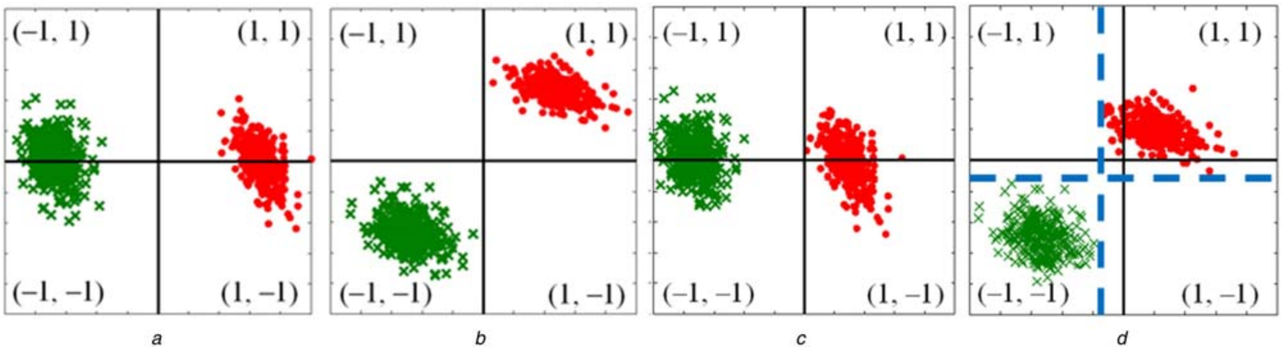

Fig. 1 *Comparing quantisedector using rotation and translation*
(a) Quantisedector vectors classified into two classes that are generated in the DR step, feature vectors satisfy the zero-centred assumption, ***(b)*** Quantisedector vectors that are only rotated, ***(c)*** Quantisedector vectors classified into two classes that are generated in the DR step feature vectors not satisfy the zero-centred assumption, ***(d)*** Quantisedector vectors that are rotated and translated

(6) is equal to (2), if $N_k(\boldsymbol{x})$ includes all the raw feature vectors, i.e. (2) is a special case of (6).

Second, the optimisation problem of the proposed UBH method is expressed as

$$\begin{aligned} \tilde{\boldsymbol{w}}_m &= \arg\min_{\boldsymbol{w}_m} \sum_{i,j} \left(\boldsymbol{w}_m^{\mathrm{T}}\boldsymbol{x}_i - \boldsymbol{w}_m^{\mathrm{T}}\boldsymbol{x}_j\right)^2 A_{ij} \\ &= \arg\min_{\boldsymbol{w}_m} \boldsymbol{w}_m^{\mathrm{T}} XLX^{\mathrm{T}}\boldsymbol{w}_m \\ \text{s.t.} \quad & \boldsymbol{w}_m^{\mathrm{T}}\boldsymbol{XDX}^{\mathrm{T}}\boldsymbol{w}_m = 1, \\ & \boldsymbol{w}_m^{\mathrm{T}}\boldsymbol{w}_1 = \boldsymbol{w}_m^{\mathrm{T}}\boldsymbol{w}_2 = \cdots = \boldsymbol{w}_m^{\mathrm{T}}\boldsymbol{w}_{k-1} = 0, \end{aligned} \tag{7}$$

where $\boldsymbol{w}_m$ is the mth ($1 \leq m \leq c$) column of the projection matrix $\boldsymbol{W}$, and $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{A}$ denotes the graph Laplacian [24]. A diagonal matrix $\boldsymbol{D}$ with diagonal element $D_{ii} = \sum_{j=1}^{n} A_{ij}$ representing the local density around $\boldsymbol{x}_i$ represents the local density matrix. If neighbouring points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are located far away from each other, the objective function (7) is increased. Therefore, if $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are close, $\boldsymbol{v}_i$ and $\boldsymbol{v}_j$ projected by the optimal projection matrix $\tilde{\boldsymbol{W}} = [\tilde{\boldsymbol{w}}_1, \tilde{\boldsymbol{w}}_2, \ldots, \tilde{\boldsymbol{w}}_c]$ are located nearby. Since the objective function (7) has the orthogonality constraint in (7), the Euclidean distance between low-dimensional vectors $\boldsymbol{v}_i$ and $\boldsymbol{v}_j$ is always equal to the Euclidean distance between raw feature vectors $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, i.e. the Euclidean metric is preserved. The second constraint in (7) also represents that binary hash codes are pairwise decorrelated [4, 11–14, 21]. To solve the optimisation problem (7), the proposed UBH method uses Lagrange multipliers [15, 16, 24].

3.2 Quantisation error minimisation

Figs. 1*a* and *b* show examples of rotating low-dimensional feature vectors to minimise the quantisation error [11–14]. Raw feature vectors are produced by a random number generator with two different Gaussian distributions. Vectors in Fig. 1*a* are generated by applying the OLPP to raw feature vectors. Low-dimensional feature vectors can be classified into two classes, where the cross marks and dots represent two different classes. Solid lines denote the boundaries used to assign each bit of the binary hash code $\boldsymbol{b}_i$ to +1 or −1. The first bit of $\boldsymbol{b}_i$ is mapped to +1, if a low-dimensional vector $\boldsymbol{v}_i$ is located in the right side of the vertical axis, otherwise, to −1. The second bit of $\boldsymbol{b}_i$ is assigned to +1 (−1), if a low-dimensional vector $\boldsymbol{v}_i$ is located above (below) the horizontal axis. In Fig. 1*a*, low-dimensional vectors represented by cross marks have two different binary hash codes: (–1, 1) and (–1, −1). Low-dimensional vectors denoted by dots also have two different binary hash codes: (1, 1) and (–1, 1). In Fig. 1*b*, the low-dimensional vector $\boldsymbol{v}_i$ is rotated by the rotation matrix calculated using (5). Rotated low-dimensional vectors $\tilde{\boldsymbol{R}}\boldsymbol{v}_i$ that are denoted by the cross mark are assigned to (–1, −1). $\tilde{\boldsymbol{R}}\boldsymbol{v}_i$ represented by dot are mapped to (1, 1). The quantisation error of Fig. 1*a* is greater than that of Fig. 1*b*. However, the key idea of [11–14] is effective, when low-dimensional vectors satisfy the zero-centred assumption. To satisfy this assumption, the sample mean is subtracted from the raw feature vector [13, 14].

Fig. 1*c* shows low-dimensional feature vectors $\boldsymbol{v}_i$ that do not satisfy the zero-centred assumption. The sample mean of these low-dimensional feature vectors is equal to zero. After applying the key idea in [11–14], the rotated low-dimensional vectors $\tilde{\boldsymbol{R}}\boldsymbol{v}_i$ are illustrated in Fig. 1*d*. The quantisation error of Fig. 1*d* is smaller than that of Fig. 1*c*, if solid lines are boundaries used to assign each bit of the binary hash code $\boldsymbol{b}_i$ to +1 or −1. Note that all $\tilde{\boldsymbol{R}}\boldsymbol{v}_i$ in the class denoted by the cross marks are assigned to (−1, −1). On the other hand, all $\tilde{\boldsymbol{R}}\boldsymbol{v}_i$ in the class represented by the dots are mapped into three different binary hash codes: (1, −1), (–1, 1), and (1, 1). If dashed lines are used to assign each bit of $\boldsymbol{b}_i$ to +1 or −1, $\tilde{\boldsymbol{R}}\boldsymbol{v}_i$ denoted by the cross marks and dots are assigned to (–1, −1) and (1, 1), respectively. In other words, the number of clusters of binary hash codes represented by dots is equal to three, when solid lines are used as decision boundaries that assign each bit of the binary hash code. Whereas all the binary hash codes represented by dots belong to the same class, when dashed lines are used as decision boundaries. Therefore, binary hash codes generated using dashed lines have a smaller quantisation error than those generated using solid lines. The horizontal (vertical) dashed line is the translation result of the horizontal (vertical) solid line in the vertical (horizontal) direction. This translation is equivalent to applying offset to $\tilde{\boldsymbol{R}}\boldsymbol{v}_i$.

The zero-centred assumption means that vectors can be classified using a threshold of zero. Therefore, under this assumption, the sample mean of these vectors is equal to zero. In contrast, some set of vectors cannot be classified using a threshold of zero, even though the sample mean of these vectors is equal to zero. Therefore, the ITQ does not always work well although the sample mean is equal to zero (see Fig. 1*d*). The ITQ can minimise the quantisation error by rotating low-dimensional vectors in Fig. 1*b*. It can also reduce the quantisation error by rotating low-dimensional vectors in Fig. 1*d*, however cannot minimise the quantisation error. The proposed UBH method can minimise the quantisation error by rotating and translating low-dimensional feature vectors.

Then, the quantisation error between the low-dimensional feature vector $\boldsymbol{v}_i$ and the binary hash code $\boldsymbol{b}_i$ is calculated by

$$\begin{aligned} & Q(\boldsymbol{B}, \boldsymbol{R}, \boldsymbol{T}) = \| \boldsymbol{B} - (\boldsymbol{RV} + \boldsymbol{T}) \|_{\mathrm{F}}^2, \\ & \text{s.t.} \qquad \boldsymbol{R}^{\mathrm{T}}\boldsymbol{R} = I_c, \end{aligned} \tag{8}$$

where $\boldsymbol{T} = \boldsymbol{t}\mathbf{1}_n^{\mathrm{T}}$ denotes the $c \times n$ offset matrix and $\mathbf{1}_n$ signifies the $n \times 1$ column vector of which all elements are 1. Compared with ITQ [13, 14], the proposed UBH method considers an additional offset matrix term $\boldsymbol{T}$ in the objective function. The proposed UBH method minimises the quantisation error (8) using an alternative optimisation method. It consists of three steps: (i) fix $\boldsymbol{T}$ and $\boldsymbol{R}$, and update $\boldsymbol{B}$, (ii) fix $\boldsymbol{B}$ and $\boldsymbol{T}$, and update $\boldsymbol{R}$, and (iii) fix $\boldsymbol{R}$ and $\boldsymbol{B}$, and update $\boldsymbol{T}$ steps.

Now the alternative algorithm used in the proposed UBH method is presented in detail. First of all, the objective function in (8) is represented by

$$\begin{aligned} Q(\boldsymbol{B}, \boldsymbol{R}, \boldsymbol{T}) &= \mathrm{tr}[(\boldsymbol{B} - \boldsymbol{RV} - \boldsymbol{T})^{\mathrm{T}}(\boldsymbol{B} - \boldsymbol{RV} - \boldsymbol{T})] \\ &= \| \boldsymbol{B} \|_{\mathrm{F}}^2 + \| \boldsymbol{V} \|_{\mathrm{F}}^2 + \| \boldsymbol{T} \|_{\mathrm{F}}^2 \\ &\quad -2\mathrm{tr}[\boldsymbol{B}^{\mathrm{T}}\boldsymbol{RV}] - 2\mathrm{tr}[\boldsymbol{B}^{\mathrm{T}}\boldsymbol{T}] \\ &\quad +2\mathrm{tr}[\boldsymbol{V}^{\mathrm{T}}\boldsymbol{R}^{\mathrm{T}}\boldsymbol{T}]. \end{aligned} \tag{9}$$

Using the constraint in (8), $\mathrm{tr}[\boldsymbol{V}^{\mathrm{T}}\boldsymbol{R}^{\mathrm{T}}\boldsymbol{RV}]$ is replaced by $\| \boldsymbol{V} \|_{\mathrm{F}}^2$ which is a constant due to fixed $\boldsymbol{V}$ in the DR. $\| \boldsymbol{B} \|_{\mathrm{F}}^2$ is equal to a constant nc [13, 14].

*3.2.1 Fix **T** and **R**, and update **B** step:* Due to fixed $\boldsymbol{R}$ and $\boldsymbol{T}$, in this step, the objective function in (9) is rewritten as

$$\begin{aligned} Q_1(\boldsymbol{B}) &= -2\mathrm{tr}[\boldsymbol{B}^{\mathrm{T}}\boldsymbol{RV}] - 2\mathrm{tr}[\boldsymbol{B}^{\mathrm{T}}\boldsymbol{T}] \\ &= -2\mathrm{tr}[\boldsymbol{B}^{\mathrm{T}}(\boldsymbol{RV} + \boldsymbol{T})] \\ &= -2\sum_{i=1}^{c}\sum_{j=1}^{n} \boldsymbol{B}_{ij}(\boldsymbol{RV} + \boldsymbol{T})_{ij}, \end{aligned}$$

where $\boldsymbol{B}_{ij}$ and $(\boldsymbol{RV} + \boldsymbol{T})_{ij}$ signify the $(i,\ j)$ element of matrices $\boldsymbol{B}$ and $\boldsymbol{RV} + \boldsymbol{T}$, respectively. $Q_1(\boldsymbol{B})$ is easily minimised with respect to $\boldsymbol{B}$, because all elements of $\boldsymbol{B}$ are either -1 or $+1$, i.e. $\boldsymbol{B} \in \{-1, 1\}^{c \times n}$. $\tilde{\boldsymbol{B}}_{ij}$ is equal to 1, if $(\boldsymbol{RV} + \boldsymbol{T})_{ij}$ is a positive real number or zero, otherwise, equal to -1 [13, 14], where $\tilde{\boldsymbol{B}}_{ij}$ represents the $(i,\ j)$ element of the optimal binary code matrix $\tilde{\boldsymbol{B}}$ that minimises $Q_1(\boldsymbol{B})$.

*3.2.2 Fix **B** and **T**, and update **R** step:* In this step, the objective function (9) corresponds to the orthogonal Procrustes problem [25–27], because $\boldsymbol{T}$ and $\boldsymbol{B}$ are fixed. Therefore, the objective function in (9) can be minimised using the singular value decomposition (SVD) [25, 26]. In this step, the proposed UBH algorithm computes the SVD of the $c \times c$ matrix $\boldsymbol{BV}^{\mathrm{T}}$ as $\boldsymbol{BV}^{\mathrm{T}} = \boldsymbol{U}\Sigma\boldsymbol{S}^{\mathrm{T}}$. Then the optimal rotation matrix $\tilde{\boldsymbol{R}}$ is computed as $\tilde{\boldsymbol{R}} = \boldsymbol{US}^{\mathrm{T}}$, where $\boldsymbol{U}$ and $\boldsymbol{S}^{\mathrm{T}}$ are orthogonal matrices, and Σ is a diagonal matrix that consists of singular values.

*3.2.3 Fix **R** and **B**, and update **T** step:* Owing to fixed $\boldsymbol{B}$ and $\boldsymbol{R}$, the objective function in (9) is represented in quadratic forms:

$$\begin{aligned} Q_2(\boldsymbol{t}) &= -2\mathrm{tr}[(\boldsymbol{B} - \boldsymbol{RV})^{\mathrm{T}}\boldsymbol{T}] + \mathrm{tr}[\boldsymbol{T}^{\mathrm{T}}\boldsymbol{T}] \\ &= -2\mathbf{1}_n^{\mathrm{T}}(\boldsymbol{B} - \boldsymbol{RV})^{\mathrm{T}}\boldsymbol{t} + n\boldsymbol{t}^{\mathrm{T}}\boldsymbol{t}. \end{aligned}$$

The solution $\tilde{\boldsymbol{t}}$, i.e. the optimal offset that is used to minimise $Q_2(\boldsymbol{t})$, can be obtained by setting the derivative of $Q_2(\boldsymbol{t})$ with respect to $\boldsymbol{t}$ to zero, i.e.

$$\frac{\partial Q_2(\boldsymbol{t})}{\partial \boldsymbol{t}} = -2(\boldsymbol{B} - \boldsymbol{RV})\mathbf{1}_n + 2n\boldsymbol{t} = 0,$$

which gives the optimal offset $\tilde{\boldsymbol{t}} = (1/n)(\boldsymbol{B} - \boldsymbol{RV})\mathbf{1}_n$.

3.3 Quantisation error comparison with ITQ

The quantisation error Q^{P} of the proposed UBH method is smaller than or equal to the quantisation error Q^{I} of the ITQ [13, 14]. Using (9), Q^{P} is represented as

$$\begin{aligned} Q^{\mathrm{P}} &= \| \tilde{\boldsymbol{B}}_1 \|_{\mathrm{F}}^2 + \| \boldsymbol{V} \|_{\mathrm{F}}^2 + \| \tilde{\boldsymbol{T}} \|_{\mathrm{F}}^2 \\ &\quad -2\mathrm{tr}[\tilde{\boldsymbol{B}}_1^{\mathrm{T}}\tilde{\boldsymbol{R}}_1\boldsymbol{V}] - 2\mathrm{tr}[\tilde{\boldsymbol{B}}_1^{\mathrm{T}}\tilde{\boldsymbol{T}}] + 2\mathrm{tr}[\boldsymbol{V}^{\mathrm{T}}\tilde{\boldsymbol{R}}_1^{\mathrm{T}}\tilde{\boldsymbol{T}}], \end{aligned}$$

where $\tilde{\boldsymbol{B}}_1$, $\tilde{\boldsymbol{T}}$, and $\tilde{\boldsymbol{R}}_1$ represent the optimal hash code, the optimal offset matrices, and the optimal rotation of the proposed UBH method, respectively. According to [13, 14], Q^{I} is expressed as

$$\begin{aligned} Q^{\mathrm{I}} &= \| \tilde{\boldsymbol{B}}_2 - \tilde{\boldsymbol{R}}_2\boldsymbol{V} \|_{\mathrm{F}}^2 \\ &= \| \tilde{\boldsymbol{B}}_2 \|_{\mathrm{F}}^2 + \| \boldsymbol{V} \|_{\mathrm{F}}^2 - 2\mathrm{tr}[\tilde{\boldsymbol{B}}_2^{\mathrm{T}}\tilde{\boldsymbol{R}}_2\boldsymbol{V}], \end{aligned}$$

where $\tilde{\boldsymbol{B}}_2$ and $\tilde{\boldsymbol{R}}_2$ denote the optimal hash code matrix and the optimal rotation matrix of the ITQ [13, 14], respectively. The subtraction of Q^{I} from Q^{P} is written as

$$\begin{aligned} Q^{\mathrm{P}} - Q^{\mathrm{I}} &= \| \tilde{\boldsymbol{T}} \|_{\mathrm{F}}^2 - 2\mathrm{tr}[(\tilde{\boldsymbol{B}}_1 - \tilde{\boldsymbol{R}}_1\boldsymbol{V})^{\mathrm{T}}\tilde{\boldsymbol{T}}] \\ &\quad +2(\mathrm{tr}[\tilde{\boldsymbol{B}}_2^{\mathrm{T}}\tilde{\boldsymbol{R}}_2\boldsymbol{V}] - \mathrm{tr}[\tilde{\boldsymbol{B}}_1^{\mathrm{T}}\tilde{\boldsymbol{R}}_1\boldsymbol{V}]) \\ &= n\tilde{\boldsymbol{t}}^{\mathrm{T}}\tilde{\boldsymbol{t}} - 2\mathbf{1}_n^{\mathrm{T}}(\tilde{\boldsymbol{B}}_1 - \tilde{\boldsymbol{R}}_1\boldsymbol{V})\tilde{\boldsymbol{t}} \\ &\quad +2(\mathrm{tr}[\tilde{\boldsymbol{B}}_2^{\mathrm{T}}\tilde{\boldsymbol{R}}_2\boldsymbol{V}] - \mathrm{tr}[\tilde{\boldsymbol{B}}_1^{\mathrm{T}}\tilde{\boldsymbol{R}}_1\boldsymbol{V}]), \end{aligned} \tag{10}$$

where $\| \tilde{\boldsymbol{B}}_1 \|_{\mathrm{F}}^2$ and $\| \tilde{\boldsymbol{B}}_2 \|_{\mathrm{F}}^2$ are equal to nc. After putting the optimal offset $\tilde{\boldsymbol{t}}$ into the first and second terms in (10), the sum of these terms is simplified as $-(1/n)\| (\tilde{\boldsymbol{B}}_1 - \tilde{\boldsymbol{R}}_1\boldsymbol{V})\mathbf{1}_n \|_2^2$, which is smaller than or equal to zero. Next, the third term in parentheses in (10) can be represented by

$$\begin{aligned} \mathrm{tr}[\tilde{\boldsymbol{B}}_2^{\mathrm{T}}\tilde{\boldsymbol{R}}_2\boldsymbol{V}] - \mathrm{tr}[\tilde{\boldsymbol{B}}_1^{\mathrm{T}}\tilde{\boldsymbol{R}}_1\boldsymbol{V}] &= \mathrm{tr}[\tilde{\boldsymbol{B}}_2^{\mathrm{T}}\boldsymbol{U}_2\boldsymbol{S}_2^{\mathrm{T}}\boldsymbol{V}] - \mathrm{tr}[\tilde{\boldsymbol{B}}_1^{\mathrm{T}}\boldsymbol{U}_1\boldsymbol{S}_1^{\mathrm{T}}\boldsymbol{V}] \\ &= \mathrm{tr}[\boldsymbol{S}_2^{\mathrm{T}}\boldsymbol{V}\tilde{\boldsymbol{B}}_2^{\mathrm{T}}\boldsymbol{U}_2] - \mathrm{tr}[\boldsymbol{S}_1^{\mathrm{T}}\boldsymbol{V}\tilde{\boldsymbol{B}}_1^{\mathrm{T}}\boldsymbol{U}_1] \\ &= \mathrm{tr}[\boldsymbol{S}_2^{\mathrm{T}}\boldsymbol{S}_2\Sigma_2\boldsymbol{U}_2^{\mathrm{T}}\boldsymbol{U}_2] - \mathrm{tr}[\boldsymbol{S}_1^{\mathrm{T}}\boldsymbol{S}_1\Sigma_1\boldsymbol{U}_1^{\mathrm{T}}\boldsymbol{U}_1] \\ &= \mathrm{tr}[\Sigma_2] - \mathrm{tr}[\Sigma_1]. \end{aligned} \tag{11}$$

$\tilde{\boldsymbol{R}}_2$ is replaced by $\boldsymbol{U}_2\boldsymbol{S}_2^{\mathrm{T}}$ [14, 15], and $\tilde{\boldsymbol{R}}_1$ is substituted by $\boldsymbol{U}_1\boldsymbol{S}_1^{\mathrm{T}}$ as in step (ii) of the alternative algorithm. $\tilde{\boldsymbol{B}}_1\boldsymbol{V}^{\mathrm{T}}$ and $\tilde{\boldsymbol{B}}_2\boldsymbol{V}^{\mathrm{T}}$ are decomposed into $\boldsymbol{U}_2\Sigma_2\boldsymbol{S}_2^{\mathrm{T}}$ and $\boldsymbol{U}_1\Sigma_1\boldsymbol{S}_1^{\mathrm{T}}$ using the SVD, respectively. So, (11) is calculated by sum of singular values of $\tilde{\boldsymbol{B}}_1\boldsymbol{V}^{\mathrm{T}}$ and $\tilde{\boldsymbol{B}}_2\boldsymbol{V}^{\mathrm{T}}$, which can be rewritten as $\mathrm{tr}\left[\sqrt{\boldsymbol{V}\tilde{\boldsymbol{B}}_2^{\mathrm{T}}\tilde{\boldsymbol{B}}_2\boldsymbol{V}^{\mathrm{T}}}\right] - \mathrm{tr}\left[\sqrt{\boldsymbol{V}\tilde{\boldsymbol{B}}_1^{\mathrm{T}}\tilde{\boldsymbol{B}}_1\boldsymbol{V}^{\mathrm{T}}}\right]$ using the definition of the nuclear norm. Owing to pairwise decorrelation of hash code vectors, $\tilde{\boldsymbol{B}}_1^{\mathrm{T}}\tilde{\boldsymbol{B}}_1$ and $\tilde{\boldsymbol{B}}_2^{\mathrm{T}}\tilde{\boldsymbol{B}}_2$ are equal to nI_n. Then, (11) is equal to zero.

Thus, because $Q^{\mathrm{P}} - Q^{\mathrm{I}}$ is smaller than or equal to zero, the proposed UBH method has lower quantisation error than ITQ [13, 14].

4 Experimental results and discussions

4.1 Datasets, test beds, and evaluation methods

The first test bed is to evaluate the performance of the NN search. The ground truth is acquired using the Euclidean distance. Datasets used in this experiment are CIFAR-10 dataset [28, 29], SIFT-1M dataset [4, 30, 31], and GIST-1M dataset [4, 31]. Each of detailed datasets is described in Table 1. In experiments on CIFAR-10 dataset, GIST features [32] are used as raw features. Parameters of GIST are the same as used in [13, 14], i.e. the dimension of the raw feature vector is equal to 320 (320-D GIST features). In each experiment, 1000 raw feature vectors are randomly selected and the remaining raw feature vectors are included in the training set. The first test bed is used to show that the low-dimensional binary hash codes calculated by the proposed UBH method well preserves the Euclidean metric and the local structures.

Also, the second test bed is to evaluate the performance of the NN search with CIFAR-10 dataset [28, 29]. However, the ground

Table 1 Details of the datasets used in experiments

Dataset	CIFAR-10	GIST-1M	SIFT-1M
number of elements	60,000 training set: 50,000 test set: 10,000	1,000,000 training set: 100,000 test set: 900,000	1,000,000 training set: 500,000 test set: 500,000
element type	image	GIST feature vector	SIFT feature vector
dimension	32 × 32	960	128
semantic class label	O	×	×
purpose	object recognition	evaluating the quality of NN search	evaluating the quality of NN search
remark	[25, 26]	[3, 26]	[3, 27, 28]

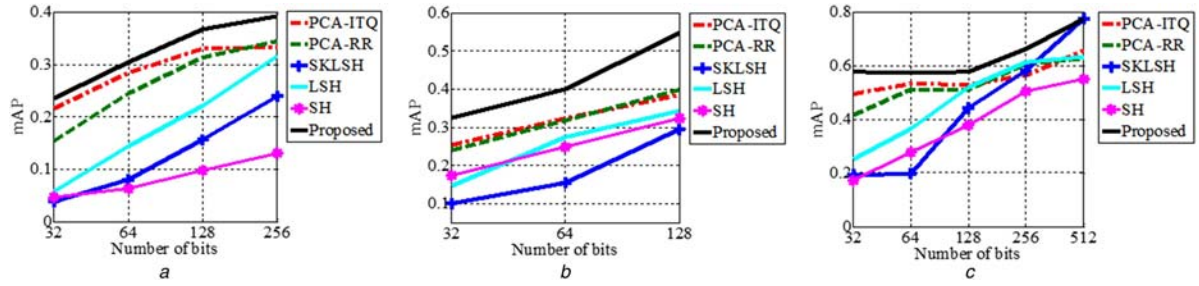

Fig. 2 *mAP comparisons of the proposed UBH method and other existing methods*
(a) CIFAR-10 dataset, *(b)* SIFT-1M dataset, *(c)* GIST-1M dataset

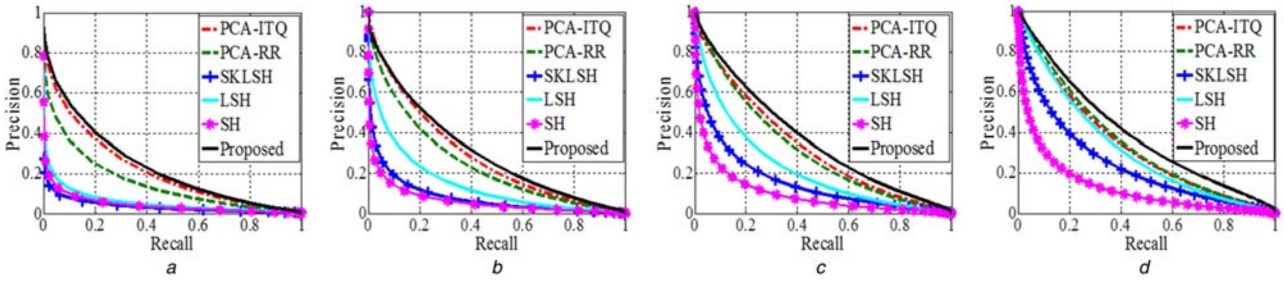

Fig. 3 *Recall–precision curves of the proposed UBH method and other existing methods (CIFAR-10 dataset)*
(a) 32 bit cases, *(b)* 64 bit cases, *(c)* 128 bit cases, *(d)* 256 bit cases

truth is class labels. A total of 1000 raw feature vectors are randomly chosen, and the rest of raw feature vectors form training set. For a given testing vector, vectors included in ground truth are training vectors that have the same class label of a given testing vector. We show the result of the second test bed with CIFAR-10 dataset only, because the second test bed needs semantic class labels, where the semantic class label means that the class label represents the semantics of the class of the raw feature vector. The second test bed is used to show that the low-dimensional binary hash codes generated by the proposed UBH method maintain the semantic consistency, where the semantic consistency is the factor that makes semantic class labels of the binary hash code and those of the raw feature vector be the same.

Now, evaluation measures used in our experiments are briefly reviewed. The precision pre and the recall re are defined as the ratio of the number of true-positive vectors to the number of returned vectors and the number of vectors in ground truth, respectively. The average precision (AP) is defined as the average value of precision values obtained from certain trials, in which a vector in ground truth was retrieved. Then, the mean average precision (mAP) is defined as the average of AP values for all queries. Also, all the experimental results are averaged over ten experiments.

4.2 Results of the first test bed

In the first test bed, the proposed UBH method is compared with other existing methods. The PCA-RR and PCA-ITQ use the PCA in the DR, and then minimise the quantisation error using the RR [11, 12] and the ITQ [13, 14], respectively. Fig. 2 shows mAP comparisons with three different datasets: CIFAR-10, SIFT-1M, and GIST-1M datasets. The mAP comparisons with CIFAR-10 dataset are shown in Fig. 2*a*, in which the mAP of the proposed UBH method is the highest for 32, 64, 128, and 256 bit cases. Fig. 2*b* shows the mAP comparisons with SIFT-1M dataset. The mAP of the proposed UBH method is higher than those of other existing methods for 32, 64, and 128 bit cases. The mAP comparisons with GIST-1M dataset are illustrated in Fig. 2*c*, in which the proposed UBH method has the highest mAP for 32, 64, 128, 256, and 512 bit cases. In these cases, because the proposed UBH method preserves Euclidean metric and local structures in the DR step and the quantisation error between low-dimensional vector $\boldsymbol{v}$ and the hash code $\boldsymbol{b}$ is minimised in the quantisation step, hash codes generated by the proposed UBH method well preserve the structure of the raw feature space. Note that mAP values of the proposed UBH method for GIST-1M dataset are similar, when the lengths of binary hash codes are 32, 64, and 128. The reason is that the space of binary hash codes is insufficient to maintain the structure of the raw feature space. However, mAP values of the proposed UBH method increase, when the length of binary hash code is >128. The increment of the mAP of the SKLSH and LSH is large. The SKLSH and LSH work well for long length binary hash codes, because long binary hash codes well preserve the distance between raw feature vectors [13, 20].

Figs. 3–5 show recall–precision curves of the proposed UBH method and other existing methods with CIFAR-10, SIFT-1M, and GIST-1M datasets, respectively, for different lengths of binary hash codes. The proposed UBH method has higher recall and precision values than other existing methods, which represents that the proposed UBH method preserves structure more properly. Note that the improvement of the performance of the long binary hash code is large, because the long hash code can preserve well the Euclidean metric and the local structure.

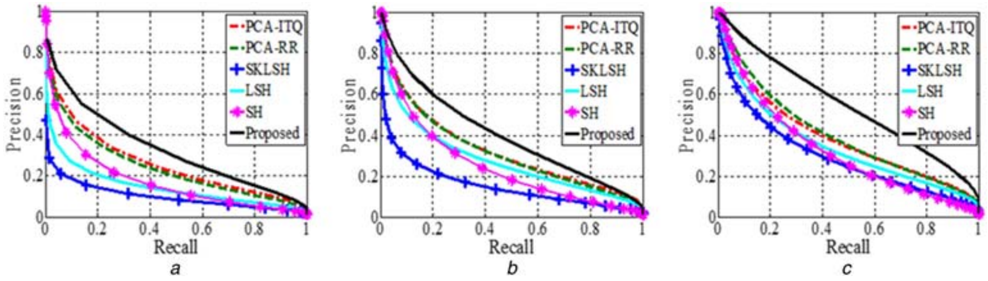

Fig. 4 *Recall–precision curves of the proposed UBH method and other existing methods (SIFT-1M dataset)*
(a) 32 bit cases, ***(b)*** 64 bit cases, ***(c)*** 128 bit cases

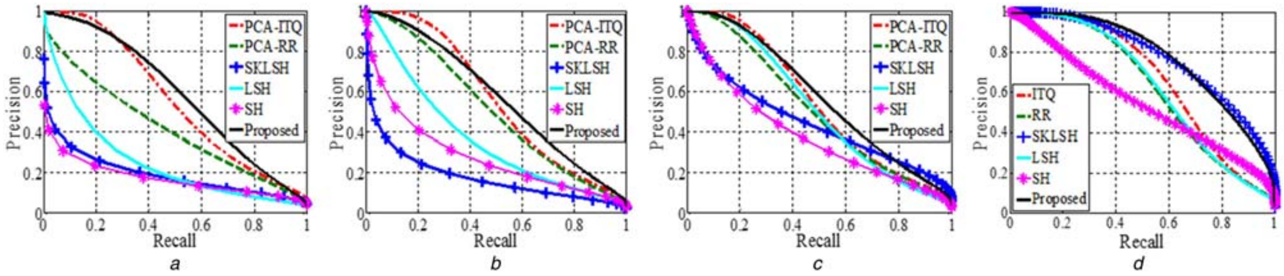

Fig. 5 *Recall–precision curves of the proposed UBH method and other existing methods (SIFT-1M dataset)*
(a) 32 bit cases, ***(b)*** 64 bit cases, ***(c)*** 128 bit cases, ***(d)*** 512 bit cases

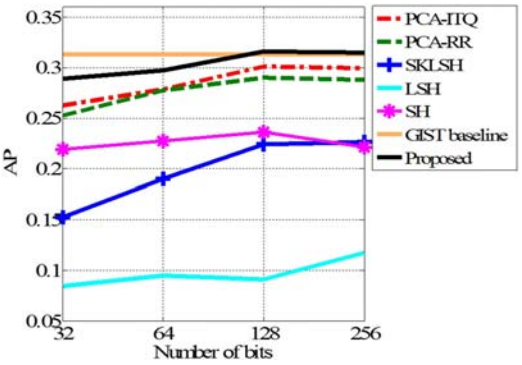

Fig. 6 *AP comparisons of the proposed UBH method and other existing methods on CIFAR-10 dataset*

For the shortest binary hash code case (32 bit cases), the PCA-ITQ has a better performance in terms of precision/recall and mAP than the PCA-RR. However, the PCA-ITQ and PCA-RR have the similar performance, when the length of hash code increases. Especially, the PCA-RR shows a better performance than the PCA-ITQ for the longest binary hash code case (256 bit cases of CIFAR-10 dataset and 128 bit cases of SIFT-1M dataset). When the hash code length increases, it is noted that the PCA-RR and PCA-ITQ similarly maintain the structure of raw feature space. However, the PCA-ITQ well maintains the structure of raw feature space for 32 bit binary hash code. The longer the binary hash code, the better the proposed UBH method than the PCA-ITQ and PCA-RR. Note that the structure of raw feature space is well preserved by the proposed UBH method than the PCA-RR and PCA-ITQ, regardless of the length of binary hash code.

For SIFT-1M dataset (see Fig. 4), the LSH has a higher precision than the SH, when the recall is low. On the other hand, the SH has a higher recall than the LSH, when the precision is low. Therefore, the LSH and SH are suitable for applications that need a high precision and high recall, respectively. However, these methods are not suitable for applications that simultaneously require high precision and recall.

For GIST-1M dataset (see Figs. 5*a–c*), the PCA-ITQ has a higher precision than the proposed UBH method, when the recall is low. On the other hand, the proposed UBH method has a higher recall than the PCA-ITQ, when the precision is low. Therefore, the proposed UBH method (PCA-ITQ) is more suitable for the application that needs a high recall (precision) value and large data compression. In Fig. 5*d*, the SKLSH has a recall–precision curve similar to that of the proposed UBH method. The reason is that the binary hash code space is sufficient to preserve the structures of the raw feature space.

4.3 Results of the second test bed

Fig. 6 shows AP comparisons of the proposed and existing UBH methods on CIFAR-10 dataset. Using class labels as ground truth, we evaluate the semantic consistency of binary hash codes calculated by the proposed and existing UBH methods. In all the lengths of binary hash codes, the proposed UBH method gives the highest AP values. The reason is that the semantic information loss caused by the DR and quantisation error of the proposed UBH method is smaller than that of the existing methods. Owing to the semantic information loss, the semantic class label of the binary hash code is not the same as that of the raw feature vector. Therefore, the proposed UBH method has smaller semantic information loss than other existing methods. In other words, the proposed UBH method has a better performance than other existing methods in terms of the semantic consistency.

Fig. 7 shows the precision values of the second test bed with six different numbers of returned binary hash code vectors: 1, 5, 50, 100, 200, and 500. To avoid clutter, the precision values of the LSH are not shown in Fig. 7. The LSH has the smallest precision values, because it has the smallest AP values for all the lengths of binary hash codes (see Fig. 6). In Fig. 7 the proposed UBH method always has the highest precision value, compared with other existing methods. Therefore, the proposed UBH method has a better performance in terms of the precision than other existing methods for all the numbers of returned binary hash codes (see Figs. 7*a–d* for 32, 64, 128, and 256 bit cases, respectively).

5 Conclusion

The proposed UBH method preserves Euclidean metric and the local structures. To preserve these structures, the OLPP in the DR step is used. Then, in the quantisation step, the optimal rotation and offset are optimised to minimise the quantisation error, in which an iterative alternative algorithm composed of three steps is used. Experimental results of the first test bed show that the proposed UBH method has smaller quantisation errors and well preserves the structure. In experimental results of the second test bed, the proposed UBH method has a higher semantic consistency than other existing methods. Especially, the proposed UBH method minimises semantic information loss, when the length of binary

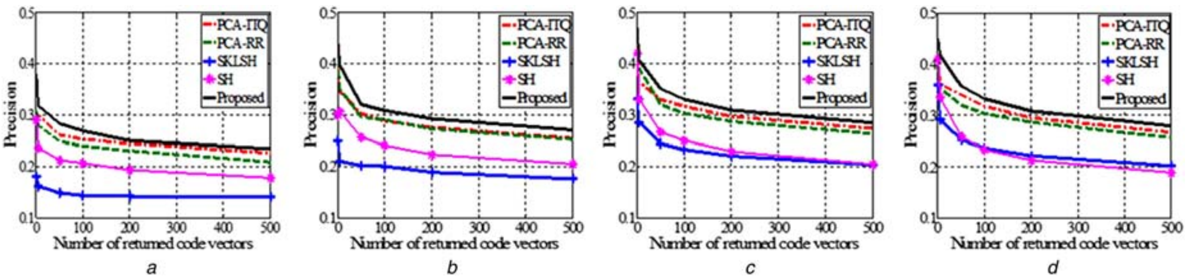

Fig. 7 *Precision comparisons of the proposed UBH method and other existing methods as a function of the number of returned code vectors (CIFAR-10 dataset)*
(a) 32 bit cases, ***(b)*** 64 bit cases, ***(c)*** 128 bit cases, ***(d)*** 256 bit cases

hash cods is ≥128. Future work will focus on the development of a supervised binary hashing method.

6 Acknowledgment

This work was supported in part by Brain Korea 21 Plus.

7 References

[1] Wang, J., Liu, W., Kumar, S.*, et al.*: 'Learning to hash for indexing big data – a survey', *Proc. IEEE*, 2016, **104**, (1), pp. 34–57

[2] Zhang, L., Zhang, Y., Gu, X.*, et al.*: 'Scalable similarity search with topology preserving hashing', *IEEE Trans. Image Process.*, 2014, **23**, (7), pp. 3025–3039

[3] Wang, J., Kumar, S., Chang, S.-F.: 'Semi-supervised hashing for large-scale search', *IEEE Trans. Pattern Anal. Mach. Intell.*, 2012, **34**, (12), pp. 2393–2406

[4] Jegou, H., Douze, M., Schmid, C.: 'Product quantization for nearest neighbor search', *IEEE Trans. Pattern Anal. Mach. Intell.*, 2011, **33**, (1), pp. 117–128

[5] Esmaeili, M.M., Ward, R.K., Fatourechi, M.: 'A fast approximate nearest neighbor search algorithm in the Hamming space', *IEEE Trans. Pattern Anal. Mach. Intell.*, 2012, **34**, (12), pp. 2481–2488

[6] Ramaswamy, S., Rose, K.: 'Fast adaptive mahalanobis distance-based search and retrieval in image databases'. Proc. IEEE Int. Conf. Image Processing, San Diego, CA, USA, October 2008, pp. 181–184

[7] Lv, Q., Josephson, W., Wang, Z.*, et al.*: 'Multi-probe LSH: efficient indexing for high dimensional similarity search'. Proc. ACM Int. Conf. Very Large Data Base Endowment, Vienna, Austria, September 2007, pp. 950–961

[8] Andoni, A., Indyk, P.: 'Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions'. Proc. IEEE Symp. Foundation of Computer Science, Berkeley, CA, USA, October 2006, pp. 459–468

[9] Raginsky, M., Lazebnik, S.: 'Locality sensitive binary codes from shift-invariant kernels'. Proc. Neural Information Processing Systems, Vancouver, BC, Canada, December 2009, pp. 1509–1517

[10] Weiss, Y., Torralba, A., Fergus, R.: 'Spectral hashing'. Proc. Neural Information Processing Systems, Vancouver, BC, Canada, December 2009, pp. 1753–1760

[11] Jegou, H., Douze, M., Schmid, C.: 'Hamming embedding and weak geometric consistency for large scale image search'. Proc. European Conf. Computer Vision, Marseille, France, October 2008, pp. 304–317

[12] Jegou, H., Douze, M., Schmid, C.*, et al.*: 'Aggregating local descriptors into a compact image representation'. Proc. IEEE Conf. Computer Vision and Pattern Recognition, San Francisco, CA, USA, June 2010, pp. 3304–3311

[13] Gong, Y., Lazebnik, S., Gordo, A.*, et al.*: 'Iterative quantization: a Procrustean approach to learning binary codes for large-scale image', *IEEE Trans. Pattern Anal. Mach. Intell.*, 2013, **35**, (12), pp. 2916–2929

[14] Gong, Y., Lazebnik, S.: 'Iterative quantization: a Procrustean approach to learning binary codes'. Proc. IEEE Computer Vision and Pattern Recognition, Providence, RI, USA, June 2011, pp. 817–824

[15] Cai, D., He, X., Han, J.*, et al.*: 'Orthogonal laplacianfaces for face recognition', *IEEE Trans. Image Process.*, 2006, **15**, (11), pp. 3608–3614

[16] Cai, D., He, X.: 'Orthogonal locality preserving indexing'. Proc. ACM Int. Conf. Special Interest Group on Information Retrieval, Salvador, Brazil, August 2005, pp. 3–10

[17] Shikkenawis, G., Mitra, S.K.: '2d orthogonal locality preserving projection for image denoising', *IEEE Trans. Image Process.*, 2016, **25**, (1), pp. 262–273

[18] Kim, J.-B., Park, R.-H.: 'Unsupervised binary hashing method using locality preservation and quantisation error minimization', *Electron. Lett.*, 2015, **51**, (3), pp. 255–257

[19] Liu, C., Ling, H., Zou, F.*, et al.*: 'Nonnegative sparse locality preserving hashing', *Inf. Sci.*, 2014, **281**, (10), pp. 714–725

[20] Gordo, A., Perronnin, F., Gong, Y.*, et al.*: 'Asymmetric distances for binary embeddings', *IEEE Trans. Pattern Anal. Mach. Intell.*, 2014, **36**, (1), pp. 33–47

[21] Wang, J., Kumar, S., Chang, S.-F.: 'Semi-supervised hashing for scalable image retrieval'. Proc. IEEE Conf. Computer Vision and Pattern Recognition, San Francisco, CA, USA, June 2010, pp. 3424–3431

[22] Belkin, M., Niyogi, P.: 'Laplacian eigenmaps and spectral techniques for embedding and clustering'. Proc. Neural Information Processing Systems, Vancouver, BC, Canada, December 2006, pp. 585–591

[23] Belkin, M., Niyogi, P.: 'Convergence of Laplacian eigenmaps'. Proc. Neural Information Processing Systems, Vancouver, BC, Canada, December 2006, pp. 129–136

[24] Yan, S., Xu, D., Zhang, B.*, et al.*: 'Graph embedding and extensions: a general framework for dimensionality reduction', *IEEE Trans. Pattern Anal. Mach. Intell.*, 2007, **29**, (1), pp. 40–51

[25] Schonemann, P.: 'A generalized solution of the orthogonal Procrustes problem', *Psychometrika*, 1966, **31**, (1), pp. 1–10

[26] Arun, K.S., Huang, T.S., Blostein, S.D.: 'Least-squares fitting of two 3-D point sets', *IEEE Trans. Pattern Anal. Mach. Intell.*, 1987, **9**, (3), pp. 698–700

[27] Nemirovski, A.: 'Sums of random symmetric matrices and quadratic optimization under orthogonality constraints', *Math. Program.*, 2006, **109**, (2–3), pp. 283–317

[28] Feng, W., Jia, B., Zhu, M.: 'Deep hash: semantic similarity preserved hash scheme', *Electron. Lett.*, 2014, **50**, (19), pp. 1347–1349

[29] Kim, S., Choi, S.: 'Semi-supervised discriminant hashing'. Proc. IEEE Int. Conf. Data Mining, Vancouver, BC, Canada, December 2011, pp. 1122–1127

[30] Pauleve, L., Jegou, H., Amsaleg, L.: 'Locality sensitive hashing: a comparison of hash function types and querying mechanisms', *Pattern Recognit. Lett.*, 2010, **31**, (11), pp. 1348–1358

[31] Brandt, J.: 'Transform coding for fast approximate nearest neighbor search in high dimensions'. Proc. Conf. Computer Vision and Pattern Recognition, San Francisco, CA, USA, June 2010, pp. 1815–1822

[32] Mikolajczyk, L., Schmid, C.: 'A performance evaluation of local descriptors', *IEEE Trans. Pattern Anal. Mach. Intell.*, 2005, **27**, (10), pp. 1615–1630