Int. J. Mach. Learn. & Cyber. (2017) 8:571–586
DOI 10.1007/s13042-015-0350-9

ORIGINAL ARTICLE

Sequential conditional entropy maximization semi-supervised hashing for semantic image retrieval

Wing W. Y. Ng[1] · Yueming Lv[1] · Ziqian Zeng[1] · Daniel S. Yeung[1] · Patrick P. K. Chan[1]

Received: 3 November 2014 / Accepted: 13 March 2015 / Published online: 26 March 2015

Abstract Hashing has been widely applied to large scale semantic image retrieval. Unsupervised hashing cannot work well for semantic image retrieval while supervised hashing requiring full label information for large databases is not practical. Semi-supervised hashing (SSH) solves this problem by learning the semantic information from a small portion of labeled images and the data structure information from the unlabeled images. The major drawback of the current SSH is that they cannot guarantee the maximization of entropy over all hash bits for a better coding efficiency. We propose a SSH which maximizes the conditional entropy of a bit with respect to all previous bits, i.e. the sequential conditional entropy maximization SSH. It is further extended to a nonlinear SSH with a new mapping method to enhance precision and recall rates. Experimental results show that the nonlinear SSH does not work well for all cases, and a heuristic guideline for the selection between linear and nonlinear hashing is given based on the characteristics of the database. We also propose a multiple hashing version of the proposed method for high recall rate with few hash bucket visits. Experimental results show that the proposed method outperforms current SSH methods.

Keywords Hashing · Semi-supervised hashing · Semantic image retrieval

✉ Wing W. Y. Ng
wingng@ieee.org

[1] School of Computer Science and Engineering, South China University of Technology, Guangzhou, China

1 Introduction

Retrieving information from the exponentially growing data on the Internet is a challenging problem. For large scale problems, linear search speed is no longer acceptable. Tree based methods, such as k-d tree [1, 2], M-trees [3], cover trees [4] and metric trees [5], cannot yield satisfactory results because the search time increases very quickly with the number of dimensions. In some cases, data structures being created are larger than the original data [6]. In large scale information retrieval problems, even linear time search methods (e.g. [7]) are too time consuming and not practical. In contrast, hashing methods provide a fast sublinear image search time complexity and require much less storage space.

Hashing methods could be categorized into: unsupervised, supervised and semi-supervised methods. The locality sensitive hashing (LSH) [8–10], the spectral hashing [11], the anchor graph hashing [12] and the SPICA [13], the iterative quantization (ITQ) [14], the Cartesian K-means [15], the compressed hashing [16], the K-means hashing [17] and the bilinear hashing [18], are all instances of unsupervised hashing. They build hash codes by generating hash bits either randomly or by the data distribution (e.g. principal components). However, unsupervised hashing methods cannot deal with semantic image retrieval problems. Supervised hashing methods generate hash bits to preserve the semantic similarity using pairwise similarity label information of the images. The kernel-based supervised hashing [19], the LDA hashing [20], the restricted Boltzmann machine hashing [21] and the minimum loss hashing [22] are all instances of supervised hashing. The major drawback of supervised hashing methods is the need of a huge number of pairwise similarity labels among images. In practice, pairwise similarity labels among most

images are not available while a small portion of them may be labeled manually. The requirement of full labels may be an over-shoot and impractical for a large scale problem. Recently, a regularized framework is proposed to address this problem [23]. It uses the non-negative matrix factorization and the holistic visual diversity minimization to complete the missing tags for social images.

The semi-supervised hashing (SSH) [24] is proposed to deal with the image retrieval problems with mixed labeled and unlabeled images. It preserves semantic similarity of labeled images and regularizes learned hash bits using unlabeled images to prevent overfitting. Both the SSH and the bootstrap sequential projection learning hashing (BSPLH) [25] maximize the lower bound of the maximum variance of each bit as the regularizer to find the maximum entropy partition. However, this regularizer is computed by the sum of lower bounds of maximum entropy of each bit only and cannot maximize the joint entropy of all bits. The joint entropy of all bits will be maximized with this optimization method if and only if all hash bits are independent. However, in most of the real applications, variances concentrate in a low dimensional subspace formed by a small number of top eigenvectors of a principal component analysis (PCA). In order to achieve a better image retrieval performance, hash bits should be allowed to be correlated, i.e. they are not independent. For instance, the PCA of the MNIST database (784 dimensions) shows that 88.88, 78.23 and 43.82 % of variances of data are concentrated in the top 10, 5 and 1 % of dimensions, respectively. Hence, the bit independence assumption of both the SSH [24] and the BSPLH [25] may not be appropriate in real cases.

Figure 1 shows three cases of hash bits generated by maximizing the sum of lower bound of maximum entropies of two hash bits. When the orthogonality (or independence) of hash bits (functions) is weakened (Fig. 1b, c), the regularizer of both the SSH and the sequential projection learning hashing (SPLH) cannot give a good approximation of the joint entropy of bits.

Therefore, we propose the SSH using Sequential Conditional Entropy Maximum (SCEM-SSH) to maximize the joint entropy sequentially. During the learning of a hash bit, the SCEM-SSH maximizes the conditional entropy and corrects errors jointly created by all previous bits. Hence, the SCEM-SSH cuts the input space in the direction yielding the largest joint entropy of all hash bits. The proposed

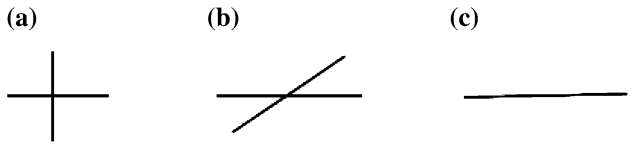

Fig. 1 Three sets of two artificial hash functions generated by maximum entropy of each bit independently

method has a time complexity of $O(BNd^2)$ where B, N and d denote the number of hash bits, the number of images in the database and the number of features of an image. The SCEM-SSH has the same time complexity as the SPLH does [24]. Figure 2a, b show the two artificial hash functions created with and without the orthogonal constraint, respectively. The hash function created without the orthogonal constraint yields a better separation of the four classes of images. This shows that correlated bits yield a better performance in partitioning images in the four semantic classes.

The nonlinear method of the BSPLH (nBSPLH) [25] captures the nonlinear structure of the data. It firstly clusters images into r clusters and uses the cluster centers as anchors to create an anchor graph. Then images are mapped from the input space onto a r-dimensional sparse nonlinear space by finding their s nearest anchors, where r and s are pre-selected constants. This method suffers from an information loss. As an illustration, the nBSPLH generates cluster anchors indicated by squares in Fig. 3a and triangles are new dissimilar images which should be hashed to different codes. However, by the nonlinear mapping of the nBSPLH with $s = 2$, they are hashed to the same hash code because they are symmetric to the two nearest anchors (i.e. symmetric to the green line connecting the two anchors) and yield the same mapped nonlinear feature vector. Therefore, we propose a new nonlinear mapping to resolve this problem. The nonlinear SCEM-SSH (nSCEM-SSH) computes the hash code of a new image based on anchors located within a hypersphere with a given radius. As shown in Fig. 3b, the two images using different number of anchors will have different hash codes. To further make use of the global data distribution, under the nSCEM-SSH scheme anchors are weighted by an inverse document frequency like function according to the density of the region that the anchor is located.

Finally, we propose a bootstrap based multiple hashing for the SCEM-SSH (M-SCEM-SSH). In comparison with the SSH using a single hash table, the M-SCEM-SSH usually yields a higher recall rate while visiting fewer hash buckets.

This work has three contributions:

1. The SCEM-SSH maximizes the joint entropy of all hash bits by sequentially maximizing the conditional entropy of each bit given all its previous bits. This will remove the restrictive bit independent assumption on both the SSH [24] and the BSPLH [25], and makes the SCEM-SSH more suitable for real-world applications. Owing to the fact that variances of data concentrate in a few top principle components, the bit independence assumption in the regularizer of current SSH methods [24, 25] may not be appropriate.
2. The nSCEM-SSH finds the anchors using all anchors located within a local neighborhood instead of s

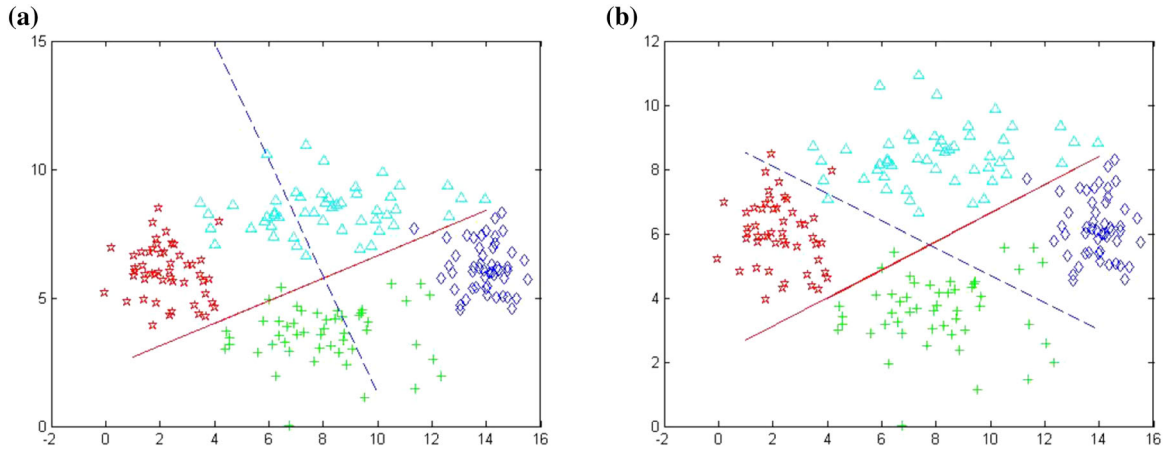

Fig. 2 Two artificial hash functions created **a** with orthogonal constraint and **b** without orthogonal constraint for an artificial image database with four classes of images

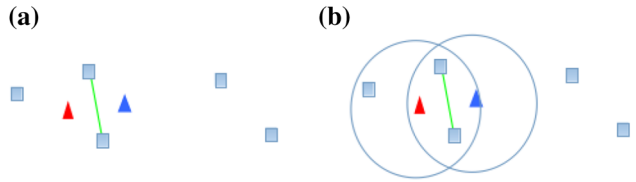

Fig. 3 Illustrations of the nBSPLH (**a**) and the nSCEM-SSH (**b**)

nearest neighbors. The nSCEM-SSH computes the nonlinear mapping by anchors combining both the global and the local information for a better mapping. In contrast, the nBSPLH [24, 25] ignores the data distribution around anchors and maps an image using anchors with the same weight without regarding they are located in a spare or a dense region.

3. A multiple hashing for the SCEM-SSH is proposed to enhance the recall rate and reduce the number of hash bucket visits. It uses less retrieval time for a similar recall rate in comparison to single table based SCEM-SSH methods.

Related works is introduced in Sect. 2. The SCEM-SSH is proposed in Sect. 3.1 while the nSCEM-SSH and the M-SCEM-SSH are proposed in Sects. 3.2 and 3.3, respectively. Section 4 shows and discusses experimental results in comparing the SCEM-SSH with popular SSH methods. We conclude this paper in Sect. 5.

2 Related works

The aim of hashing is to learn a hash code consisting of B hash bits (functions), such that similar images are hashed to the same hash code. A typical bth bit hash function of a hash code is defined as follows:

$$h_b(x) = sign\left(w_b^T x + t_b\right) \quad (1)$$

where x denotes the d-dimensional input vector of an image and both w_b and t_b denote parameters of the hash function (h_b). In Sect. 2.1, we will briefly introduce current supervised and unsupervised hashing methods. SSH is introduced in Sect. 2.2 and multiple hashing is introduced in Sect. 2.3.

2.1 Supervised and unsupervised hashing

The LSH [8–10] is one of the most popular unsupervised hashing methods which generate hash bits by randomly generating parameters of hash functions. The locality preserving principle of the LSH guarantees that the probability of two images being hashed to the same hash code is directly proportional to the similarity between them. There are several variants of LSH being proposed: the l_p norm based LSH [26], the LSH with learned metric for supervised semantic problems [27] and the LSH with image kernel [28]. The major drawback of the LSH is the requirement of a long hash code for a good precision rate in practice [9].

The spectral hashing [11] is proposed to learn compact hash codes. It is an unsupervised data-dependent method which requires hash codes to be uncorrelated and balanced. The spectral hashing assumes data points following a uniform distribution and its hash functions are obtained by the top B smallest single Laplacian eigenfunctions. The anchor graph hashing [12] is a graph based unsupervised hashing method which preserves neighborhood information of images in the Hamming space. The SPICA [13] finds compact hash codes by minimizing the mutual information with relaxed constraints through an independent component analysis based method in an unsupervised manner. The iterative quantization [14] minimizes the average quantization error of images with respect to assigned hash

codes by iteratively rotating the orthogonal projection of hash functions.

However, unsupervised hashing cannot deal with semantic image retrieval problems. Hence, supervised hashing methods are proposed to learn hash codes based on labeled information. The linear discriminant analysis hashing [20] finds hash functions by simultaneously minimizing the expected Hamming distances between similar image pairs and maximizing the expected Hamming distances between dissimilar image pairs. On the other hand, the kernel-based supervised hashing [19] projects data onto a reproducing kernel Hilbert Space by kernel tricks. Then, a hashing is generated in the projected space by minimizing the sum of squared difference between targeted outputs (+1 for similar and −1 for dissimilar image pairs) and normalized inner products of hash codes of image pairs. The restricted Boltzmann machine hashing [21] trains a multilayer perceptron neural network to find a hash code to preserve semantic similarities among images. The minimal loss hashing [22] finds hash functions by minimizing the upper bound of a hinge loss function based on the structural support vector machines with latent variables.

In general, supervised hashing methods yield higher precision rates than that of unsupervised hashing methods. However, supervised hashing methods require a fully labeled database which is infeasible for many large scale image retrieval problems.

2.2 Semi-supervised hashing methods

The SSH [24, 29, 30] is proposed to make use of both the huge number of unlabeled images and the small portion of labeled images to learn hash codes. In general, the objective function of the SSH methods is defined as follows:

$$\max_{W} \left(\frac{1}{2} tr\left(W^T X_l S X_l^T W\right) + \frac{\lambda}{2} tr\left(W^T X X^T W\right) \right) \tag{2}$$

where λ, X, X_l, S and W denote the penalty parameter, the whole set of images, the set of images with labels, the matrix consisting of the pairwise similarity information of image pairs in X_l, and the projection matrix of the hash functions, respectively. The SSH maximizes the empirical similarity accuracy for labeled images ($tr(W^T X_l S X_l^T W)$) and maximizes the lower bound of the maximum variance of all images ($tr(W^T X X^T W)$) as a regularizer to prevent overfitting. In addition to hashing-based retrieval problems, semi-supervised methods are also widely adopted in classification problems [31, 32] and clustering problems [33, 34].

In the SSH [24], three different methods are proposed to learn hash functions. The orthogonal projection learning imposes an orthogonal constraint to force projection directions to be orthogonal to each other. Then, the top B eigenvectors of $M = \left(X_l S X_l^T + \lambda X X^T\right)$ are used to form the W to compute the B-bit hash code. However, in practice, the number of principal components with high variance (information) of a given image database may be smaller than B. In this case, the effectiveness of the SSH is reduced because the orthogonal constraint forces the SSH to add directions with low variances to hashing. Hence, the non-orthogonal projection learning adds an orthogonal penalty to the objective function to replace the hard orthogonal constraint. However, the objective function of the non-orthogonal projection learning is non-convex and only an approximated solution can be found. The semi-supervised spectral hashing also uses orthogonality as a penalty instead of a constraint and it replaces the binary objective function by a squared Euclidean based objective function [35]. Instead of computing all hash functions simultaneously, the SPLH finds hash functions iteratively. In the iteration, the top eigenvector of the current matrix $M_b = \left(X_l S_b X_l^T + \lambda X X^T\right)$ is used as the projection vector w_b for the bth hash bit. In the first iteration, $S_b = S$. In the $(b+1)$th iteration, the matrix S is updated as follows:

$$S_{b+1} = S_b - \tau T\left(\tilde{S}^b, S_b\right) \tag{3}$$

where τ is a constant.

$$\tilde{S}^b = X_l^T w_b w_b^T X_l \tag{4}$$

$$T\left(\tilde{S}_{ij}^b, S_{ij}\right) = \begin{cases} \tilde{S}_{ij}^b & : \quad sign\left(\tilde{S}_{ij}^b S_{ij}\right) < 0 \\ 0 & : \quad sign\left(\tilde{S}_{ij}^b S_{ij}\right) \geq 0 \end{cases} \tag{5}$$

where S_{ij} denotes the (i,j) element of the original pairwise label matrix S. Larger weight values are assigned to image pairs (i.e. the value of the S_{ij}) which violate pairwise similarity labels in the S. The sign of the S_{ij} remains unchanged and only its magnitude is updated according to error made by each bit. At the end of the iteration, the contribution of the spanned subspace of the projection vector of the current hash bit is removed from X to reduce redundancy as follows:

$$X = X - w_b w_b^T X. \tag{6}$$

However, the SPLH [24] only compensates errors made by the previous bits individually without considering them together. This may not be effective in reducing the accumulated error and hence a significantly reduced performance. The BSPLH is proposed to correct errors made by all previous bits. In the $(b+1)$th iteration, the matrix S is updated as follows:

$$S_{b+1} = S + \Delta S^b \tag{7}$$

$$\Delta S_{ij}^{b} = \begin{cases} \left(\alpha b - H_{ij}^{b}\right)/2b & : \quad H_{ij}^{b} - \alpha b < 0,\ S_{ij} > 0 \\ \left(\beta b - H_{ij}^{b}\right)/2b & : \quad H_{ij}^{b} - \beta b > 0,\ S_{ij} < 0 \\ 0 & : \quad otherwise \end{cases} \tag{8}$$

where $H^{b+1} = H^{b} + sign\left(X_l^T w_b w_b^T X_l\right)$, and αb (βb) is the threshold for similarity (dissimilarity). Instead of finding linear hash functions, the nonlinear BSPLH (nBSPLH) [25] finds hash functions in a nonlinearly mapped space $z(x)$:

$$y_b = sign\left(w_b^T z(x) - t_b\right) \tag{9}$$

where $t_b = \sum_{i=1}^{n} w_b^T z(x_i)/n$, and $z(x)$ denotes the column of the regression matrix Z corresponding to the image x. The regression matrix Z of the nBSPLH [25] is computed by the distances between the input vectors and their corresponding s nearest anchors. Firstly, input vectors of images in the database are clustered into r centers, which are used as anchors $U = \{u_i \in R^d\}_i^r$. The (i,j) element of the regression matrix Z is computed as follows:

$$Z_{ij} = \begin{cases} \dfrac{\exp(-\left\|u_i - x_j\right\|^2/\theta)}{\sum_{i' \in <j>} \exp(-\left\|u_{i'} - x_j\right\|^2/\theta)}, & \text{if } i \in \langle j \rangle \\ 0, & \text{otherwise} \end{cases} \tag{10}$$

where $\langle j \rangle \subset [1{:}r]$ and θ denote the index value of the s nearest anchors in U for the image x_j and the bandwidth parameter, respectively. The relaxed objective function by removing the sign function is defined as follows:

$$\max_{W} \frac{1}{2} tr\{W^T Z_l S Z_l^T W\} + \frac{\lambda}{2} tr\{W^T Z Z^T W\} \tag{11}$$

where Z_l and Z denote regression matrices for the labeled images and all images, computed by Eq. (10), respectively. Then, the hash functions are computed in the same way as the BSPLH does.

2.3 Multiple hashing methods

Instead of using a single hash table (code), multiple hash tables yield better recall rates while requiring fewer hash bucket visits in general. A multiple hashing returns the union of images returned by the hash bucket with a small Hamming distance from the query image from each hash table. The LSH [9, 10, 26] can be extended to multiple hashing easily by repeating the random generation hash code for K times.

The complementary hashing (CH) [36] learns a hash table by compensating the error made by each previous table individually to construct multiple hash tables. The CH learns a hash table by maximizing Eq. (2) and finding the top B eigenvectors of M in the same way as the orthogonal SSH. Then, the matrix S is updated according to the error made by the currently learned hash table. The dual complementary hashing (DCH) [37] uses the SPLH to replace the orthogonal SSH method used in the CH. Hence, a single hash bit of an individual hash table is learned in the same way as the SPLH. Moreover, the DCH learns each hash table using boosting to further enhance its performance. However, the DCH deletes the pairwise labels from the labeled set when the similarity of two corresponding images is classified correctly which may introduce new error. A boosting based ITQ framework [38] is proposed to construct multiple hash tables. Samples returned from multiple tables are re-ranked by bit-level weights learned by a query-adaptive re-ranking method.

3 Semi-supervised hashing using sequential conditional maximization entropy

Throughout this manuscript, the SCEM-SSH will be referred to the linear version of the SCEM-SSH and the nSCEM-SSH as its nonlinear version. The SCEM-SSH and the nSCEM-SSH are proposed in Sects. 3.1 and 3.2, respectively. In Sect. 3.3, we propose the M-SCEM-SSH. The time complexities of the proposed methods are discussed in Sect. 3.4.

3.1 The SCEM-SSH

The objective function of the SSH based methods consists of two components: an empirical error $tr\left(W^T X_l S X_l^T W\right)$ and a regularizer ($tr(W^T X X^T W)$). In [24, 25, 29, 30], the regularizer $tr(W^T X X^T W)$ of the objective function [as shown in Eq. (2)] ignores interactions among hash bits and maximizes the lower bound of the maximum variance of each hash bit only. Hence, we propose a new regularizer in the SCEM-SSH which generates the bth hash bit at the steepest increasing direction of the entropy for all b bits. Firstly, the conditional entropy of the bth hash bit ($h_b(x)$) given all previous bits ($h_1(x), h_2(x), \ldots, h_{b-1}(x)$) is defined as follows:

$$H(h_b(x)|h_1(x), h_2(x), \ldots, h_{b-1}(x)) = \sum_{i=1}^{N_b} p_i \xi_i \tag{12}$$

where p_i denotes the probability of images in the database falling into the ith bucket (U_i) created by all previous hash bits, $\xi_i = -(q_i \log(q_i) + (1 - q_i)\log(1 - q_i))$, $q_i = P(h_b(x) = 1 | x \text{ falls into } U_i)$, and $(1 - q_i) = P(h_b(x) = -1 | x \text{ falls into } U_i)$. For a hash table with b bits, the input space consisting of all images is cut into at most 2^b buckets by b hash functions. Let $N_b = \min\left(N, 2^{b-1}\right)$ be the number of non-empty hash buckets at the bth iteration and N be the number of images in the database. It is obvious that ξ_i is a

concave function reaching its maximum value when the partition of the U_i is balanced. Then, we derive the objective function of the SCEM-SSH following the steps listed in [24].

The bth hash bit with maximum conditional entropy maximizes the conditional variance and vice versa. Let $h_{bi}(x)$ be the value of the bth hash bit given that the image x falling into the U_i. Then, we have:

$$\max\ \xi_i \Leftrightarrow \max\ Var(h_{bi}(x)). \tag{13}$$

Then, we can extend it to all hash buckets easily as follows:

$$\max\ \sum_{i=1}^{N_b} p_i \xi_i \Leftrightarrow \max\ \sum_{i=1}^{N_b} p_i Var(h_{bi}(x)). \tag{14}$$

The conditional expectation and the conditional variance of values of the bth hash bit given that images x fall into the ith bucket (h_{bi}) are as follows:

$$E(h_{bi}(x)) = 2q_i - 1 \tag{15}$$

$$\begin{aligned} Var(h_{bi}(x)) &= E\Big((h_{bi}(x) - (2q_i - 1))^2\Big) \\ &= 4q_i(1 - q_i). \end{aligned} \tag{16}$$

The $Var(h_{bi}(x))$ is a concave function which yields the maximum value 1 when images falling into the U_i created by all previous hash bits is partitioned evenly by the bth hash function (i.e. $q_i = \frac{1}{2}$).

Then, we define the new conditional entropy maximizing regularizer as follows:

$$\begin{aligned} R(w_b) &= \sum_{i=1}^{N_b} (P_{bi}) Var(h_{bi}(x)) \\ &= \sum_{i=1}^{N_b} (P_{bi}) Var\big(sign\big(w_b^T x\big) | x \in U_i\big) \end{aligned} \tag{17}$$

where P_{bi} denotes the probability of the image x falling into the bucket U_i created by all previous hash functions. We can approximate P_{bi} by (n_i/N) where n_i denotes the number of images falling into the bucket U_i. The approximated $R(w_b)$ is computed by the following equation:

$$\breve{R}(w_b) = \sum_{i=1}^{N_b} \frac{n_i}{N} Var\big(sign\big(w_b^T x\big) | x \in U_i\big) \tag{18}$$

Equation (18) is non-differentiable with respect to w_b, so we find its lower bound instead. The conditional variance of a hash bucket is bounded by Eq. (19).

$$\begin{aligned} Var(sign(h_{bi}(x))) \geq \frac{1}{\varepsilon} E\Big(\big\|w_b^T x\big\|^2 | x \in U_i\Big) \\ - \big(E\big(sign\big(w_b^T x\big) | x \in U_i\big)\big)^2. \end{aligned} \tag{19}$$

Let $\|x_i\|^2 \leq \varepsilon\ \forall i, \varepsilon > 0$. Given that $\|w_b\|^2 = 1\ \forall b$, by the Cauchy–Schwarz inequality, we have:

$$\big\|w_b^T x\big\|^2 \leq \|w_b\|^2 \|x\|^2 \leq \varepsilon = \varepsilon \big\|sign\big(w_b^T x\big)\big\|^2. \tag{20}$$

Hence, we have:

$$\begin{aligned} E\Big(\|sign(h_{bi}(x))\|^2\Big) &= E\Big(\big\|sign\big(w_b^T x\big)\big\|^2 | x \in U_i\Big) \\ &\geq \frac{1}{\varepsilon} E\Big(\big\|w_b^T x\big\|^2 | x \in U_i\Big). \end{aligned} \tag{21}$$

Since $E\Big(\big\|sign\big(w_b^T x\big)\big\|^2 | x \in U_i\Big) = Var(sign(h_{bi}(x))) + \big(E\big(sign\big(w_b^T x\big) | x \in U_i\big)\big)^2$, we get Eq. (19).

With the relaxation given in [24], the $E(sign\ \big(w_b^T x\big) | x \in U_i)$ is replaced by $E\big(w_b^T x | x \in U_i\big)$. Then the regularizer in Eq. (18) can be computed as follows:

$$\begin{aligned} &\breve{R}(w_b) \\ &= \sum_{i=1}^{N_b} \frac{n_i}{N} \left(\frac{1}{\varepsilon} E\Big(\big\|w_b^T x\big\|^2 | x \in U_i\Big) - \big(E\big(w_b^T x | x \in U_i\big)\big)^2\right) \\ &= \sum_{i=1}^{N_b} \frac{n_i}{N} \left(\left(\frac{1}{\varepsilon n_i} \sum_{x_j \in U_i} w_b^T x_j x_j^T w_b\right) - w_b^T \mu_i \mu_i^T w_b\right) \\ &= \frac{1}{\varepsilon N} \left(\sum_{j=1}^{N} w_b^T x_j x_j^T w_b - \sum_{i=1}^{N_b} \varepsilon n_i w_b^T \mu_i \mu_i^T w_b\right) \\ &= \frac{1}{\varepsilon N} w_b^T \big(XX^T - \varepsilon V \Omega^T\big) w_b \end{aligned} \tag{22}$$

where μ_i, Ω and V denote the average value of the input vectors of x in U_i, the matrix with the ith column equal to μ_i and the matrix with the ith column equal to $n_i \mu_i$, respectively.

Then, the SCEM-SSH sequentially finds all B hash bits by maximizing the following objective function:

$$\max_{w_b}\ J(w_b) = w_b^T \big(X_l S X_l^T + \eta XX^T - \lambda V \Omega^T\big) w_b \tag{23}$$

where $\eta = 1/(\varepsilon N)$ and $\lambda = 1/N$. The w_b maximizing the $J(w_b)$ is computed by the top eigenvector of $\breve{M} = X_l S X_l + \eta XX^T - \lambda V \Omega^T$. Same as the SSH [24], $w_b^T X_l S X_l^T w_b$ maximizes the accuracy of the empirical similarity for labeled images while $w_b^T XX^T w_b$ maximizes the variance of hash values to prevent overfitting. The new term in Eq. (23), i.e. $w_b^T V \Omega^T w_b$, gives a higher priority to partition buckets (created by all previous bits) with larger n_i (i.e. number of images in the bucket) because a bucket with a larger n_i yields a larger influence to Eq. (23). This maximizes the conditional entropy of the bth bit with respect to all previous bits. Hence the SCEM-SSH sequentially maximizes the joint entropy of all hash bits. Figure 4 illustrates the partitions of the 3rd hash bits given two previously trained hash bits (indicated by the blue dotted lines) generated with and without maximizing conditional entropy, respectively. The 1st and the 2nd hash functions are represented by blue lines. Images labeled to be similar

are connected by green dotted lines while images labeled to be dissimilar are connected by pink dotted lines. As shown in Fig. 4a, the 3rd hash bit maximizing the conditional entropy of all previous bits cuts through the buckets consisting of a large number of images. Hence, the SCEM-SSH yields a better coding efficiency of hashing. Procedures of creating a hash table using the SCEM-SSH is shown in Algorithm 1.

Algorithm 1 SCEM-SSH

Inputs: images X, pairwise labeled images X_l, initial pairwise labels $S^1 = S$, length of hash code B, constants η, λ,

Outputs: projection matrix W, H^b;

Initialize $H^0 = 0$, $C = XX^T$, $Y = C$, $h = []$, $W = []$

for b = 1 to B **do**

$\breve{M}^b = X_l S^k X_i^T + \eta Y$

Set w_b equal to the top eigenvector of $\breve{M}^b$

$H^b = H^{b-1} + sign(X_l^T w_b w_b^T X_l)$

Calculate the ΔS^b according Equation (8)

Set $S^{b+1} = S^b + \Delta S^b$ according to Equation (7)

$h = [h, sign(X^T w_b)]$

$W = [W, w_b]$

Use hash table h to get bucket ID for each image

Compute V,Ω

Set $Y = C - (\lambda/\eta)V\Omega$

end for

return W, H^b

3.2 Nonlinear SCEM-SSH

The nSCEM-SSH computes the nonlinear mapping based on both global and local information of images instead of local information only [25]. For local information, instead of computing the nonlinear mapping by s nearest anchors as in [25], the nSCEM-SSH computes the nonlinear mapping of an image by all anchors located within a Q-ball centered at the image, where Q is a user defined constant. This will allow the proposed method to preserve local similarity within a Q distance. For instance, images located in a dense region of the input space will use more anchors to provide more precise nonlinear mapping, while images located in a sparse region will use less anchors. In contrast, the nonlinear mapping used by the nBSPLH [25] computes the nonlinear mapping using s anchors which may be far away from images located in a sparse region. This is misleading and counterproductive to the retrieval result. Anchors are created using a k-means clustering on all training images with the number of anchors selected as in [25].

The value of Q is computed by the average value of distances between images and their sth nearest neighbors over the database. A heuristic method to select the Q value is to try out different numbers of nearest neighbors (i.e. s). Then the Q value yielding the best performance will be selected. If two consecutive s values yield similarly good performances, the Q values of them are averaged to generate a new Q value. In our experiments this new Q value yields the best performance. At the end of Sect. 4.1, we will illustrate this simple heuristic using the MNIST database.

The global information of the database is captured by a function similar to the inverse document frequency. In contrast to treating each anchor with equal weight [25], each anchor is weighted by its inverse frequency such that anchors located within Q-balls of more images (i.e. frequently used by images in the database) weight less. If this image is located at a dense region, its Q-ball consists of a large number of anchors and the norm of its feature vector of the nonlinear mapping will be large. In contrast, the norm of the feature vector of the image located in a sparse region is small and will be easily ignored during the optimization if all anchors are weighted the same. Therefore, anchors located in dense regions are assigned with smaller weight values to balance the importance of feature vectors of images in regions with different image densities. Moreover, anchors in sparse regions provide more discriminative information because they have higher probabilities to be the representative of clusters consisting images in the same semantic class.

The nSCEM-SSH computes the set of r anchors $U = \{u_i \in R^d\}_i^r$ by a k-means clustering with ten iterations in the same way as in [25]. Then, a similarity matrix between r anchors and N images is defined as follows:

$$\kappa_{ij} = \begin{cases} \exp(-D^2(u_i, x_j)/\theta), & \text{if } i \in \langle j \rangle \\ 0, & \text{otherwise} \end{cases} \tag{24}$$

where D and $\langle j \rangle$ denote a l_2 distance function and the index set of anchors located within the Q-ball of the image x_j, respectively. If the number of anchors in $\langle j \rangle$ is less than 1, the nearest anchor is added to the $\langle j \rangle$ to avoid degeneracy. Then, the inverse frequency based weight function is defined as follows:

$$\phi_i = \log_2\left(\frac{S}{S_i}\right) \tag{25}$$

where $S = \sum_i S_i$ and $S_i = \sum_j \kappa_{ij}$. S serves as a normalization constant. We use an anchor book (U) to map the original images (X) onto a nonlinear feature space (Z') as below:

$$Z'_{ij} = \begin{cases} \phi_i \exp(-D^2(u_i, x_j)/\theta), & \text{if } i \in \langle j \rangle \\ 0, & \text{otherwise.} \end{cases} \tag{26}$$

Then, the SCEM-SSH described in the previous section is applied to the nonlinear mapped feature domain Z' instead of the original feature domain X to find hash

functions for the nSCEM-SSH. The algorithm of the nSCEM-SSH is shown in Algorithm 2.

Algorithm 2 nSCEM-SSH

Inputs: images X, pairwise labeled images X_l, initial pairwise labels $S^1 = S$, length of hash code B, constants η,λ, Q, s

Outputs: projection matrix W and anchor book U.

Perform clustering to get r anchors $U = \{u_i \in R^d\}_i^r$

Compute the similarity matrix κ by Equation (24)

Compute ϕ_i for each anchor by Equation (25)

Compute the regression matrix Z' by Equation (26)

Perform SCEM-SSH(Algorithm 1) on Z' to compute the projection matrix W

return W, U

3.3 Multiple hashing SCEM-SSH

The M-SCEM-SSH constructs K hash tables in a bootstrap manner. The major difference between the DCH [37] and the M-SCEM-SSH is that the M-SCEM-SSH constructs each hash table to correct errors made by all previous tables holistically. Multiple hashing returns the union of images yielding similar hash code of the query image in at least one hash table, so the minimum Hamming distance (m) between a pair of images (x_i, x_j) among all hash tables determines the precision of the returned result. If the value m between a pair of dissimilar images is too small, the precision of the multiple hashing decreases. On the other hand, the recall rate of the multiple hashing decreases if m is too large for a pair of similar images. Therefore, in the M-SCEM-SSH, Eq. (27) updates the pairwise label matrix after every bit of individual hash table being learned by using the SCEM-SSH.

$$\Delta S_{ij}^k = \begin{cases} \left(\alpha B - G_{ij}^k\right)/2B & : \quad G_{ij}^k - \alpha B < 0,\ S_{ij}^1 > 0 \\ \left(\beta B - G_{ij}^k\right)/2B & : \quad G_{ij}^k - \beta B > 0, S_{ij}^1 < 0 \\ 0 & : \quad \text{otherwise} \end{cases} \tag{27}$$

where k, K, B, G_{ij}^k denote the current number of hash table under construction out of K tables, the number of hash tables, the number of bits in an individual hash table and the maximum value of inner products between images (x_i, x_j) among k hash tables, respectively. Note that $D(x_i, x_j) = (B - < h(x_i), h(x_j) >)/2$ [19], so the maximum value of G_{ij}^k is equivalent to the minimum Hamming distance between images x_i and x_j among the k hash tables.

Algorithm 3 shows the procedures of the M-SCEM-SSH. It is easy to extend the proposed method to a nonlinear multiple hashing by replacing the linear SCEM-SSH step by the nSCEM-SSH. The nonlinear multiple hashing version of our method is denoted by the M-nSCEM-SSH.

3.4 Time complexities of different SCEM-SSH versions

The time complexity of the linear SCEM-SSH is $O(BNd^2 + BdL^2)$ where B, N, L and d denote the total number of hash bits in a table, the total number of both labeled and unlabeled images, the number of the labeled images and the number of input dimensions of data, respectively. When L is small, the time complexity of the SCEM-SSH is $O(BNd^2)$. For the iteration, the time complexities of computing XX^T, $X_l S X_l^T$, VU^T and the eigenvector of the $\breve{M}$ are $O(Nd^2)$, $O(dL^2)$, $O(Nd^2)$ and $O(d^3)$, respectively. The overall computation of one hash bit of the SCEM-SSH is $O(Nd^2 + dL^2 + d^3)$. Owing to the fact that $N \gg d$, the time complexity of one iteration is $O(Nd^2 + dL^2)$. Therefore, the total time complexity of the SCEM-SSH with B bits is $O(BNd^2 + BdL^2)$. Note that buckets with no image will not participate in the computation of Eq. (23) and the number of non-empty hash bucket is at most equal to the number of images (N) in the database. The sizes of both matrices V and U do not exceed $d \times N$. Hence, the total time complexity of computing VU^T is $O(Nd^2)$. For an iteration, the time complexity of computing XX^T is also $O(Nd^2)$.

The time complexity for the nSCEM-SSH is $O(aNrd + BNr^2 + BrL^2)$, where a denotes the number of iterations of the k-means clustering algorithm to compute r anchors. As in [25], $a = 10$ is usually good enough. If $r < d$, the complexity of the nSCEM-SSH is smaller than that of the linear SCEM-SSH. The time complexities of M-SCEM-SSH and M-nSCEM-SSH with K hash tables are $O(K(BNd^2 + BdL^2))$ and $O(pNrd + K(BNr^2 + BrL^2))$, respectively.

The time complexity of querying using linear and nonlinear versions of SCEM-SSH are $O(Bd)$ and $O((B + r)d)$, respectively. As K is a small constant in comparison to B and d, the time complexities of M-SCEM-SSH is the same as the SCEM-SSH.

Algorithm 3 M-SCEM-SSH

Inputs: data X, pairwise labeled images X_l, initial pairwise labels $S^1 = S$, codes length of single hash table B, constants ξ, number of hash tables K

Outputs: projection matrix W;

Initialize maximal inner product matrix $G^0 = -\infty$, W=[]

for k = 1 to K **do**

Perform SCEM-SSH to learn an individual hash table to get W_k, H

$W = [W, W_k]$

$G^k = max(H, G^{k-1})$

Get ΔS^k using Equation (27).

Set $S^{k+1} = S^l + \xi \Delta S^k$

end for

return W

4 Experiments

In this section, the four versions of the SCEM-SSH are compared with popular hashing methods. Both the SCEM-SSH and the nSCEM-SSH are compared with the SPLH [24] and the nBSPLH [25] in Sect. 4.1. The M-SCEM-SSH and the M-nSCEM-SSH are compared with the CH [36] and the DCH [37] in Sect. 4.2. Experiments are carried out to compare their precision–recall curves using four benchmarking databases: the MNIST, the ISOLET, the USPS and the Caltech101.

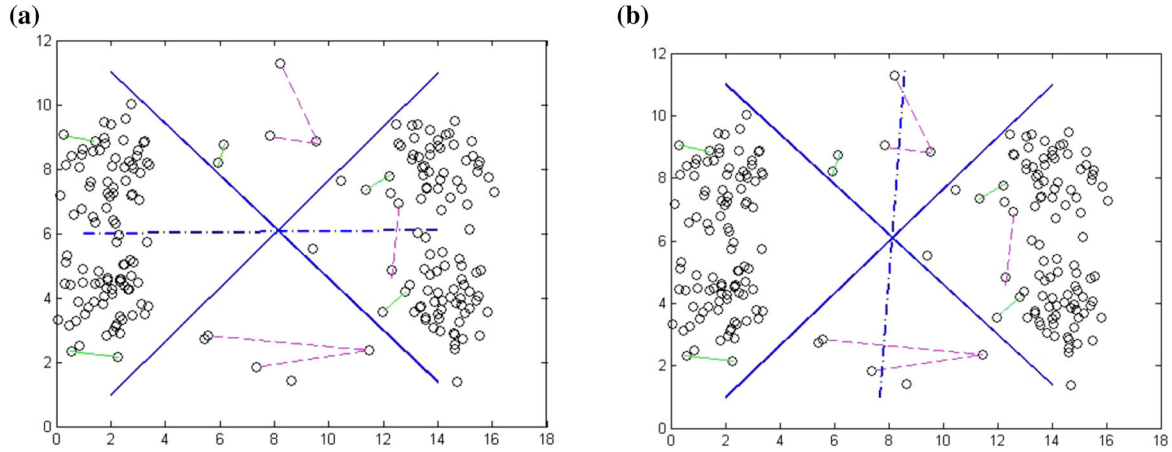

Fig. 4 Partitions of the 3rd hash bit (*blue dotted line*) with (**a**) and without (**b**) maximizing conditional entropy (color figure online)

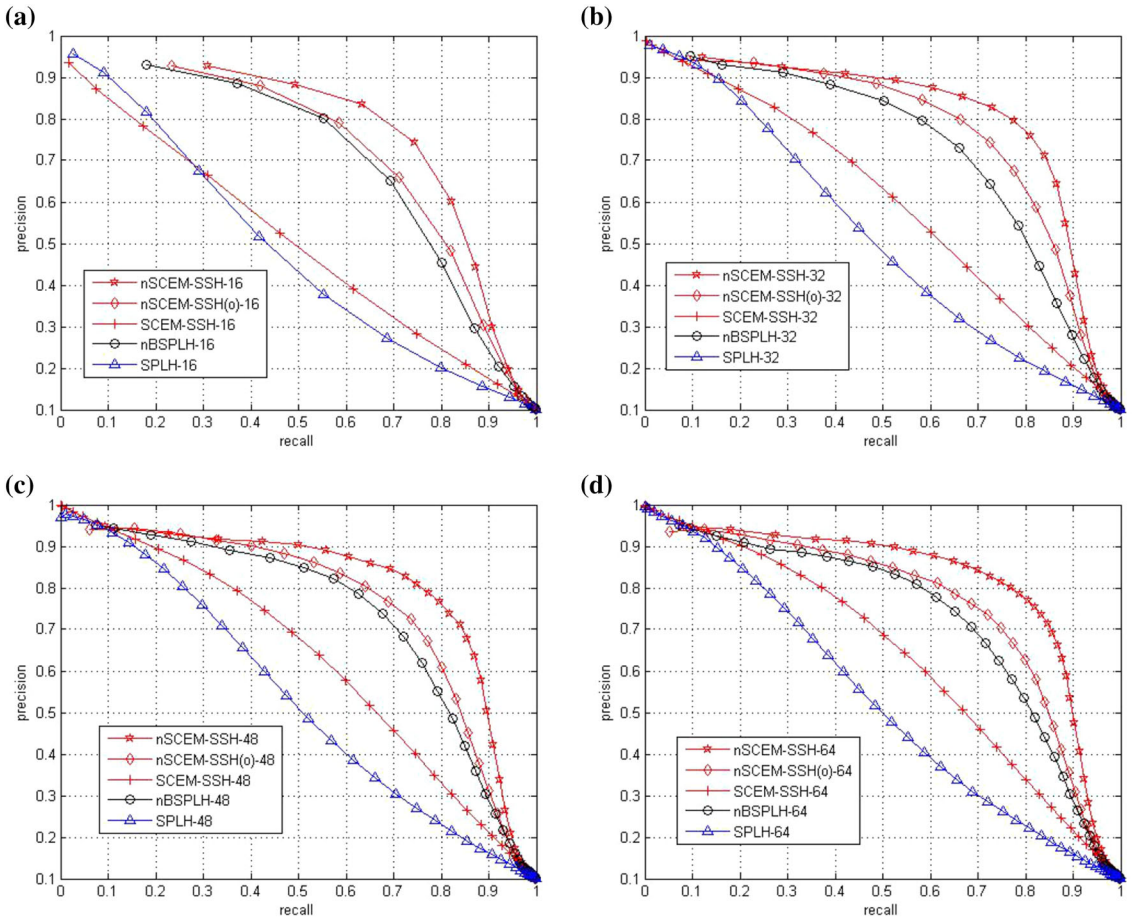

Fig. 5 Precision–recall curves of different hashing methods on MNIST database with 16 bits (**a**), 32 bits (**b**), 48 bits (**c**) and 64 bits (**d**)

The MNIST handwritten digit database consists of 70 K samples in 10 classes with 784 features. The ISOLET spoken letter recognition database consists of 7797 samples distributed in 26 classes with 617 features. For the USPS handwritten digit database, we follow [25] to select a subset of 9298 16×16 handwritten images. For these three databases, 1 K images are randomly selected to serve as queries while another 1 K images are randomly selected as labeled images. Remaining images serve as unlabeled images. The Caltech101 database [39] consists of 9144 images of objects associated with 101 categories. Each category consists of 40–800 images, and most categories consist of about 50 images. The 1024 dimensional bag-of-word feature set [40] is adopted to describe images of the Caltech101 database. To avoid the imbalanced data problem among categories, 30 images from each category are randomly selected as labeled data. The remaining images are randomly split into 1 K queries and 5114 unlabeled data as in [25]. Training of hashing for all methods is performed using both the labeled and the unlabeled image sets. Parameters of all methods (including the proposed methods) are set in the same way as in [24, 25], and the number of anchors (r) is equal to 300 as in [25]. We will further investigate the effect of the number of anchors to the proposed nonlinear mapping method at the end of Sect. 4.1.

4.1 Experimental results of linear and nonlinear SCEM-SSH

Figures 5, 6, 7 and 8 show precision–recall curves yielded by different methods for the MNIST, the ISOLET, the USPS and the Caltech101, respectively. In these figures, nSCEM-SSH(o) denotes the nSCEM-SSH using the nonlinear mapping method of nBSPLH [25] instead of our proposed method. The comparison between the nSCEM-SSH(o) and the nSCEM-SSH shows the usefulness of the proposed nonlinear mapping method combining both local and global information. In this set of experiments, 16-, 32-, 48- and 64-bit hash tables are built to show the trend of performance of different methods using different number of hash bits in sub-figures (a), (b), (c) and (d), respectively, in Figs. 5, 6, 7 and 8 for the four databases.

In experiments of the MNIST, the ISOLET and the USPS, the nSCEM-SSH yields the best precision–recall

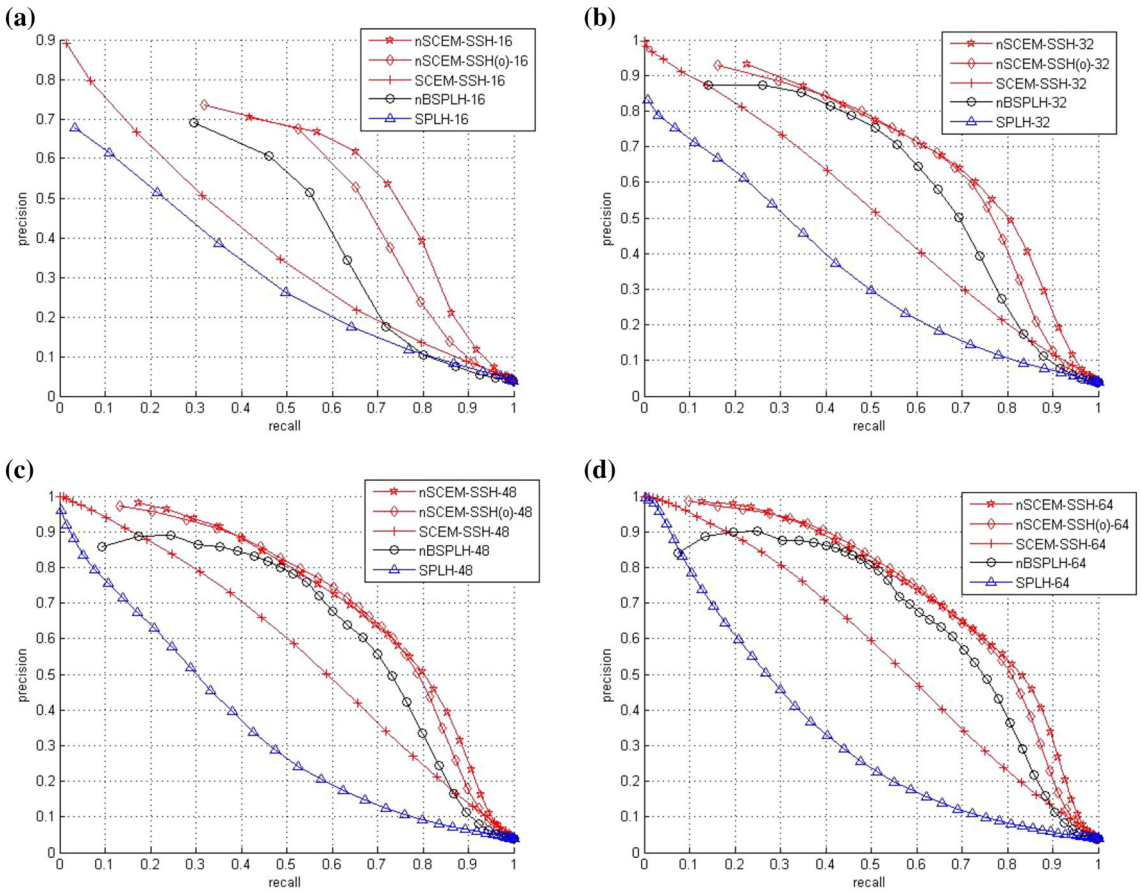

Fig. 6 Precision–recall curves of different hashing methods on ISOLET database with 16 bits (**a**), 32 bits (**b**), 48 bits (**c**) and 64 bits (**d**)

curves and nonlinear methods always outperform the linear ones. This shows that the proposed nonlinear mapping method is more effective in comparison to the nonlinear mapping method of the nBSPLH [i.e. both the nBSPLH and the nSCEM-SSH(o)]. Both the nSCEM-SSH and the nSCEM-SSH(o) outperform the popular nBSPLH while the SCEM-SSH outperforms the SPLH. When the number of hash bits increases, the SCEM-SSH yields larger increment of both precision and recall rates in comparison to those of the SPLH. These show that the maximization of conditional entropy among all hash bits yields better precision and recall rates. As aforementioned, variances in databases usually concentrate in the top few principle components. Therefore, the bit independence assumption in both the SPLH and the nBSPLH degrades their performances when the number of hash bits increases. This is not obvious in the nonlinear cases because images are mapped onto a nonlinear space before hashing.

In experiments of the Caltech101, nonlinear methods perform worse than the linear ones. When the number of hash bits increases, the SCEM-SSH, the SPLH and the BSPLH yield better performance while the nSCEM-SSH maintains a poor performance. Hence, we further investigate the four databases and find that the problem lies on the Euclidean based l_2 metric which is used to compute the nonlinear mapping in both the nBSPLH and the nSCEM-SSH. When the metric does not reflect the semantic similarity well, the nonlinear methods perform worse than the linear methods. The Caltech101 database consists of 101 classes of images which are very difficult to distinguish using the Euclidean metric. Table 1 shows the nearest neighbor (NN) classification results on images in query sets. The NN classifier performs well for the MNIST, the ISOLET and the USPS databases, but it drops to 41.02 % of accuracy for the Caltech101 database. This shows that the Euclidean metric does not preserve a good semantic similarity for the Caltech101. Table 2 shows the maximum and the average numbers of classes of images in $\lceil N/r \rceil$ nearest neighbors of anchors. On average, images located near anchors for the MNIST, the ISOLET and the USPS belong to 2.69, 2.50 and 1.95 classes, respectively. These numbers are reasonably small. Therefore, the feature

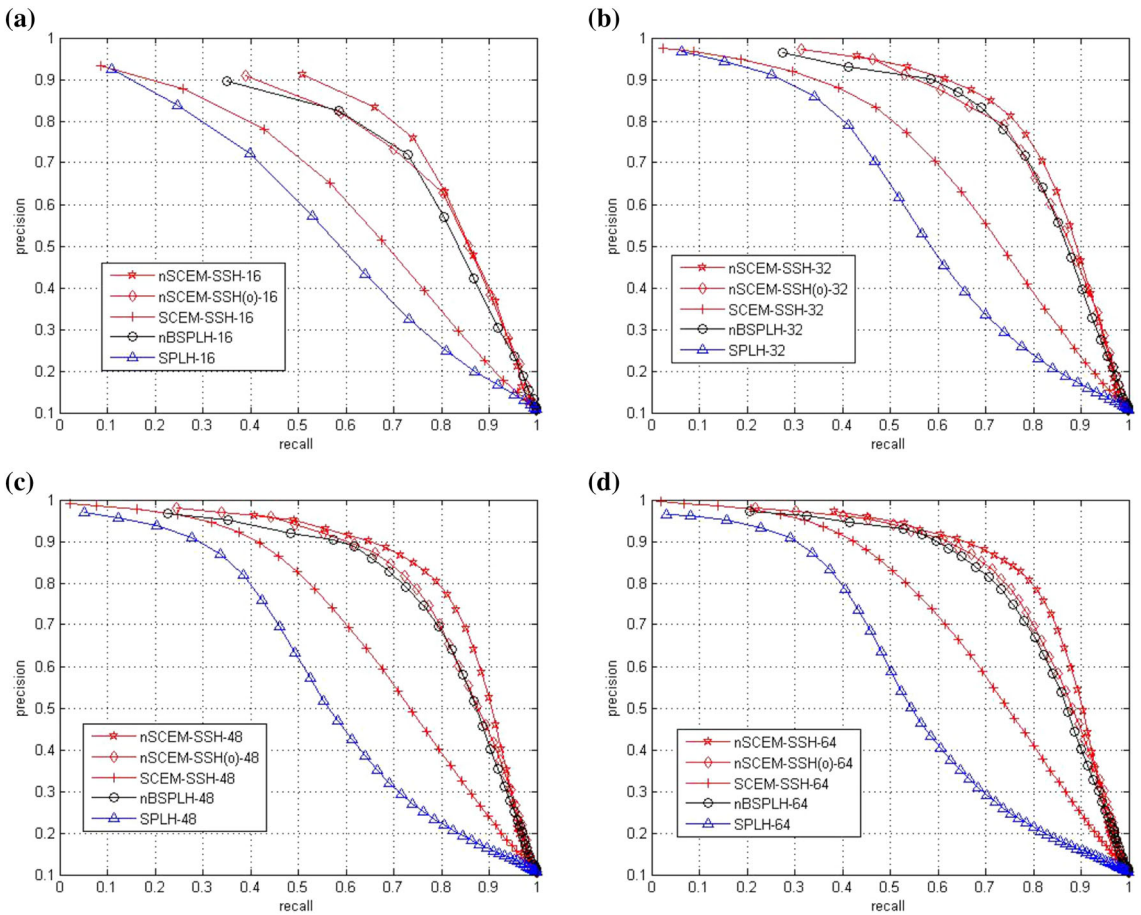

Fig. 7 Precision–recall curves of different hashing methods on USPS database with 16 bits (**a**), 32 bits (**b**), 48 bits (**c**) and 64 bits (**d**)

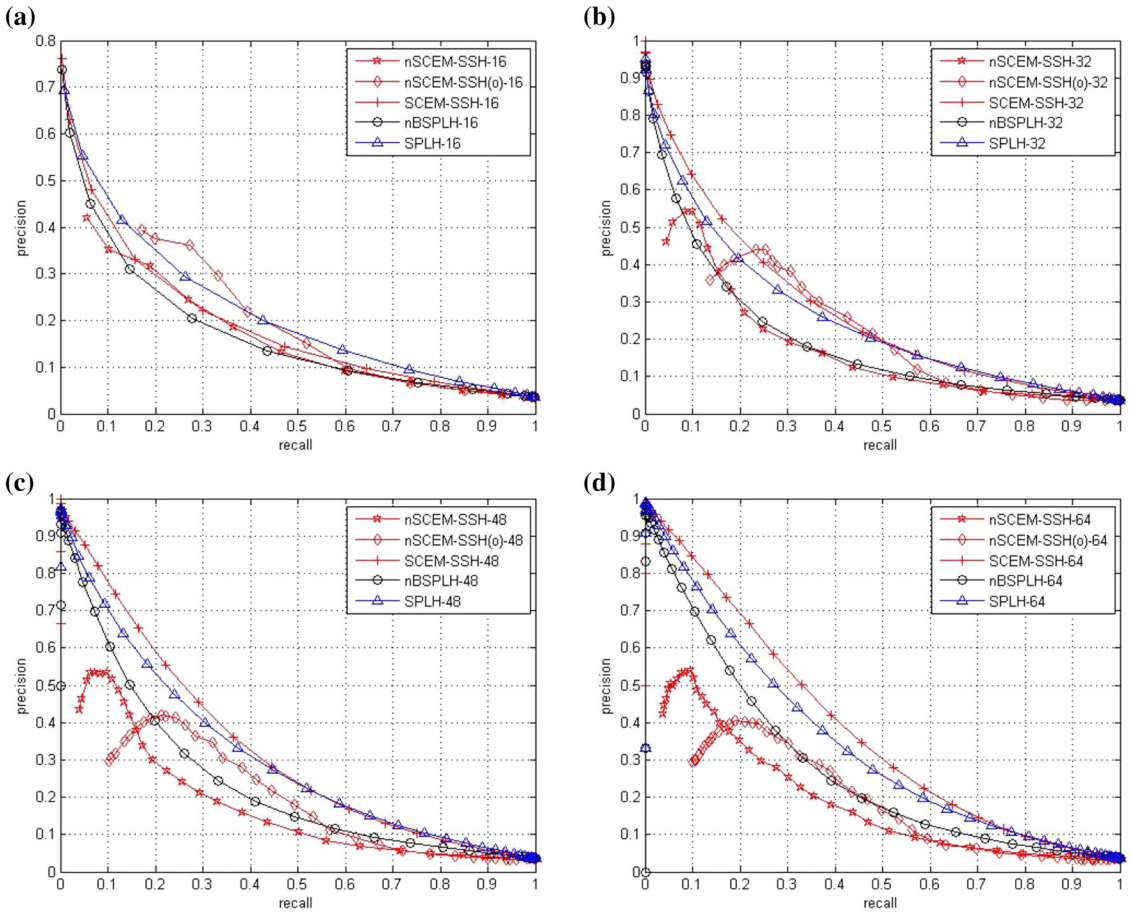

Fig. 8 Precision–recall curves of different hashing methods on Caltech101 database with 16 bits (**a**), 32 bits (**b**), 48 bits (**c**) and 64 bits (**d**)

vectors of mapped images in these three databases preserve their semantic similarity well. In contrast, for the Caltech101 database, images located near anchors belong to 8.39 classes on average and 23 classes in maximum. Anchors for the Caltech101 database fail to distinguish images in different classes and therefore the nonlinear mapping does not preserve semantic similarity well. This leads to the failure of nonlinear hashing methods for the Caltech101 database. Based on experimental results in this section, we propose the following heuristic guideline for the selection of hashing methods:

1. When either the number of classes is large or the NN classification result is poor, the linear SCEM-SSH is suggested.
2. Otherwise, the nonlinear nSCEM-SSH is suggested.

This is the first guideline proposed for SSH methods to select between linear and nonlinear hashing based on characteristics of a given database.

The query times after the encoding for all methods are the same when the number of hash bits is the same. Therefore, only the average encoding time for the query image is shown in Table 3 to compare the query speed of different methods. Nonlinear hashing methods take longer time to encode a query image in comparison to linear methods. Overall, Table 3 shows that both the SCEM-SSH and the nSCEM-SSH are very fast and take less than 1ms to encode a query image.

Table 1 Nearest neighbor classification accuracies among the four databases

MNIST (%)	ISOLET (%)	USPS (%)	Caltech101 (%)
97.21	85.19	97.33	41.02

Table 2 The maximum and the average number of classes of images of $\lceil N/r \rceil$ nearest neighbors of anchors for different databases

	MNIST	ISOLET	USPS	Caltech101
Max	10	11	8	23
Mean	2.69	2.50	1.95	8.39

Table 3 Average encoding time of different methods for the MNIST database with different number of bits

	16-bit (ms)	32-bit (ms)	48-bit (ms)	64-bit (ms)
nSCEM-SSH	0.7632	0.7637	0.7649	0.7667
nSCEM-SSH(o)	0.7632	0.7636	0.7649	0.7663
SCEM-SSH	5.454×10^{-3}	7.503×10^{-3}	1.254×10^{-2}	2.546×10^{-2}
nBSPLH	0.7776	0.7855	0.7875	0.7915
SPLH	2.308×10^{-3}	2.921×10^{-3}	3.879×10^{-3}	4.878×10^{-3}

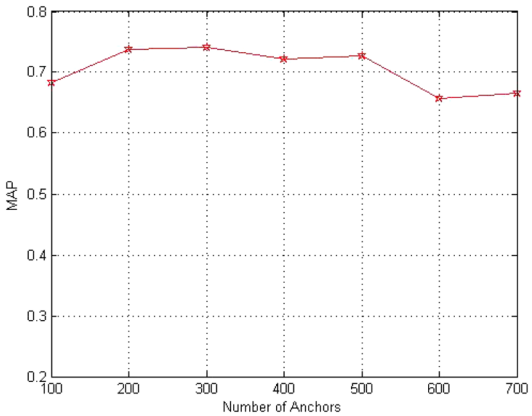

Fig. 9 Number of anchors selection on MNIST using 16 bits

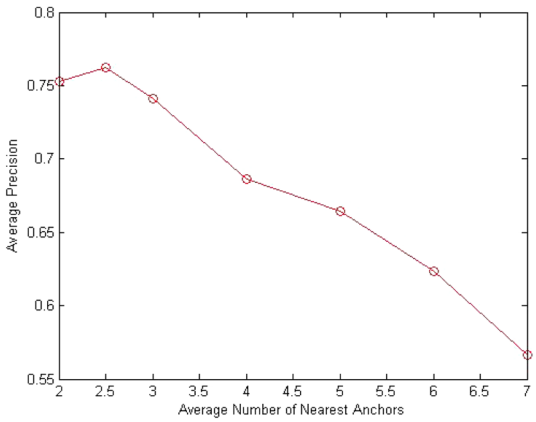

Fig. 11 Q value selection of MNIST database

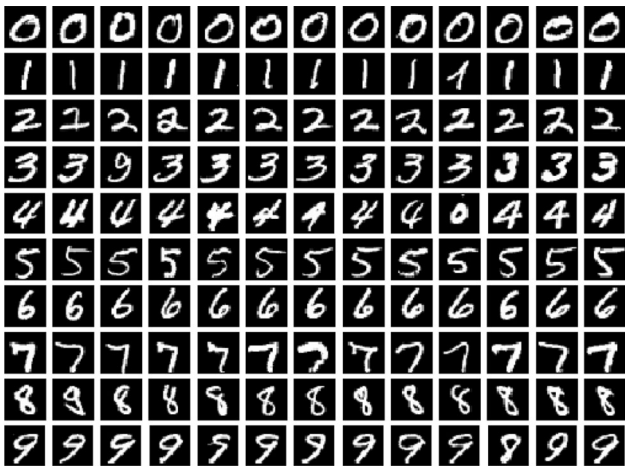

Fig. 10 Images retrieved by the nSCEM-SSH with 64 bits for the MNIST database

Similar to [25], we perform experiments on parameters selection using the MNIST database. Figure 9 shows the MAP of different numbers of anchors using the 16-bit nSCEM-SSH. The 16-bit nSCEM-SSH with 300 anchors yields the best MAP. Moreover, it seems that the number of anchors within the range between 200 and 500 has no significant effect on the performance of the nSCEM-SSH.

Figure 10 shows the retrieval results of 10 queries for the MNIST database. For each row in Fig. 10, the first image is the query image while others are images retrieved by the nSCEM-SSH with 64 bits. The first retrieved images for classes "2" and "8" are very ambiguous and look very similar to "1" and "9", respectively. Such cases cannot be accurately recognized even by human easily.

Finally, the nSCEM-SSH uses a new nonlinear mapping method which uses all anchors located within a Q-ball instead of a pre-selected number of anchors in the nBSPLSH. In the experiments, the average distance between images in the database and their corresponding sth nearest anchors is used as the value of Q. Essentially we convert the problem of selecting the Q value into a problem of selecting the s value for better use of the distribution information of the given database. Figure 11 shows the average precisions of the nSCEM-SSH yielded by Q values computed using different values of s. Overall, a larger s yields a larger Q value because the average distance between images and their sth nearest anchors increases. Figure 11 shows that a large Q value yields poor precision because an over large Q-ball could compute the nonlinear mapping based on irrelevant anchors. Both $s = 2$ and $s = 3$ yield high average precisions. Hence, we compute a new Q value by averaging these two Q values and denote it as $s = 2.5$ or the distance between images and their "2.5th nearest anchors". Figure 11 shows that this Q value yields the best average precision. Readers could follow these steps to find the optimal Q value for different databases.

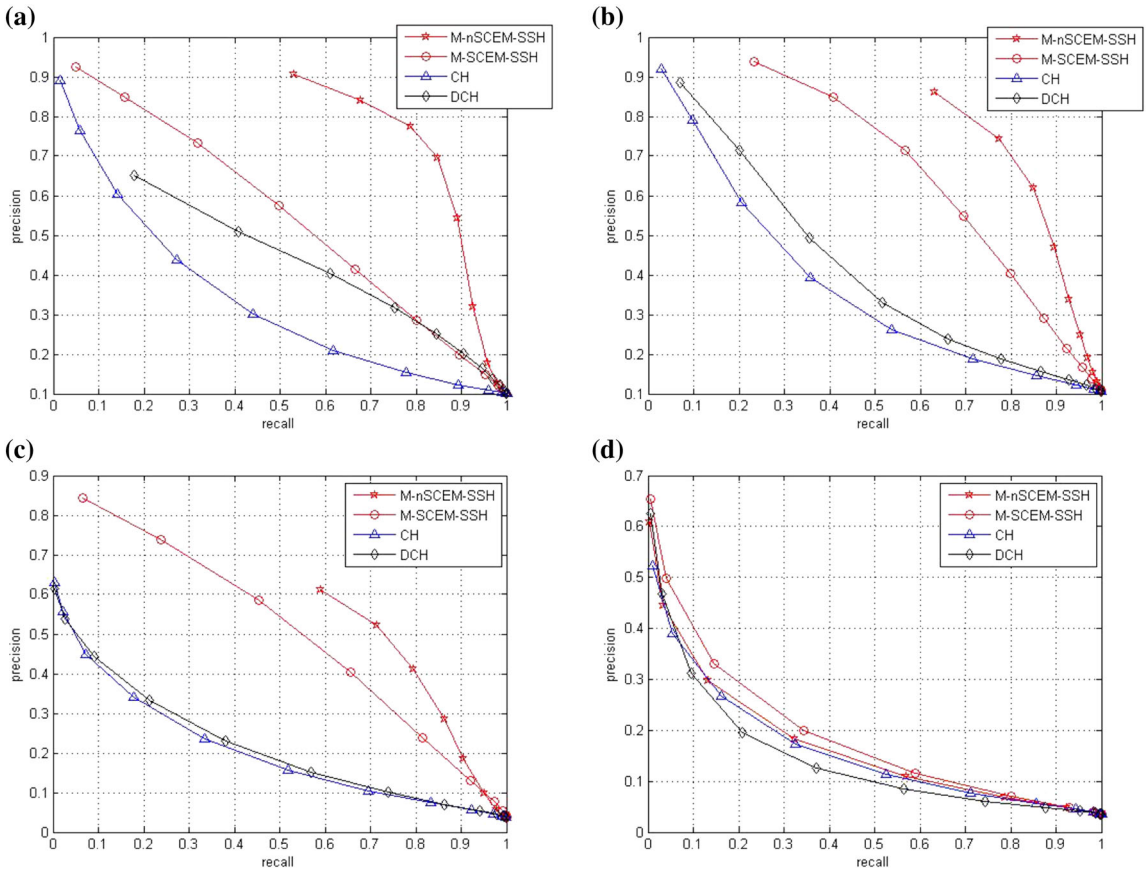

Fig. 12 Precision–recall curves of M-nSCEM-SSH, M-SCEM-SSH, CH and DCH on MNIST (**a**), USPS (**b**), ISOLET (**c**) and Caltech101 (**d**)

4.2 Experimental results for multiple hashing

Experimental comparison among the M-nSCEM-SSH, the M-SCEM-SSH, the CH and the DCH are performed using the four benchmarking databases. All methods use five hash tables with 16 bits each as in [37]. Results are shown in Fig. 12. The M-nSCEM-SSH yields the best performance in all databases. Both the M-nSCEM-SSH and the M-SCEM-SSH outperform the CH and the DCH significantly in the MNIST, the ISOLET and the USPS databases. In experiments of the Caltech101, the M-SCEM-SSH outperforms the M-nSCEM-SSH, which is the same as in experiments using a single hash table. These show the effectiveness of the proposed multiple hashing using the SCEM-SSH.

Then, we use the MNIST database as an example to show the effectiveness of multiple hashing over single hashing for SSH. As aforementioned, multiple hashing visits much fewer hash buckets in comparison to single hashing methods. In Fig. 13, F1-scores versus Hamming distance between hash codes of returned images and the query for 16-, 32-, 48- and 64-bit nSCEM-SSHs and M-nSCEM-SSHs are plotted. For the M-nSCEM-SSH, it only uses the exact match of hash code, so the single result is shown as a straight line across different Hamming distances in Fig. 13. To obtain a similar F1-score of the M-nSCEM-SSH, the nSCEM-SSH using a single hash table needs to visit a very large number of hash buckets and therefore spends longer time for retrieving images. For 16-,

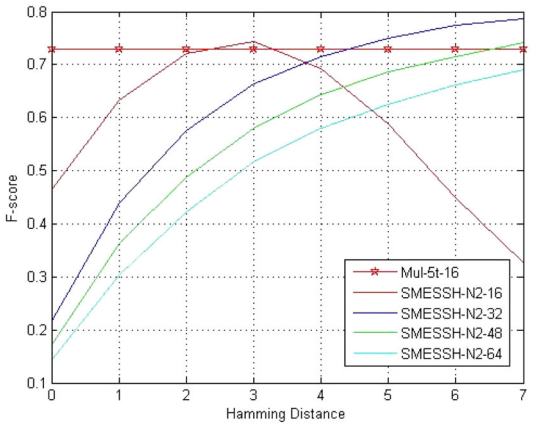

Fig. 13 F1-score versus Hamming distance curves of the M-nSCEM-SSH and 16-, 32-, 48- and 64-bit nSCEM-SSH for MNIST database

32-, 48- and 64-bit nSCEM-SSHs, visiting all hash buckets with Hamming distances less than 2, 4, 6 and 7, respectively, are needed. In other words, the M-nSCEM-SSH only visits five hash buckets while $O(16^2)$, $O(32^4)$, $O(48^6)$, $O(64^7)$ hash buckets are visited by 16-, 32-, 48- and 64-bit nSCEM-SSHs, respectively, to achieve a similar F1-score. When more hash bits are used for single hashing methods, hash buckets are usually smaller and therefore it results in a higher precision and a lower recall rates. In contrast, when the number of hash bits is too small, a higher recall and a lower precision rates are expected. From Fig. 13, images with exact match of hash code to the query using a single 16-bit nSCEM-SSH yields only 0.48 F1-score while the 5-table 16-bit M-nSCEM-SSH yields 0.72 F1-score. The union obtained by using the exact match of hash code from different hash tables provides a large coverage of images with the same hash code to the query image. This region created by multiple hashing is not necessarily convex, while hash buckets created by a single hashing method are convex based on the fact that they are defined by the intersection of a set of hyperplanes.

Query time is critical to the success of semantic image retrieval. Smaller number of hash bucket visits yields a faster response time of image retrieval. Hence overall, the M-nSCEM-SSH is the most powerful version among the four proposed methods. Certainly, user needs to select between linear and nonlinear version of the multiple hashing SCEM-SSH according to the guideline proposed in Sect. 4.1 for optimal performance.

5 Conclusion

This work proposes the SCEM-SSH which maximizes the conditional entropy among all hash bits holistically. The nSCEM-SSH finds the nonlinear mapping by using both local and global information in the database. Then, multiple hashing versions of both the SCEM-SSH and the nSCEM-SSH are proposed. Experimental results show that the proposed methods outperform existing popular SSH methods.

Both the SCEM-SSH and other SSH methods are shown to be useful for dealing with semantic image retrieval problems. However, current experiments (including ours) are performed using a fixed ratio of labeled images. The selection of the optimal ratio between labeled and unlabeled images should be further investigated. Moreover, for an unlabeled database of semantic image retrieval problem, how many images must be labeled to achieve a good performance? A theoretical guideline may be difficult to find owing to the complexity of the semantic representation in feature vectors and uncertainty of future queries. It could be an important future work to study the minimum ratio of labeled images in the database to yield the best precision and recall rates of using SSH.

On the other hand, the selection of both the number of hash bits and the number of hash tables is also important to the success of large scale image retrieval using semi-supervised methods. This problem could be linked to the data distribution and the number of dimensions in the database. A future research direction is to embed the selection of both the number of bits and the number of tables in the optimization process of the M-nSCEM-SSH. A new objective function for the optimization will be needed.

Finally, Euclidean based nonlinear mapping may not be the best choice for semantic image retrieval problems. A further investigation is needed to find a new nonlinear mapping for the SSH and the SCEM-SSH to further improve semantic image retrieval performance for large scale problems.

Acknowledgments This work is supported by a National Natural Science Foundation of China (61272201) and a Program for New Century Excellent Talents in University of China (NCET-11-0162).

References

1. Friedman JH, Bentley JL, Finkel RA (1977) An algorithm for finding best matches in logarithmic expected time. ACM Trans Math Softw 3(3):209–226
2. Silpa-Anan C, Hartley R (2008) Optimised kd-trees for fast image descriptor matching. In: Proceedings of IEEE conference on computer vision and pattern recognition. IEEE, pp 1–8
3. Ciaccia P, Patella M, Zezula P (1997) M-tree: an efficient access method for similarity search in metric spaces. In: Proceedings of the international conference on very large data bases, vol 23. Morgan Kaufmann Pub, San Francisco, pp 426–435
4. Beygelzimer A, Kakade S, Langford J (2006) Cover trees for nearest neighbor. In: Proceedings of the 23rd international conference on machine learning. ACM, New York, pp 97–104
5. Uhlmann JK (1991) Satisfying general proximity/similarity queries with metric trees. Inf Process Lett 40(4):175–179
6. Indyk P (2004) Nearest neighbors in high-dimensional spaces. CRC Press LLC, FL
7. Kundu MK, Chowdhury M, Banerjee M (2012) Interactive image retrieval using m-band wavelet, earth movers distance and fuzzy relevance feedback. Int J Mach Learn Cybern 3(4):285–296
8. Indyk P, Motwani R (1998) Approximate nearest neighbors: towards removing the curse of dimensionality. In: Proceedings of the thirtieth annual ACM symposium on theory of computing. ACM, New York, pp 604–613
9. Gionis A, Indyk P, Motwani R et al (1999) Similarity search in high dimensions via hashing. In: Proceedings of the international conference on very large data bases, vol 99, pp 518–529
10. Andoni A, Indyk P (2006) Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. In: 47th annual IEEE symposium on foundations of computer science. IEEE, pp 459–468
11. Weiss Y, Torralba A, Fergus R (2008) Spectral hashing. NIPS 9(1):6

12. Liu W, Wang J, Kumar S, Chang SF (2011) Hashing with graphs. In: Proceedings of the 28th international conference on machine learning (ICML-11), pp 1–8
13. He J, Radhakrishnan R, Chang SF, Bauer C (2011) Compact hashing with joint optimization of search accuracy and time. In: Proceedings of IEEE conference on computer vision and pattern recognition. IEEE, pp 753–760
14. Gong Y, Lazebnik S, Gordo A, Perronnin F (2013) Iterative quantization: a procrustean approach to learning binary codes for large-scale image retrieval. IEEE Trans Pattern Anal Mach Intell 35(12):2916–2929
15. Norouzi M, Fleet DJ (2013) Cartesian k-means. In: Proceedings of IEEE conference on computer vision and pattern recognition. IEEE, pp 3017–3024
16. Lin Y, Jin R, Cai D, Yan S, Li X (2013) Compressed hashing. In: Proceedings of IEEE conference on computer vision and pattern recognition, ser. CVPR'13. IEEE Computer Society, Washington, DC, pp 446–451. doi:10.1109/CVPR.2013.64
17. He K, Wen F, Sun J (2013) K-means hashing: an affinity-preserving quantization method for learning binary compact codes. In: Proceedings of IEEE conference on computer vision and pattern recognition. IEEE, pp 2938–2945
18. Gong Y, Kumar S, Rowley HA, Lazebnik S (2013) Learning binary codes for high-dimensional data using bilinear projections. In: Proceedings of IEEE conference on computer vision and pattern recognition. IEEE, pp 484–491
19. Liu W, Wang J, Ji R, Jiang YG, Chang SF (2012) Supervised hashing with kernels. In: Proceedings of IEEE conference on computer vision and pattern recognition. IEEE, pp 2074–2081
20. Strecha C, Bronstein AM, Bronstein MM, Fua P (2012) Ldahash: improved matching with smaller descriptors. IEEE Trans Pattern Anal Mach Intell 34(1):66–78
21. Salakhutdinov R, Hinton G (2009) Semantic hashing. Int J Approx Reason 50(7):969–978
22. Norouzi M, Blei DM (2011) Minimal loss hashing for compact binary codes. In: Proceedings of the 28th international conference on machine learning (ICML-11), pp 353–360
23. Xia Z, Feng X, Peng J, Wu J, Fan J (2015) A regularized optimization framework for tag completion and image retrieval. Neurocomputing 147:500–508. (Advances in self-organizing maps subtitle of the special issue: selected papers from the workshop on self-organizing maps 2012 (WSOM'12)
24. Wang J, Kumar S, Chang S-F (2012) Semi-supervised hashing for large-scale search. IEEE Trans Pattern Anal Mach Intell 34(12):2393–2406
25. Wu C, Zhu J, Cai D, Chen C, Bu J (2013) Semi-supervised nonlinear hashing using bootstrap sequential projection learning. IEEE Trans Knowl Data Eng 25(6):1380–1393
26. Datar M, Immorlica N, Indyk P, Mirrokni VS (2004) Locality-sensitive hashing scheme based on p-stable distributions. In: Proceedings of the twentieth annual symposium on computational geometry. ACM, New York, pp 253–262
27. Jain P, Kulis B, Grauman K (2008) Fast image search for learned metrics. In: Proceedings of IEEE conference on computer vision and pattern recognition. IEEE, pp 1–8
28. Grauman K, Darrell T (2007) Pyramid match hashing: sub-linear time indexing over partial correspondences. In: Proceedings of IEEE conference on computer vision and pattern recognition. IEEE, pp 1–8
29. Wang J, Kumar S, Chang S-F (2010) Semi-supervised hashing for scalable image retrieval. In: Proceedings of IEEE conference on computer vision and pattern recognition. IEEE, pp 3424–3431
30. Wang J, Kumar S, Chang S-F (2010) Sequential projection learning for hashing with compact codes. In: Proceedings of the 27th international conference on machine learning (ICML-10), pp 1127–1134
31. Tanha J, van Someren M, Afsarmanesh H (2015) Semi-supervised self-training for decision tree classifiers. Int J Mach Learn Cybern 1–16. doi:10.1007/s13042-015-0328-7
32. Chen W-J, Shao Y-H, Hong N (2014) Laplacian smooth twin support vector machine for semi-supervised classification. Int J Mach Learn Cybern 5(3):459–468. doi:10.1007/s13042-013-0183-3
33. Jiang J, Yan X, Yu Z, Guo J, Tian W (2014) A chinese expert disambiguation method based on semi-supervised graph clustering. Int J MachLearn Cybern 1–8. doi:10.1007/s13042-014-0255-z
34. Alok A, Saha S, Ekbal A (2015) Semi-supervised clustering for gene-expression data in multiobjective optimization framework. Int J Mach Learn Cybern 1–19. doi:10.1007/s13042-015-0335-8
35. Yao C, Bu J, Wu C, Chen G (2013) Semi-supervised spectral hashing for fast similarity search. Neurocomputing 101:52–58
36. Xu H, Wang J, Li Z, Zeng G, Li S, Yu N (2011) Complementary hashing for approximate nearest neighbor search. In: IEEE international conference on computer vision. IEEE, pp 1631–1638
37. Li P, Cheng J, Lu H (2013) Hashing with dual complementary projection learning for fast image retrieval. Neurocomputing 120:83–89
38. Fu H, Kong X, Lu J (2013) Large-scale image retrieval based on boosting iterative quantization hashing with query-adaptive reranking. Neurocomputing 122:480–489
39. Fei-Fei L, Fergus R, Perona P (2007) Learning generative visual models from few training examples: an incremental bayesian approach tested on 101 object categories. Comput Vis Image Underst 106(1):59–70
40. Wang J, Yang J, Yu K, Lv F, Huang T, Gong Y (2010) Locality-constrained linear coding for image classification. In: Proceedings of IEEE conference on computer vision and pattern recognition. IEEE, pp 3360–3367