Multimedia Systems (2015) 21:569–579
DOI 10.1007/s00530-014-0415-8

REGULAR PAPER

Tensor index for large scale image retrieval

**Liang Zheng · Shengjin Wang · Peizhen Guo ·
Hanyue Liang · Qi Tian**

Received: 8 April 2014 / Accepted: 26 August 2014 / Published online: 12 October 2014

Abstract Recently, the bag-of-words representation is widely applied in the image retrieval applications. In this model, visual word is a core component. However, compared with text retrieval, one major problem associated with image retrieval consists in the visual word ambiguity, i.e., a trade-off between precision and recall of visual matching. To address this problem, this paper proposes a tensor index structure to improve precision and recall simultaneously. Essentially, the tensor index is a multi-dimensional index structure. It combines the strengths of two state-of-the-art indexing strategies, i.e., the inverted multi-index [Babenko and Lempitsky (Computer vision and pattern recognition (CVPR), 2012 IEEE Conference, 3069–3076, 2012)] as well as the joint inverted index [Xia et al. (ICCV, 2013)] which are initially designed for approximate nearest neighbor search problems. This paper, instead, exploits their usage in the scenario of image retrieval and provides insights into how to combine them effectively. We show that on the one hand, the multi-index enhances the discriminative power of visual words, thus improving precision; on the other hand, the introduction of multiple codebooks corrects quantization artifacts, thus improving recall. Extensive experiments on two benchmark datasets demonstrate that tensor index significantly improves the baseline approach. Moreover, when incorporating methods such as Hamming embedding, we achieve competitive performances compared to the state-of-the-art ones.

Keywords Tensor index · Image retrieval · Bag-of-words model

1 Introduction

This paper considers the task of large-scale partial duplicate image retrieval. Given a query image, our goal is to retrieve images which contain the same object or scene from a large database. A successful search engine must return the most relevant images (effective) to the user in a short amount of time (efficient). In this paper, we aim at designing effective methods at the price of affordable memory or time cost (Fig. 1).

The past decade has witnessed a great progress in the community of bag-of-words (BoW) based image retrieval [21, 23, 28]. Motivated by the pipeline of text retrieval, the image retrieval process detects local invariant features [20] and quantizes them to visual words using a pre-trained codebook. Therefore, each image is converted into an orderless bag of visual words, analogous to words in the textual documents. These visual words are weighted by tf-idf scheme or its variants [10, 46]. To improve retrieval efficiency and deal with large-scale data, the inverted index data structure is employed.

Essentially, the visual words play a vital role in the BoW model: two local descriptors are considered as a match if

Communicated by F. Wu.

L. Zheng · S. Wang (✉) · P. Guo · H. Liang
Department of Electronic Engineering, Tsinghua University,
Beijing 100084, China
e-mail: wgsgj@tsinghua.edu.cn

L. Zheng
e-mail: zheng-l06@mails.tsinghua.edu.cn

P. Guo
e-mail: gpz0617@126.com

H. Liang
e-mail: mslianghy@sina.com

Q. Tian (✉)
University of Texas, San Antonio, TX 78249, USA
e-mail: qitian@cs.utsa.edu

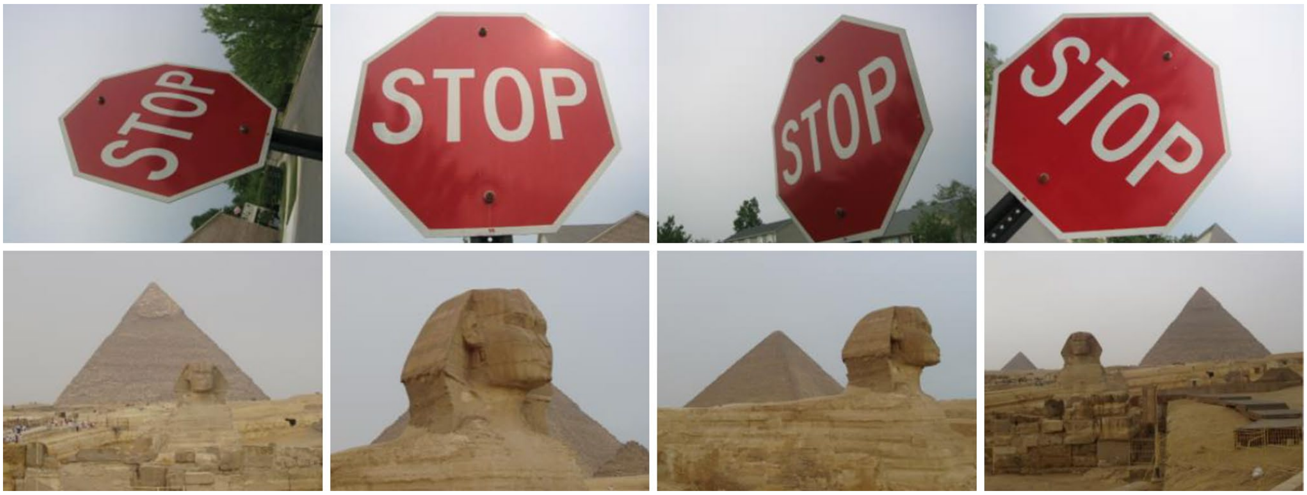

Fig. 1 Two examples of partial-duplicate images from (*top*) Ukbench [21] and (*bottom*) Holidays [10] datasets

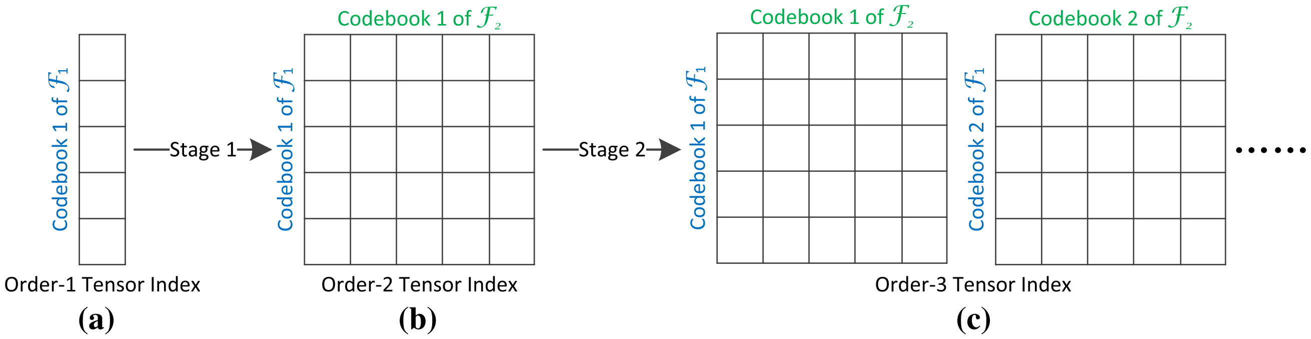
Codebook 1 of $\mathcal{F}_1$
Stage 1
Codebook 1 of $\mathcal{F}_2$
Codebook 1 of $\mathcal{F}_1$
Stage 2
Codebook 1 of $\mathcal{F}_2$
Codebook 1 of $\mathcal{F}_1$
Codebook 2 of $\mathcal{F}_2$
Codebook 2 of $\mathcal{F}_1$
......
Order-1 Tensor Index
(a)
Order-2 Tensor Index
(b)
Order-3 Tensor Index
(c)

Fig. 2 The construction of the tensor index. **a** Order-1 tensor index (the classic inverted index). **b** Order-2 tensor index (inverted multi-index). **c** Order-3 tensor index. Stage 1 converts **a**, **b** and stage 2 transforms **b**, **c**. Two features used are denoted as $\mathcal{F}_1$ and $\mathcal{F}_2$. For each feature, multiple codebooks are trained

they are quantized to the same visual word. However, this process is impaired by visual word ambiguity [7]. On the one hand, features which are visually dissimilar maybe located in the same voronoi cell, leading to low precision. On the other hand, features which are visually similar may be quantized to different visual words, leading to low recall. Generally speaking, precision and recall are competing forces that often counteract each other. It is often the case that a high precision tends to be accompanied by a low recall, and vice versa. Consequently, sustained precision gains should often be counter-balanced by improved recall if possible.

To tackle this problem, this paper proposes the tensor index data structure to improve precision and recall of visual matching simultaneously. In a nutshell, tensor index is composed of two stages (see Fig. 2). First, a two-dimensional multi-index [2] (also called order-2 tensor index) is constructed, with each dimension corresponding to a distinct codebook. As a result, a keypoint in the image is quantized to two complementary visual words instead of only one in the BoW baseline. This scheme serves to enhance the discriminative power of each keypoint, so the precision of visual matching is improved. In the second stage, for each dimension, multiple codebooks are generated using the joint inverted index method proposed in [36]. Therefore, several two-dimensional inverted indexes are produced, i.e., order-3 tensor index. With multiple codebooks trained for each dimension of the multi-index, the retrieval process merges the candidate images from multiple order-2 tensor indexes. Through this manner, the recall is improved as well. If not specified, we refer to order-3 tensor index throughout this paper when "tensor index" is mentioned.

In the first stage, we provide two alternative strategies to build a order-2 tensor index. The first strategy is similar to the one proposed in [2], where a 128-D SIFT descriptor is decomposed into two 64-D segments. Then, two codebooks are trained for each segment, respectively. The second strategy consists in a feature fusion scheme. Each keypoint is described by a color feature and a SIFT feature, respectively, from which two codebooks are generated, respectively.

Experiments on two benchmark datasets confirm that tensor index is capable of improving the baseline performance dramatically. If we further incorporate other complementary methods, the proposed method achieves competitive results compared with the state-of-the-art systems. Moreover, large-scale experiments indicate that tensor index consumes similar query time to the baseline approach.

The remainder of this paper is organized as follows: In Sect. 2, we briefly review several closely related aspects in BoW based image retrieval. Then, our method is described in Sect. 2.1. After a detailed presentation and discussion of the experimental results in Sect. 3, we draw our conclusions in Sect. 4.

2 Related work

Visual word is the core component of the BoW model. Therefore, the trade-off between precision and recall of visual matching has always been a focus of recent researches.

In order to improve matching precision, one strategy is to design better quantization algorithms. Methods such as sparse coding [32, 38], constrained quantization [6], and soft quantization [24] try to reduce quantization error by assigning confidence coefficients to quantized visual words. To accelerate quantization, raw features can be first converted into binary features, on which schemes such as scalar quantization [48], nested quantization [4] depends. Another strategy involves augmenting visual words with complementary information, e.g., spatial constraints [27, 37, 44], contextual description [16, 18, 22, 30], multiple features [29, 31, 34, 39], and social and behavioral cues [17]. For example, RANSAC verification [23] estimates a global affine transformation as a post-processing step, at a cost of expensive computational complexity. Meanwhile, visual elements of higher orders, such as visual phrases [43], visual phraselets [44], provide pairwise constraints to eliminate false matches. Nevertheless, a typical drawback of these methods is the corresponding impact on efficiency due to their complex nature.

On the other hand, many methods are proposed to improve recall. Typically, a large codebook (1 M) means a fine partition of the feature space, corresponding to a low recall, while a small codebook (20 K) [10] guarantees a high recall. However, with a small codebook, the lists in the inverted index can be very long, thus demanding much longer query time. Other solutions involve multiple assignment (MA) [10], or image-level feature fusion techniques [41, 42]. In [41], relevant images are retrieved by different features, and a graph fusion is undertaken to refine the rank results. In [42], the inverted index is expanded according to the consistency in global feature spaces, which in turn improves recall. Moreover, using multiple codebooks or inverted indexes [3, 5, 9, 47] is also beneficial since it corrects quantization artifacts to some extent and covers more area in the feature space.

This paper focuses on improving precision and recall from the view of indexing strategies. The inverted index greatly promotes the efficiency of BoW based image retrieval. Each entry of the inverted index contains a list of indexed features or postings. For a document-level inverted index [23], each posting stores the image ID (imgID) and the term frequency (tf); for a word-level inverted index [10], each posting stores the imgID and other information associated with this word. In the field of ANN search, two state-of-the-art inverted index organizations include the inverted multi-index [2] and the joint inverted index [36]. The former strategy is a 2-D inverted index aiming at improving precision, while the latter involves constructing multiple one-dimensional inverted index, aiming at improving recall. This paper first evaluates the two methods in the scenario of image retrieval which differs from ANN search in that a query feature does not necessarily have a true nearest neighbor in image retrieval. Second, we proposes to couple the two methods into an order-3 tensor index, which achieves even better performance.

2.1 Our approach

As shown in Fig. 2, the proposed method consists of two stages: the construction of the multi-index in Sect. 2.2, and the construction of the tensor index in Sect. 2.3. We describe the query process in Sect. 2.4.

2.2 Constructing inverted multi-index

This paper considers the two-dimensional multi-index, also called second-2 tensor index in this paper. In order to exploit the usage of multi-index in image retrieval, we propose two variants. The first is similar to [2]: the two dimensions correspond to two 64-D segments of the SIFT descriptor. For the second variant, the two dimensions correspond to the SIFT and color features, respectively. We denote the two variants as MI_{S-S} and MI_{S-C}, respectively.

2.2.1 SIFT-SIFT multi-index

For MI_{S-S}, we essentially follow the same procedure as [2]. Each SIFT descriptor is split into two 64-D segments

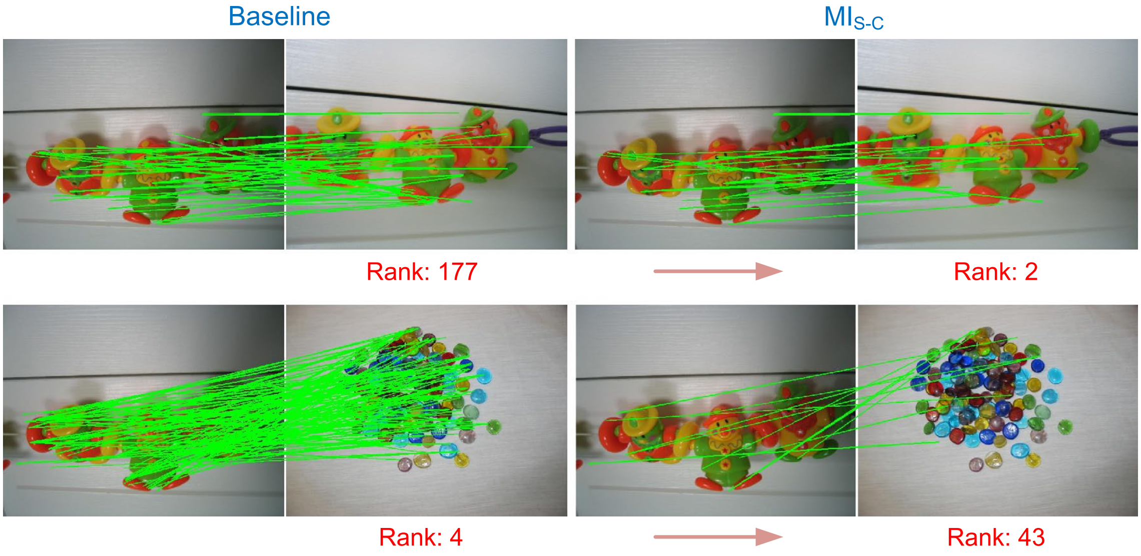

Fig. 3 An example of visual matching using the baseline (*left*) and MI_{S-C} (*right*) methods. For each image pair, the *left one* is the query image. *Images* in the *first row* are partial-duplicates, while the *second row* contains irrelevant ones. Also shown are the ranks of the candidate images. We can see that the combination of SIFT and color descriptors effectively removes false matches. Specifically, a non-trivial fraction of matches remains for the relevant image pair, while only few features are preserved for the irrelevant images. The rank of the candidate images are refined correspondingly

with product quantization (PQ) [13]. With PQ, two codebooks are trained for each segment. In terms of Fig. 2b, the two segments correspond to $\mathcal{F}_1$ and $\mathcal{F}_2$, and the two codebooks correspond to "codebook 1 of $\mathcal{F}_1$" and "codebook 1 of $\mathcal{F}_2$", respectively.

During the offline indexing procedure, each SIFT descriptor is first split into halves and then quantized to a visual word pair (u, v), where u and v are visual words defined in codebooks U and V, respectively. Then, the entry (u, v) in the multi-index is identified, in which the imgID and other meta data of the input descriptor is stored.

2.2.2 *SIFT-color multi-index*

The SIFT descriptor captures the gradient distribution of a local region. To enhance the discriminative power of the SIFT descriptor, this paper further extracts a color descriptor at each keypoint in the image. Specifically, we use the Color Names (CN) descriptor [26], which calculates a 11-D feature vector for each pixel. Each entry of the CN descriptor encodes one of the eleven basic colors: black, blue, brown, grey, green, orange, pink, purple, red, white, and yellow. For each detected keypoint, a 128-D SIFT descriptor and a 11-D CN descriptor are computed. Then, two codebooks are trained for the SIFT and CN features, respectively.

The procedure of constructing the MI_{S-C} structure is in essence similar to that described in Sect. 2.2.1. For a detected keypoint, after extracting the SIFT and CN descriptors, a visual word pair is also generated, each corresponding to the nearest visual word in the SIFT and CN codebooks, respectively. Then, the two-dimensional multi-index is padded with the input features.

The impact of MI_{S-C} on visual matching is illustrated in Fig. 3. In this example, the codebook sizes for SIFT and CN descriptors are 20 K and 200, respectively, and we exert a Multiple Assignment of 30 during CN quantization. We can see that the baseline method produces many false matches, most of which can be eliminated when MI_{S-C} is applied. For the relevant image pair in the first row, the rank promotes from 177 to 2; for the irrelevant image pair, the rank drops from 4 to 43. As a consequence, Fig. 3 indicates that the MI_{S-C} method greatly enhances the discriminative ability of visual matching.

2.2.3 *Discussion*

As can be seen from Sects. 2.2.1 and 2.2.2 as well as Fig. 4, the common property of the two methods concerns that a keypoint is described by a visual word pair (u, v), instead of one single visual word in the baseline. With this representation, two keypoints are matched iff they are quantized to an identical visual word pair. As a result, the discriminative ability (precision) is enhanced.

On the other hand, the difference between MI_{S-S} and MI_{S-C} is that the former creates a finer partition of the SIFT feature space, while the latter is a feature fusion scheme. For MI_{S-S}, two SIFT descriptors are viewed as a

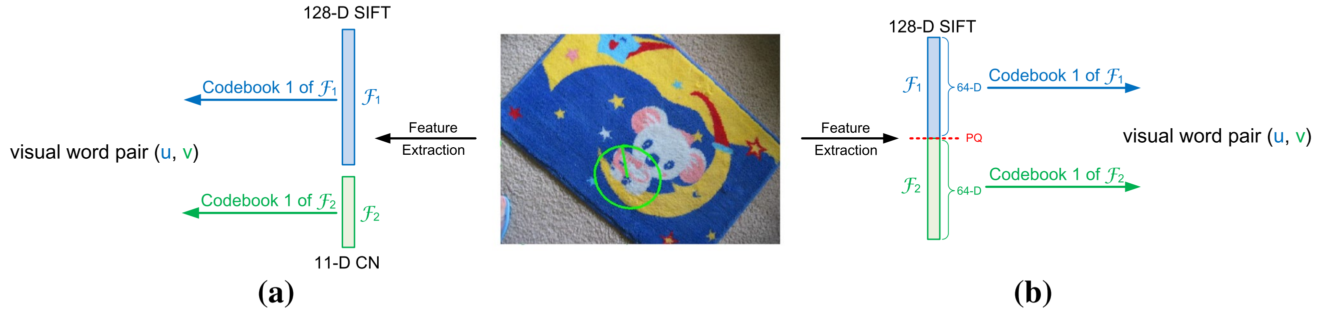

Fig. 4 Feature extraction and quantization for **a** MI_{S-C} and **b** MI_{S-S}. In **a**, a keypoint in the image is co-described by SIFT and CN descriptors, and subsequently quantized to a visual word pair using two codebooks. In **b**, the 128-D SIFT descriptor is split into two segments, again producing a visual word pair

match if they are similar in both the first and the second halves. For MI_{S-C}, however, a local region is co-described by SIFT and color features, both of which have to be similar to generate a valid match.

2.2.4 *Injecting binary signatures*

To further enhance the discriminative ability of visual word pairs, we embed binary signatures [10] into the multi-indexes.

For MI_{S-S} method, the binary signature is generated in a similar manner to [10]. The binarization threshold is calculated using training data (128-D SIFT) which fall into the corresponding entry of MI_{S-S}. We produce 64-bit binary signatures. During online query, the Hamming distance d_b between two binary signatures is computed. If d_b is smaller than a pre-defined threshold κ, then a weight in the form of $\exp(-\frac{d_b^2}{\sigma^2})$ is added to the scoring function, i.e.,

$$w(d_b) = \begin{cases} \exp(-\frac{d_b^2}{\sigma^2}), & \text{if } d_b < \kappa, \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

As with the MI_{S-C} method, we calculate two binary signatures, each for the SIFT and CN descriptors, respectively. For the SIFT descriptor, we use the method described above to produce a 64-bit signature. For the 11-D CN descriptor represented as $(f_1, f_2, \ldots, f_{11})^{\text{T}}$, a 22-bit binary feature $\boldsymbol{b}$ can be produced as follows:

$$(b_i, b_{i+11}) = \begin{cases} (1,1), & \text{if } f_i > \hat{th}_1, \\ (1,0), & \text{if } \hat{th}_2 < f_i \le \hat{th}_1, \\ (0,0), & \text{if } f_i \le \hat{th}_2, \end{cases} \tag{2}$$

where $b_i (i = 1, 2, ..., 11)$ is the ith entry of the resulting binary feature $\boldsymbol{b}$. Thresholds $\hat{th}_1 = g_5$, $\hat{th}_2 = g_2$, where $(g_1, g_2, ..., g_{11})^{\text{T}}$ is the sorted vector of $(f_1, f_2, ..., f_{11})^{\text{T}}$ in descending order. Given a query feature, we calculate the Hamming distances and corresponding weights (similar to MI_{S-S}) of the SIFT and CN signatures, respectively, and use the multiplication as the final weight for visual matching.

2.3 Constructing tensor index

In essence, the tensor index is a multiple multi-index structure (see Fig. 2c). Via stage 2 in Fig. 2, one order-2 tensor index is expanded to order-3 tensor index.

Basically, starting from the order-2 tensor index shown in Fig. 2b, for features $\mathcal{F}_1$ and $\mathcal{F}_2$ each, multiple codebooks are trained using Approximate K-means (AKM) [23]. For each feature, however, if the multiple codebooks are trained independently, we may encounter the problem described in [9, 36], i.e., the correlation among them may be large, which counteracts the benefits of multi-codebook merging. To avoid the problem of codebook correlation, the codebooks are trained using the method proposed in [36], i.e., the joint inverted index.

Specifically, for each feature (or dimension) of the multi-index, assume that the desired number of codebooks is K, and the codebook size is S. Following [36], instead of training K codebooks of size S independently, we actually train one codebook of size $K \cdot S$. Then, the $K \cdot S$ cluster centers are assigned to K codebooks after a grouping operation using balanced clustering algorithms. Nevertheless, we find in our experiment that a random assignment yields similar results. This is probably due to the low value of K and large value of S. Therefore, in the visual word optimization step, visual words are assigned randomly, generating K codebooks of size S. In Sect. 3, we will provide a brief comparison between generating codebooks jointly and generating them independently.

After the codebooks for each feature are jointly trained, the order-3 tensor index can be assembled. Assume that we want to construct K order-2 tensor index. Then, for each of the K multi-indexes to be generated, we randomly pick one codebook from each of the two features. Through this manner, we are capable of building the order-3 tensor index which consists of K two dimensional multi-indexes.

2.4 Querying tensor index

Given an query image q, invariant keypoints are first detected. Then, as shown in Fig. 4, each keypoint is quantized to a visual word pair (u, v), using two codebooks of features $\mathcal{F}_1$ and $\mathcal{F}_2$, respectively. Then, the entries (u, v) in each of the K multi-indexes are located, from which K lists of postings or candidate images are identified. With these K lists, we simply concatenate them and these postings contribute to the final score of the corresponding images.

In the tensor index framework, basic elements in the classic BoW model such as the tf-idf weights, the L_2 normalization can be readily adopted. Specifically, for each visual word pair (entry) in the multi-index (MI_{S-S} or MI_{S-C}), its idf value is calculated as,

$$\text{idf}(u, v) = \log\left(\frac{N}{n_{uv}}\right), \tag{3}$$

where N is the total number of images in the database, and n_{uv} encodes the number of images containing the visual word pair (u, v). Moreover, the L_2 norm of a database image can be computed as ,

$$\|I\|_2 = \left(\sum_u \sum_v h_{u,v}^2\right)^{\frac{1}{2}}, \tag{4}$$

where $h_{u,v}$ is the term-frequency (tf) of visual word pair (u, v) in image I. The L_2 normalization is exerted on the image scores, so as to penalize images with more visual words, and vice versa.

Since the MI_{S-S} and MI_{S-C} typically achieve high precision due to the two-dimension nature, we employ multiple assignment (MA) [10] to improve recall. We apply MA only on the query image, and in consideration for the illumination changes, we set a relatively large MA value for CN quantization. The tuning of parameter MA is fully discussed in Sect. 3. Note, however, that the MA strategy may have an influence on the effectiveness of normalization scheme in Eq. 4. Specifically, we find in the experiments that Eq. 4 works well for MI_{S-S} which can be viewed as a symmetrical structure. On the other hand, for MI_{S-C}, the baseline L_2 norm for the SIFT feature works better, probably due to the large MA value for CN.

To sum up, the tensor index is a composite inverted index structure that takes advantage of both the inverted multi-index and the joint inverted index. During query time, the query requests are processed independently in each of the K multi-indexes, before the scores are merged to yield the final results.

3 Experiments

In this section, experimental results on two public available datasets are summarized and discussed.

3.1 Datasets

Holidays [10] This dataset consists of 1,491 images from personal holiday photo collections. 500 images are selected as query images. Most queries have 1–2 ground truth images. The mean average precision (mAP) is employed for accuracy measurement.

Ukbench [21] This dataset contains 10,200 images of 2,550 groups. Each group has four images containing the same object, but taken under different views or illuminations. Each of the 10,200 images is taken as the query image in turn. The number of relevant images in the top-4 ranked images is averaged over the 10,200 queries, denoted as N-S score (maximum 4).

MIR Flickr 1M [8] This is a distractor dataset, with one million images randomly retrieved from Flickr. We add this dataset to test the scalability of our method.

3.2 Baseline

This paper adopts the baseline approach proposed in [10, 23]. For each image, we employ the Hessian-Affine detector and the SIFT descriptor. Moreover, RootSIFT [1] is used at every point in the system using l_1-normalization followed by a square root operation, as it yields good performance under Euclidean distance. In [1], it is shown that RootSIFT consistently brings about improvement of +0.02 to +0.03 in mAP. Our preliminary experiments also confirmed this performance gain. For MI_{S-S}, RootSIFT is applied on the two segments separately. We also employ the average IDF proposed in [46] to produce a higher baseline. For clustering, the AKM algorithm [23] is implemented. With a 20 K codebook trained on independent data, the baseline results for Holidays and Ukbench are 49.23% in mAP and 3.02 in N-S score, respectively. Both baselines are higher than those reported in [10, 12].

3.3 Parameter selection

Hamming embedding There are two main parameters in HE: the Hamming threshold κ and the weighting parameter σ. For the SIFT codebooks, we follow the settings in [10] and set $\kappa = 22, \sigma = 16$. On the other hand, for a CN codebook of size 200 in MI_{S-C}, we set $\kappa = 7, \sigma = 4$, which yields satisfying performance in our experiments.

Multiple assignment For MI_{S-S} and MI_{S-C}, MA is applied to the query image. In order for MA to work well,

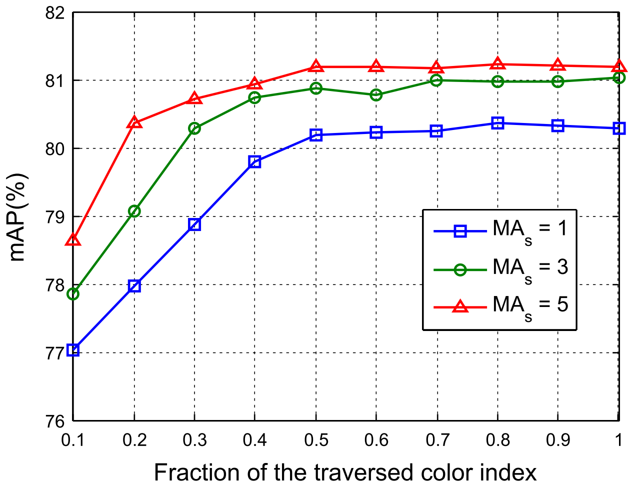

Fig. 5 Impact of MA for MI_{S-C} on Holidays dataset. The codebook sizes for SIFT and CN features are 20 K and 200, respectively. We set MA_c to 200 × 50 % = 100, and MA_s to 5

Table 1 The impact of MA on the performance of MI_{S-S} for holidays dataset

MA	1	2	3	4	5
Holidays, mAP (%)	69.51	73.33	74.36	73.28	72.58

Table 2 The performance of various methods on Holidays and Ukbench datasets

Methods	Holidays	Ukbench	
	mAP (%)	N-S	mAP (%)
MI_{S-S}	54.41	3.15	79.21
MI_{S-S} + HE + MA	74.36	3.38	87.16
Tensor_{S-S}	77.34	3.49	90.75
MI_{S-C}	57.65	3.21	81.92
MI_{S-C} + HE + MA	83.19	3.63	92.69
Tensor_{S-C}	84.06	3.68	93.82

we use HE in the experiments, which is also suggested in [10]. Table 1 presents the impact of MA on MI_{S-S}. When MA = 3, MI_{S-S} obtains an mAP of 74.36 %, so we set MA to 3 in MI_{S-S}. The influence of MA on MI_{S-C} is demonstrated in Fig. 5. From Fig. 5, we can see that when MA for color feature increases, mAP first rises and then remains stable. Since a smaller MA leads to less the query time, we set MA_c to 50%. In addition, MA_s is set to 5 for MI_{S-C}.

3.4 Evaluation

Impact of codebooks The codebooks may have an influence on retrieval performance. For the color codebook, we set its size to 200, because this size produces superior performance in the preliminary experiments. Meanwhile, SIFT codebooks of various sizes are generated, and the results are presented in Fig. 6, from which three major conclusions can be drawn.

First, we can see that results vary with codebook sizes. Specifically, for Holidays dataset, the superior codebook size is 20 K and 1 K for MI_{S-C} and MI_{S-S}, respectively. For Ukbench, however, we observe a better performance of 10 K and 1 K for MI_{S-C} and MI_{S-S}, respectively. Nevertheless, the performance gap is not big, so in the following experiments, we use 20 K and 1 K codebooks for the two structures, respectively.

Second, as the number of multi-indexes (the parameter K in Sect. 2.3) increases, we typically obtain a better performance. For example, with a 20 K codebook, the mAP is improved from 83.19 to 84.06 % when K increase from 1 to 2 on Holidays; with the same settings, the N-S score rises from 3.63 to 3.67 on Ukbench. However, we note that using three multi-indexes brings about little, if any, improvement. This is because the extra information introduced by merging the third codebook is very limited. Our observation is very similar to [9].

Third, we compare our method with generating codebooks independently (denoted as “Indep” in Fig. 6). The results indicate that when $K = 1$, independently trained codebook has a similar performance to jointly trained codebook. But when K is increased to 2 or 3, joint training has a clear advantage. According to [36], jointly trained codebooks have lower correlation among each other, so the merging action brings more benefits.

Comparison between MI_{S-S} and MI_{S-C} The performance of the two tensor index variants can be observed in Fig. 6 and Table 2. These results demonstrate that MI_{S-C} has a superior performance over MI_{S-S}. The primary reason is that MI_{S-C} employs complementary information (color feature) to provide additional discriminative power. On Ukbench and Holidays datasets, the color feature is a good discriminator [35, 41, 45]. But when the illumination changes dramatically, it might be the case that MI_{S-S} works better. Furthermore, we also find some problems associated with the multi-index scheme in image retrieval. In MI_{S-S} with two 1K codebooks, the total number of visual word pairs equals 1M. But we find in our experiments that a majority of the entries in the multi-index are empty, i.e., many entries have been wasted. Therefore, the performance of MI_{S-S} can be further promoted if this problem is addressed. In Fig. 7, we present some sample query results of the 20 K baseline, tensor_{S-S} and Tensor_{S-C}, respectively. Different working mechanism can be observed from these results.

Large-scale experiments To test the scalability of the proposed method, we populate the Holidays dataset with

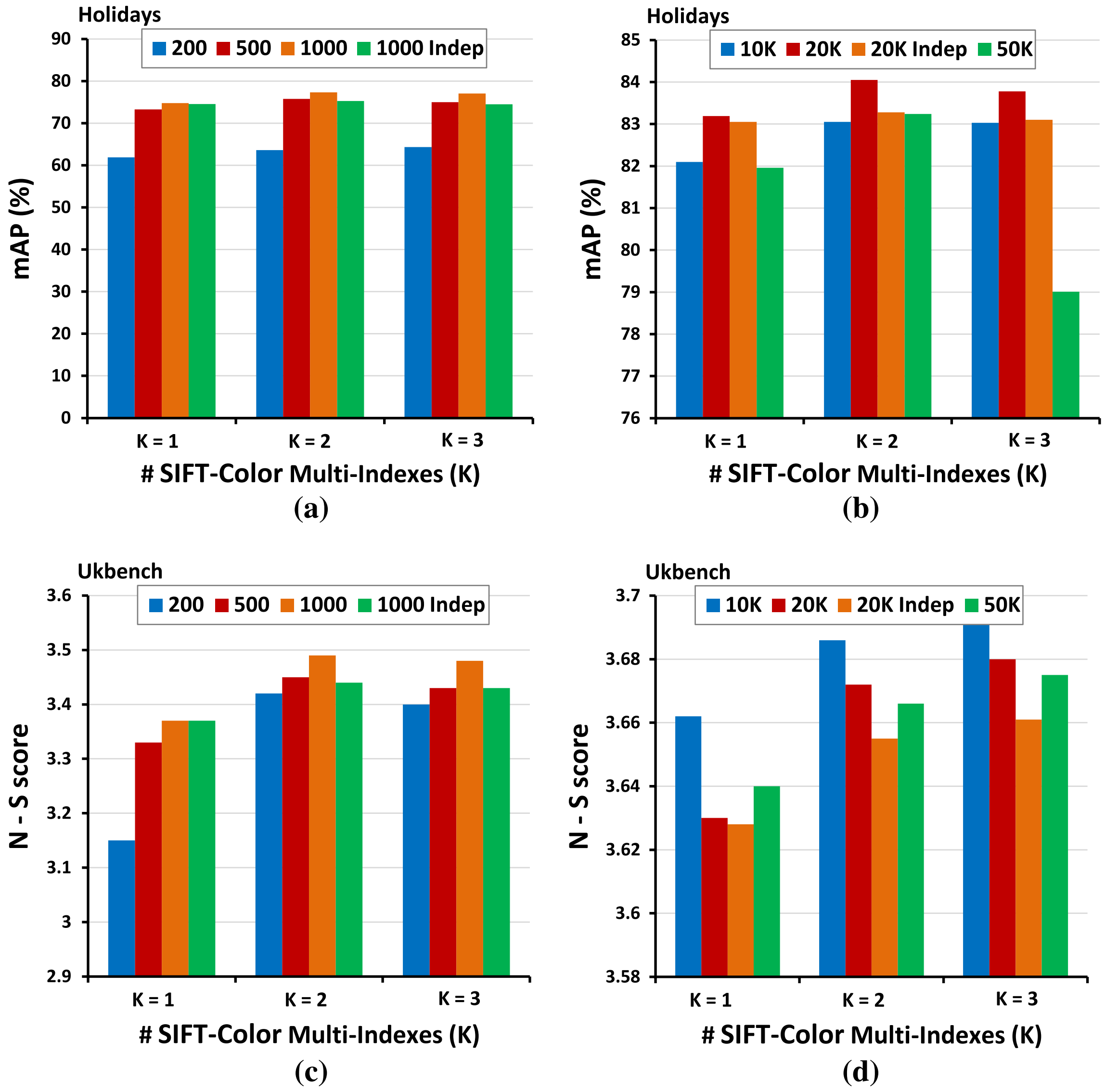

Fig. 6 Results of Tensor$_{S-S}$ (**a**, **c**) and Tensor$_{S-C}$ (**b**, **d**) on Holidays and Ukbench datasets. SIFT codebooks of different sizes are compared. The CN codebook size is set to 200. We merge different numbers (K) of multi-indexes jointly trained to derive the results of the tensor index. The bar denoted by "indep" shows the results obtained by training codebooks independently, not jointly **a** Holidays with tensor$_{S-S}$, **b** Holidays with tensor$_{S-C}$, **c** Ukbench with Tensor$_{S-S}$, **d** Ukbench with Tensor$_{S-C}$

various fractions of the MIR Flickr 1M dataset. Specifically, we set $K = 2$ for both tensor$_{S-S}$ and Tensor$_{S-C}$. Two 1K codebooks are used for MI$_{S-S}$; one SIFT codebook of size 20 K and one CN codebook of size 200 are used for MI$_{S-C}$. For comparison, we plot curves obtained by combinations of different techniques. Note that we do not set a baseline for Tensor$_{S-S}$ since there is not a suitable codebook size of a baseline to be compared with.

Figure 8 demonstrates the performance on Holidays dataset of tensor$_{S-S}$ and tensor$_{S-C}$. We can observe that the proposed tensor index method has consistently superior results under each database size. Specifically, on the Holidays + 1M dataset, we obtain mAP results of 61.35 and 73.26 % for tensor$_{S-S}$ and tensor$_{S-C}$, respectively. Although the MA component has similar effects on improving recall, we still find improvements of tensor index over MI + HE + MA. This result proves the effectiveness of joint indexing strategy towards a higher recall.

We also present the average query time for the Holidays + 1M dataset in Table 3. The experiments are performed on a server with 3.46 GHz CPU and 64 GB memory. The baseline approach with a 20 K codebook consumes 2.28 s to perform a query. On the other hand, it takes 1.65 s and 2.01 s for tensor$_{S-S}$ and tensor$_{S-C}$, respectively. Note that in this case, we do not apply MA, since it dramatically increases the query time.

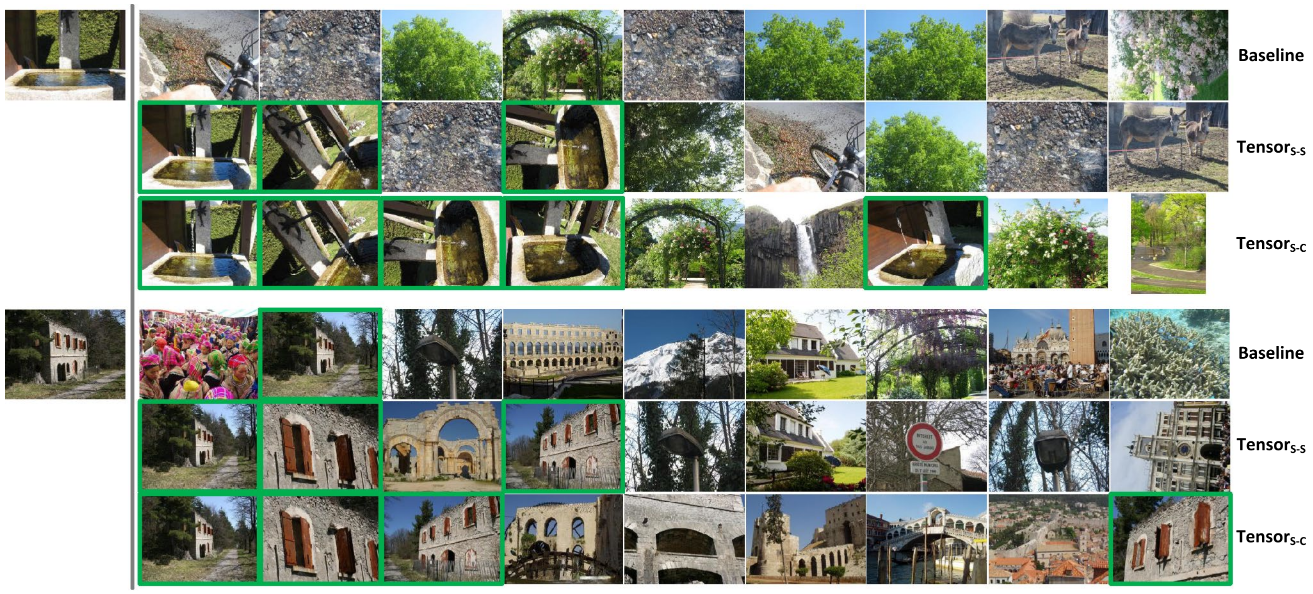

Fig. 7 Sample retrieval results of query 366 and 448 on Holidays dataset. The query is on the *left*. For each query, the *three rows* correspond to results of the 20 K baseline, tensor$_{S-S}$ and tensor$_{S-C}$, respectively

Table 3 The average query time (s) of tensor$_{S-S}$ and tensor$_{S-C}$ for Holidays + 1M dataset

Method	20 K baseline	Tensor$_{S-S}$	Tensor$_{S-C}$
Query time (s)	2.28	1.65	2.01

Table 4 Memory cost for different approaches, i.e., baseline with 20 K codebook, tensor$_{S-S}$, and tensor$_{S-C}$

Methods	Methods	Tensor$_{S-S}$		Tensor$_{S-C}$	
		Order-2	Order-3	Order-2	Order-3
Per feature (bytes)	4	12	24	14.75	29.5
1M dataset (GB)	1.7	5.0	10.1	6.1	12.2

Furthermore, Table 4 provides a summary of the memory usage of different methods. In the baseline, 4 bytes are consumed to store image ID for each indexed feature. For order-2 tensor$_{S-S}$ (also called MI$_{S-S}$), another 8 bytes are allocated for the 64-bit Hamming signature, while order-3 tensor$_{S-S}$ (simplified as Tensor$_{S-S}$ in the text) doubles the memory cost. When integrating color feature, 2.75 more bytes are introduced for order-2 tensor$_{S-C}$, and 5.5 bytes for order-3 tensor$_{S-C}$. As a consequence, Fig. 8, Tables 3 and 4 demonstrate that our method has relatively low query time and demands acceptable memory cost.

3.5 Comparison with state-of-the-arts

In this section, we compare our method against some state-of-the-art systems in the literature. The results are shown in Table 5. Note that the listed results are obtained without post-processing steps. Table 5 indicates that the proposed method compares favorably to the state-of-the-arts. Notably, our final result is mAP = 84.1 % for Holidays, and N-S score = 3.68 for Ukbench. Our result exceeds [14] by 0.07 in N-S score on Ukbench, and 0.2 % in mAP on Holidays. These comparisons confirm the effectiveness of our method. We point out that techniques such as burstiness weighting [11], spatial constraints [10], etc, may also contribute to our framework. On the other hand, various post processing steps, such as the graph fusion [19, 41], RANSAC verification [23] and query expansion [1], can be directly applied on top of our method.

Table 5 Performance comparison with state-of-the-art methods without post-processing

Methods	Tensor	[33]	[14]	[27]	[42]	[35]	[12]	[25]	[11]
Ukbench, N-S score	3.68	3.56	3.61	3.52	3.60	3.50	3.42	–	3.54
Holidays, mAP (%)	84.1	78.0	–	76.2	80.9	78.9	81.3	82.1	83.9

Note that order-2 tensor$_{S-S}$ and order-2 tensor$_{S-C}$ are referred to as MI$_{S-S}$ and MI$_{S-C}$ in the text, while order-3 tensor$_{S-S}$ and order-3 tensor$_{S-C}$ are simplified as tensor$_{S-S}$ and tensor$_{S-C}$ in the text, respectively

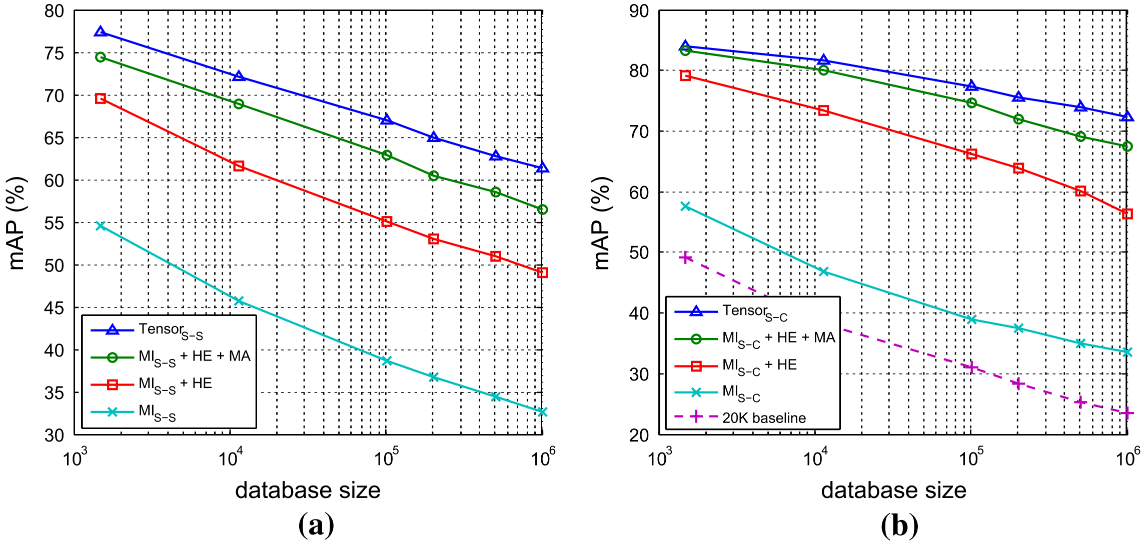

Fig. 8 Large-scale experimental results on Holidays + MIR Flickr 1M dataset. Combinations of different methods are presented. The SIFT codebooks are 1 K and 20 K for tensor$_{S-S}$ and tensor$_{S-C}$, respectively. The color codebook size is 200 for tensor$_{S-C}$. **a** Holidays + 1M with tensor$_{S-S}$, **b** Holidays + 1M with tensor$_{S-C}$

4 Conclusions

In this paper, we propose the tensor index data structure, which integrates both the inverted multi-index [2] and the joint inverted index [36]. With the multi-index component, we are capable of improving the precision of visual matching. On the other hand, with the joint index method, codebooks with less correlation are generated, which serves to improve recall. Since the inverted index and the joint index are initially proposed in the scenario of ANN search, we exploit their usage in image retrieval by constructing two tensor index variants, i.e., the SIFT-SIFT tensor index (tensor$_{S-S}$) and SIFT-color tensor index (tensor$_{S-C}$). To further enhance the discriminative power, we inject binary signatures. Extensive experiments on Holidays and Ukbench datasets show that tensor index is both effective and efficient. Moreover, our method compares favorably with the state-of-the-art results.

In future study, we plan to further investigate the feasibility of multiple kinds of features in image retrieval, especially using the deep learning architectures [15, 40] which has shown superior performance in various fields.

Acknowledgements This work was supported by the National High Technology Research and Development Program of China (863 program) under Grant No. 2012AA011004 and the National Science and Technology Support Program under Grant No. 2013BAK02B04. This work was supported in part to Dr. Qi Tian by ARO grant W911NF-12-1-0057 and Faculty Research Awards by NEC Laboratories of America. This work was supported in part by National Science Foundation of China (NSFC) 61429201.

References

1. Arandjelovic, R., Zisserman, A.: Three things everyone should know to improve object retrieval. In: Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference, pp. 2911–2918. IEEE (2012)
2. Babenko, A., Lempitsky, V.: The inverted multi-index. In: Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference, pp. 3069–3076. IEEE (2012)
3. Bai, S., Wang, X., Yao, C., Bai, X.: Multiple stage residual model for accurate image classification. In: Computer Vision-ACCV 2012. Springer (2014)
4. Boix, X., Roig, G., Leistner, C., Van Gool, L.: Nested sparse quantization for efficient feature coding. In: Computer Vision-ECCV 2012, pp. 744–758. Springer (2012)
5. Cai, J., Liu, Q., Chen, F., Joshi, D., Tian, Q.: Scalable image search with multiple index tables. In: Proceedings of International Conference on Multimedia Retrieval, p. 407. ACM (2014)
6. Cai, Y., Tong, W., Yang, L., Hauptmann, A.G.: Constrained keypoint quantization: towards better bag-of-words model for large-scale multimedia retrieval. In: Proceedings of the 2nd ACM International Conference on Multimedia Retrieval, p. 16. ACM (2012)
7. van Gemert, J.C., Veenman, C.J., Smeulders, A.W., Geusebroek, J.M.: Visual word ambiguity. Pattern Anal. Mach. Intell. IEEE Trans. **32**(7), 1271–1283 (2010)
8. Huiskes, M.J., Thomee, B., Lew, M.S.: New trends and ideas in visual concept detection: the mir flickr retrieval evaluation initiative. In: Proceedings of the international conference on Multimedia information retrieval, pp. 527–536. ACM (2010)
9. Jégou, H., Chum, O.: Negative evidences and co-occurences in image retrieval: The benefit of pca and whitening. In: Computer Vision-ECCV 2012, pp. 774–787. Springer (2012)
10. Jegou, H., Douze, M., Schmid, C.: Hamming embedding and weak geometric consistency for large scale image search. In: Computer Vision-ECCV 2008, pp. 304–317. Springer (2008)
11. Jégou, H., Douze, M., Schmid, C.: On the burstiness of visual elements. In: Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on, pp. 1169–1176. IEEE (2009)

12. Jégou, H., Douze, M., Schmid, C.: Improving bag-of-features for large scale image search. Int. J. Comput. Vis. **87**(3), 316–336 (2010)
13. Jegou, H., Douze, M., Schmid, C.: Product quantization for nearest neighbor search. Pattern Anal. Mach. Intell. IEEE Trans. **33**(1), 117–128 (2011)
14. Jegou, H., Schmid, C., Harzallah, H., Verbeek, J.: Accurate image search using the contextual dissimilarity measure. ern Anal. Mach. Intell. IEEE Trans. **32**(1), 2–11 (2010)
15. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proc. IEEE **86**(11), 2278–2324 (1998)
16. Liu, J., Wang, S.: Salient region detection via simple local and global contrast representation. Neurocomputing **147**, 435–443 (2015)
17. Liu, S., Cui, P., Zhu, W., Yang, S., Tian, Q.: Social embedding image distance learning. In: Proceedings of the 20th ACM international conference on Multimedia (2014)
18. Liu, Z., Li, H., Zhou, W., Zhao, R., Tian, Q.: Contextual hashing for large-scale image search. Image Process. IEEE Trans. **23**(4), 1606–1614 (2014)
19. Liu, Z., Wang, S., Zheng, L., Tian, Q.: Visual reranking with improved image graph. In: ICASSP, pp. 6889–6893. IEEE (2014)
20. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. Int. J. Comput. Vis. **60**(2), 91–110 (2004)
21. Nister, D., Stewenius, H.: Scalable recognition with a vocabulary tree. In: Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference, vol. 2, pp. 2161–2168. IEEE (2006)
22. Niu, Z., Hua, G., Gao, X., Tian, Q.: Context aware topic model for scene recognition. In: Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference, pp. 2743–2750. IEEE (2012)
23. Philbin, J., Chum, O., Isard, M., Sivic, J., Zisserman, A.: Object retrieval with large vocabularies and fast spatial matching. In: Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference, pp. 1–8. IEEE (2007)
24. Philbin, J., Chum, O., Isard, M., Sivic, J., Zisserman, A.: Lost in quantization: Improving particular object retrieval in large scale image databases. In: Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference, pp. 1–8. IEEE (2008)
25. Qin, D., Wengert, C., Van Gool, L.: Query adaptive similarity for large scale object retrieval. In: Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference, pp. 1610–1617. IEEE (2013)
26. Shahbaz Khan, F., Anwer, R.M., van de Weijer, J., Bagdanov, A.D., Vanrell, M., Lopez, A.M.: Color attributes for object detection. In: Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference, pp. 3306–3313. IEEE (2012)
27. Shen, X., Lin, Z., Brandt, J., Avidan, S., Wu, Y.: Object retrieval and localization with spatially-constrained similarity measure and k-nn re-ranking. In: Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference, pp. 3013–3020. IEEE (2012)
28. Sivic, J., Zisserman, A.: Video google: A text retrieval approach to object matching in videos. In: Computer Vision, 2003. Proceedings. Ninth IEEE International Conference, pp. 1470–1477. IEEE (2003)
29. Su, B., Ding, X., Peng, L., Liu, C.: A novel baseline-independent feature set for arabic handwriting recognition. In: Document Analysis and Recognition (ICDAR), 2013 12th International Conference, pp. 1250–1254. IEEE (2013)
30. Su, Y., Fu, Y., Gao, X., Tian, Q.: Discriminant learning through multiple principal angles for visual recognition. Image Process. IEEE Trans. **21**(3), 1381–1390 (2012)
31. Su, Y., Tao, D., Li, X., Gao, X.: Texture representation in aam using gabor wavelet and local binary patterns. In: Systems, Man and Cybernetics, 2009. SMC 2009. IEEE International Conference, pp. 3274–3279. IEEE (2009)
32. Wang, D., Lu, H., Yang, M.H.: Online object tracking with sparse prototypes. Image Process. IEEE Trans. **22**(1), 314–325 (2013)
33. Wang, X., Yang, M., Cour, T., Zhu, S., Yu, K., Han, T.X.: Contextual weighting for vocabulary tree based image retrieval. In: Computer Vision (ICCV), 2011 IEEE International Conference, pp. 209–216. IEEE (2011)
34. Wang, Y., Liu, C., Ding, X.: Similar pattern discriminant analysis for improving chinese character recognition accuracy. In: Document Analysis and Recognition (ICDAR), 2013 12th International Conference, pp. 1056–1060. IEEE (2013)
35. Wengert, C., Douze, M., Jégou, H.: Bag-of-colors for improved image search. In: Proceedings of the 19th ACM international conference on Multimedia, pp. 1437–1440. ACM (2011)
36. Xia, Y., He, K., Wen, F., Sun, J.: Joint inverted index. In: ICCV (2013)
37. Xie, L., Tian, Q., Zhang, B.: Spatial pooling of heterogeneous features for image classification. Image Process. IEEE Trans. **23**(5), 1994–2008 (2013)
38. Yang, J., Yu, K., Gong, Y., Huang, T.: Linear spatial pyramid matching using sparse coding for image classification. In: Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference, pp. 1794–1801. IEEE (2009)
39. Yang, Y., Liu, J.: Exploring the large-scale tdoa feature space for speaker diarization. In: HCI International 2014-Posters Extended Abstracts, pp. 551–556. Springer (2014)
40. Yuan, H., Qian, Y., Zhao, J., Liu, J.: Mispronunciation detection with an optimized detection network and multi-layer perception based features. J. Tsinghua Univ. (Sci. Technol.) **4**, 027 (2012)
41. Zhang, S., Yang, M., Cour, T., Yu, K., Metaxas, D.N.: Query specific fusion for image retrieval. In: Computer Vision-ECCV 2012, pp. 660–673. Springer (2012)
42. Zhang, S., Yang, M., Wang, X., Lin, Y., Tian, Q.: Semantic-aware co-indexing for image retrieval. In: Computer Vision (ICCV), 2013 IEEE International Conference, pp. 1673–1680. IEEE (2013)
43. Zhang, Y., Jia, Z., Chen, T.: Image retrieval with geometry-preserving visual phrases. In: Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference, pp. 809–816. IEEE (2011)
44. Zheng, L., Wang, S.: Visual phraselet: Refining spatial constraints for large scale image search. Signal Process. Lett. IEEE **20**(4), 391–394 (2013)
45. Zheng, L., Wang, S., Tian, Q.: Coupled binary embedding for large-scale image retrieval. Image Process. IEEE Trans. **23**(8), 3368–3380 (2014)
46. Zheng, L., Wang, S., Tian, Q.: Lp-norm idf for scalable image retrieval. Image Process. IEEE Trans. **23**(8), 3604–3617 (2014)
47. Zheng, L., Wang, S., Zhou, W., Tian, Q.: Bayes merging of multiple vocabularies for scalable image retrieval. In: CVPR (2014)
48. Zhou, W., Lu, Y., Li, H., Tian, Q.: Scalar quantization for large scale image search. In: Proceedings of the 20th ACM international conference on Multimedia, pp. 169–178. ACM (2012)