Journal of Zhejiang University-SCIENCE C (Computers & Electronics)
ISSN 1869-1951 (Print); ISSN 1869-196X (Online)
www.zju.edu.cn/jzus; www.springerlink.com
E-mail: jzus@zju.edu.cn

Review:

High-dimensional indexing technologies for large scale content-based image retrieval: a review*

Lie-fu AI[1], Jun-qing YU‡[1,2], Yun-feng HE[1], Tao GUAN[1]

([1]*School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China*)

([2]*Center of Network and Computation, Huazhong University of Science and Technology, Wuhan 430074, China*)

E-mail: ailiefuhu@gmail.com; yjqing@hust.edu.cn; yfhe@hust.edu.cn; qd_gt@126.com

Received Dec. 29, 2012; Revision accepted May 21, 2013; Crosschecked June 6, 2013

Abstract: The boom of Internet and multimedia technology leads to the explosion of multimedia information, especially image, which has created an urgent need of quickly retrieving similar and interested images from huge image collections. The content-based high-dimensional indexing mechanism holds the key to achieving this goal by efficiently organizing the content of images and storing them in computer memory. In the past decades, many important developments in high-dimensional image indexing technologies have occurred to cope with the 'curse of dimensionality'. The high-dimensional indexing mechanisms can mainly be divided into three categories: tree-based index, hashing-based index, and visual words based inverted index. In this paper we review the technologies with respect to these three categories of mechanisms, and make several recommendations for future research issues.

Key words: Tree-based index, Hashing-based index, Bag-of-features (BOF), Descriptor encoding, Inverted index
doi:10.1631/jzus.CIDE1304 **Document code:** A **CLC number:** TP391.7

1 Introduction

The rapid development of Internet and multimedia technologies leads to the explosion of multimedia information, especially picture and image, which, however, has placed people in an awkward situation where they can hardly get those favorite images with accuracy and efficiency from a vast amount of images unless those images are efficiently organized for browsing, searching, and retrieval. Image retrieval has been a very active research area since the 1970s. There are two research directions with different emphases: data management and computer vision. The former is based on the textual description of images and the latter on the visual content of images.

Text-based image retrieval can be traced back to the late 1970s. A very popular framework of image retrieval was then to first annotate all the images in a database by text and use text-based management systems to perform image retrieval (Rui and Huang, 1999). Along this research direction, many approaches, such as data modeling, multi-dimensional indexing structure, and query evaluation, have been proposed. However, the huge image collection (more than millions or billions) usually makes manual image annotation labor-consuming. Moreover, for the same image content, different people may perceive it differently due to its rich content. Perception subjectivity and annotation impreciseness may cause unrecoverable mismatches in later retrieval.

In the 1990s, content-based image retrieval was proposed to overcome those difficulties that text-based image retrieval has long been confronting. As a result, manual annotation becomes quite uncalled-for since images are indexed by their own visual features, such as color, GIST, and scale-invariant feature

‡ Corresponding author

* Project supported by the National Natural Science Foundation of China (Nos. 61173114, 61202300, and 61272202) and the Guangdong Provincial Research Project (No. 2011B090400251)

transform (SIFT). Up to now, many technologies have been developed for content-based image retrieval, for research and commercial purposes. With respect to a huge amount of image collections, there are three fundamental bases for content-based image retrieval: visual feature extraction, high-dimensional indexing mechanism, and retrieval system design. To retrieve images quickly, storing a huge amount of visual images in computer memory has become a tendency. Thus, a high-dimensional indexing mechanism by which the images are organized efficiently is the core of achieving fast and accurate retrieval.

In this paper, we devote our efforts primarily to reviewing high-dimensional indexing technologies in large scale content-based image retrieval. The high-dimensional indexing technologies can mainly be divided into three categories: tree-based index, hashing-based index, and visual words based inverted index.

2 Tree-based index

As a conventional indexing technique, the tree-based indexing structure successively partitions the data space from coarse to fine and forms a tree of hierarchical structure. The non-leaf nodes are the directory nodes where the information of the data space is stored, and the leaf nodes are the data objects storing the information that needs to be indexed.

2.1 Classification of tree-based index

According to the different similarity measure metrics, tree-based indexing structures can be classified into indexing structures in the vector space and the metric space (Fig. 1) (Zhou, 2011). The former adopts the Euclidian distance to compute the similarity between features, such as R-tree (Guttman, 1984) and KD-tree (Bentley, 1975; Robinson, 1981). The later adopts the non-Euclidian distance to compute the similarity between features, such as M-tree (Ciaccia *et al.*, 1997) and BK-tree (Burkhard and Keller, 1973).

According to the different shapes obtained by partitioning the data space at a specified dimension, the tree-based indexing structures can be divided into three categories: TV-tree (Lin *et al.*, 1994) on behalf of hyper-rectangle, KD-tree on behalf of hyper-sphere and SR-tree (Katayama and Satoh, 1997) on behalf of a hybrid space.

2.2 KD-tree

KD-tree was first introduced as a generalization of a balanced binary tree to index high-dimensional data (Bentley, 1975). Later on, Friedman *et al.* (1977) proposed an optimized KD-tree with a theoretically logarithmic search time. Although KD-tree is efficient for low dimensional data, its efficiency reduces

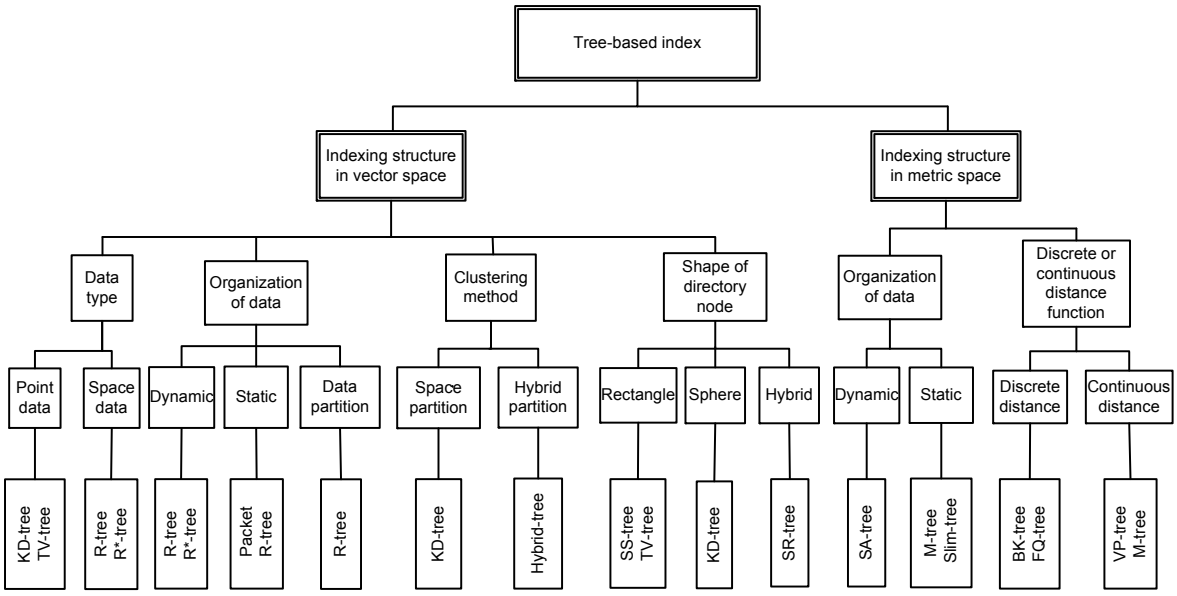

Fig. 1 Classification of tree-based indexing structures (Zhou, 2011)

dramatically when the dimension of data exceeds 20. The search time may be approximately equal to that of a linear search, suffering from 'curse of dimensionality'. This is because with high dimensional data, a KD-tree usually takes a lot of time to backtrack through the tree to find the optimal solution. By limiting the amount of backtracking, the time efficiency can be improved at the sacrifice of the certainty of finding the absolute minimum. Based on this approach, to increase the probability of success and keep backtracking within reasonable limits, two similar approximated indexing methods (Arya and Mount, 1993; Beis and Lowe, 1997) are proposed, i.e., best-bin-first search and priority search.

By creating multiple KD-trees from the same feature set and simultaneously searching among these trees, Anan and Hartley (2008) improved the KD-tree indexing structure to index 500 000 SIFT descriptors. It has been demonstrated that multiple KD-trees significantly outperform a single KD-tree. Moreover, if principal component analysis (PCA) is used to align the principal axes of the feature with coordinate axes, the performance of multiple KD-trees will be further increased.

2.3 R-tree

Pushed by the urgent demand of spatial indexing from the geographic information system (GIS) and computer aided design (CAD) systems, Guttman (1984) proposed the R-tree indexing structure. The variants of R-tree include R^*-tree (Beckman *et al.*, 1990), R^+-tree (Sellis *et al.*, 1987), SR-tree (Katayama and Satoh, 1997), and X-tree (Berchtold *et al.*, 1996). Although R-tree and its variants can perform efficiently in low dimensions, their performance deteriorates rapidly as the dimension increases due to the 'curse of dimensionality'.

2.4 M-tree

M-tree (Ciaccia *et al.*, 1997) is the first dynamic indexing structure that provides good performance in the metric space. The variants of M-tree include Slim-tree (Caetano *et al.*, 2000), PM-tree (Skopal, 2004), M*-tree (Skopal and Hoksza, 2007), and NM-tree (Skopal and Lokoc, 2008; 2009).

The M-tree index is a hierarchical structure, where some of the data objects are selected as centers of sphere-shaped regions, and the remaining objects are partitioned among the regions to build up a balanced and compact hierarchy. Similar to the construction of the B-tree, each region is recursively indexed in a bottom-up way. M^*-tree (Skopal and Hoksza, 2007) is an extension of M-tree, where each node is additionally equipped by a nearest-neighbor graph (NN-graph). The NN-graph is used to filter out the non-relevant objects. The motivation for utilizing NN-graphs is related to the advantages of methods using global pivots. M^*-tree can reduce the computation costs, but the I/O costs are the same as in M-tree.

Slim-tree (Caetano *et al.*, 2000) introduces a new node splitting algorithm based on the minimum spanning tree (MST), so that the CPU costs during insertion can be reduced. Besides, Slim-tree develops a slim-down algorithm to improve the query performance.

By combining the hierarchy of M-tree with a set of p global pivots, PM-tree (Skopal, 2004) improves the search efficiency. With the global pivot, PM-tree maps the metric data into a space of dimensionality p.

Skopal and Lokoc (2008; 2009) proposed M-tree-based NM-tree, which can provide exact and approximate search in metric and non-metric spaces by combining the TriGen algorithm (Skopal, 2007). It has been demonstrated that a single NM-tree index can search as fast as multiple M-tree indexes.

2.5 EHD-tree

To support fast k-nearest-neighbor (kNN) search in a high-dimensional space, Zhuang *et al.* (2008) proposed a high-dimensional indexing scheme based on the technique of symmetrical encoding-based hybrid distance tree, called EHD-tree. In an EHD-tree, all the feature points are first grouped by the K-means algorithm. Then, the uniform ID (UID) number of each feature point is obtained by a dual-distance-driven encoding scheme, where start-distance (Fonseca and Jorge, 2003) and centroid-distance (Jagadish *et al.*, 2005) are used to partition each cluster sphere. Finally, the uniform index key of each feature point is obtained through linearly combining its UID with the centroid-distance, and indexed by a partition-based B^+-tree. With the EHD-tree indexing scheme, the kNN search of a query feature point in a high-dimensional space can be transformed into search in a single-dimensional space.

3 Hashing-based Index

Hashing-based high-dimensional indexing structures, which project feature data from high dimensions to low dimensions via hash functions, have been widely used in recent years. It comprises mainly two categories: E2LSH on behalf of projecting feature points into a low-dimensional Euclidian space and spectral hashing on behalf of mapping close data points in a Euclidian space to similar binary codes in a low dimensional Hamming space. The former uses the Euclidian distance as the similarity measure metric, while the latter uses the Hamming distance as the similarity measure metric.

3.1 Locality sensitive hashing

Locality sensitive hashing (LSH) was first proposed along with rigorous mathematical description (Indyk and Motwani, 1998). Gionis *et al.* (1999) implemented LSH in the Hamming space and evaluated the performance with low level features in high dimensions. LSH has been proved to be sub-linearly dependent on the data size. The core of LSH is to construct a family of hashing functions so as to ensure that, for each hashing function, the probability of collision is much higher for data objects that are close to each other than for those which are far apart.

According to LSH, feature points can be indexed by hashing the feature data into the buckets in hashing tables. For a query feature point, LSH hashes it into a bucket and uses the feature points in the bucket as the candidate set of the results, which will be ranked using the corresponding distance measure metric. Moreover, LSH can be naturally extended to the dynamic setting due to its hash-based scheme, i.e., when insertion and deletion operations need to be supported. LSH usually needs tens or hundreds of hashing tables to achieve good retrieval accuracy for high-dimensional datasets. The kernelized version of LSH is called kernelized LSH (Kulis and Grauman, 2009).

Although the LSH scheme has been used in numerous applications (Gionis *et al.*, 1999; Buhler, 2002; Georgescu *et al.*, 2003), it is fast and simple only when the input feature points lie in the binary Hamming space $\{0, 1\}^d$. When the LSH algorithm needs to be extended to the l_2 norm, the input feature points are first embedded from the l_2 space into the l_1 space, and then embedded from the l_1 space into the Hamming space. However, it increases the query time or error by a large factor and complicates the algorithm.

Based on p-stable distributions, Datar *et al.* (2004) proposed exact Euclidian locality sensitive hashing (E2LSH). Unlike LSH, the E2LSH algorithm works directly on the data points in the Euclidian space without embeddings. Currently, there is an open-source and available software package. Up to now, E2LSH has been successfully used for indexing local descriptors (Ke *et al*., 2004) and 3D objects (Matei *et al.*, 2006). E2LSH is efficient on those high-dimensional and dense feature points. Similar to LSH, E2LSH needs a large number of hash tables to achieve good retrieval accuracy. Since the Euclidian distance metric is used to rank the candidate set of query results, the feature points need to be kept in computer memory to reduce the time of computing Euclidian distances. Therefore, the dataset that E2LSH can deal with is restricted in size.

Andoni and Indyk (2006; 2008) presented a near-optimal hashing algorithm for c-approximate nearest neighbor search in the Euclidian space, significantly improving the earlier running time of E2LSH. Moreover, the query time and the memory requirement almost match the lower bounds of the hashing-based algorithm obtained in Motwani *et al.* (2008).

To reduce the memory requirements of E2LSH, Panigraphy (2006) proposed entropy-based LSH. Entropy-based LSH randomly generates new objects near the query object and takes them as new query objects in addition to the original query object. The union of all the query results is returned as the candidate set. The intention of entropy-based LSH is to trade time for memory requirements. The construction process of indexing structure in entropy-based LSH is similar to that in E2LSH, while entropy-based LSH expands one query object q to a group of query objects at distance D_q from q, where D_q denotes the distance from the query object to its nearest neighbor. However, D_q is difficult to choose in a data-dependent way. If D_q is too small, expansive queries may not produce the desired number of objects in the candidate set. If D_q is too large, it would

require many expansive queries to achieve good retrieval accuracy.

Lv *et al.* (2007) proposed multi-probe LSH to overcome the drawbacks of entropy-based LSH. For each hash table, multi-probe LSH uses a derived probing sequence to look up multiple buckets that have a high probability of containing the nearest neighbor of the query object. Two probing schemes, stepwise probing sequence and query-directed probing sequence, have been developed. The query-directed probing sequence first visits the buckets with a high probability of success, while the stepwise probing sequence first visits the neighbor buckets. Therefore, the query-directed probing sequence is far superior to the stepwise probing sequence. It has been demonstrated that multi-probe LSH is much more space efficient than entropy-based LSH and E2LSH in the context of achieving comparable accuracy and search time.

Another method of improving E2LSH introduces two expansion strategies, intra-expansion and inter-expansion (Kuo *et al.*, 2009). Intra-expansion uses existing query features to obtain more similar features as new query features. Inter-expansion obtains new query features which are not presented in the query image but mined from initial search results. Both intra- and inter-expansion can improve the retrieval performance significantly and reduce the number of hash tables. Moreover, the retrieval performance can be further improved when they are combined iteratively.

To further improve E2LSH, Jegou *et al.* (2008b) proposed a query-adaptive LSH (QA-LSH), which performs an on-line selection of the most appropriate hash functions from a pool of hash functions. A hash function relevance criterion, measuring the expected accuracy of a given hash function, is defined and used for on-line selection of hash functions. In addition, hyper-diamond E_8 lattice (Conway and Sloane, 1982) is used for geometric hashing instead of 1D random projection in E2LSH. Compared to E2LSH, QA-LSH is able to arrive at a better compromise between speed and accuracy by improving hash function construction and introducing a strategy of on-line selection of hash functions.

Pauleve *et al.* (2010) proposed KLSH by using a *K*-means clustering method to construct hash functions. The hash functions assign each feature point to the closest centroid according to the corresponding codebook. *K*-means is an unsupervised training algorithm with good quality, but it may take a long time to train codebook when *K* is large. So, hierarchical *K*-means (Nister and Stewenius, 2006) and approximate *K*-means (Philbin *et al.*, 2007) can be used to replace *K*-means to reduce the training time. Deriving from the ideas of multi-probe LSH and QA-LSH, Pauleve *et al.* (2010) also developed multi-probe KLSH and query-adaptive KLSH. It has been demonstrated that KLSH can obtain very high recall by using a limited number of hash functions.

3.2 Hashing coding

E2LSH-related indexes need to store initial image features in computer memory for ranking the candidate set, which restricts E2LSH in retrieval speed and database size. In recent years, some research communities have been focusing on encoding features by hashing projection, so that millions of feature codes can be stored in computer memory and images that are similar to the query image can be quickly retrieved. The intension of hashing coding is that similar features have the same or similar binary codes. The general similarity metric is the Hamming distance, which is much faster than the Euclidian distance in similarity computing.

The original LSH (Gionis *et al.*, 1999) can be considered as the first hashing coding method. To apply LSH to the high-dimensional kernelized data when underlying feature embedding for the kernel is unknown, Kulis and Grauman (2009) presented a kernelized LSH (KLSH), which performs fast similarity search over arbitrary kernel functions. Like LSH, the hash functions of KLSH involve computing random projections; unlike LSH, however, these random projections are constructed using only the kernel function and a sparse set of samples from the dataset itself.

Semantic hashing (Salakhutdinov and Hinton, 2007a; 2009), which uses an auto-encoder with several hidden layers, is a typical work on hashing coding. The architecture of semantic hashing can be considered as a restricted Boltzmann machine (RBM) in which there are only connections between units in layers and units not within layers. For example, to learn 32 bits, the middle layer of the auto-encoder has 32 hidden units. Salakhutdinov and Hinton (2007b)

used multiple stacked RBMs to learn a non-linear mapping between the input vector and code bits. Back-propagation using a neighborhood component analysis (NCA) objective function is used to refine the weights in the RBM, to preserve the neighborhood structure of the input space.

Torralba *et al.* (2008) used RBMs and boosting to learn binary codes for encoding a database of millions of images. This method not only significantly reduces memory requirements, but also significantly improves the retrieval speed over LSH. Based on Torralba *et al.* (2008), Weiss *et al.* (2009) proposed spectral hashing (SH), assuming that features follow multi-dimensional uniform distribution. SH transforms the problem of encoding features into the problem of graph partitioning, where the NP-hard problem arises. Besides, SH designs an efficient eigenvector solution to graph partition in order to generate the best binary code for indexing features. To avoid the NP-hard problem, spectral relaxation is realized in SH to obtain a number of eigenvectors with the minimal eigenvalue from the Laplacian matrix (Belkin and Niyogi, 2001) after the PCA step. SH outperforms the method proposed by Torralba *et al.* (2008); moreover, it is easier to implement.

Based on SH, Shao *et al.* (2012) proposed sparse spectral hashing (SSH), where sparse principal component analysis (sparse PCA) and boosting similarity sensitive coding are introduced into SH. SSH formulates the problem of binary coding as thresholding a subset of eigenvectors of the Laplacian graph by constraining the number of nonzero features. Convex relaxation and eigenfunction learning are conducted in SSH to make coding optimal and effective to the features outside the training dataset. Another improvement on SH is hyper-graph spectral hashing (HSH) (Zhuang *et al.*, 2011) by which a hashing algorithm is designed for fast retrieval of large scale social images. To represent the complex and high-order relationships among social content of images, a hyper-graph was modeled (Bu *et al.*, 2010), where an edge can connect more than two vertices. Then, to accelerate the similarity search of social images, SH was extended to hyper-graph by mapping related vertices to similar binary codes within a short Hamming distance. To encode the novel image features outside the training dataset, supervised self-taught hashing was extended to HSH for learning the hash functions (Zhang *et al.*, 2010).

By taking advantage of supervised information, supervised hashing methods such as linear discriminant analysis hashing (LDAH) (Strecha *et al.*, 2012), binary reconstructive embeddings (BRE) (Kulis and Darrell, 2009), and minimal loss hashing (MLH) (Norouzi and Fleet, 2011) have been proved to be able to obtain better retrieval accuracy than unsupervised hashing methods. However, supervised hashing is usually slower than unsupervised hashing in the training procedure. Besides, they are prone to over fitting when labeled data is small or noisy. To improve the time efficiency of supervised hashing, Liu W *et al.* (2012) proposed a kernel-based supervised hashing (KSH), which requires only a limited amount of supervised information and a feasible training cost in achieving high quality hashing. KSH maps image features to compact binary codes where the Hamming distances between similar features are minimized and those between dissimilar features maximized. Different from the supervised methods in Kulis and Darrell (2009), KSH uses the algebraic equivalence between optimizing the code inner products and the Hamming distances, so that KSH can sequentially train the hash functions one bit at a time.

Wang *et al.* (2010) proposed a semi-supervised hashing (SSH), minimizing empirical errors on the labeled data while maximizing variance and independence of hash bits over the labeled and unlabeled data. Afterwards, Wang *et al.* (2012) developed three different SSH methods: orthogonal hashing, non-orthogonal hashing, and sequential hashing. Particularly, sequential hashing generates robust codes in which each hash function is designed to correct the errors made by the previous ones.

For multi-labeled data, Liu X *et al.* (2012) proposed a boosted shared hashing (BSH) to enhance the hashing efficiency. BSH is a combination of query-adaptive hashing (Mu *et al.*, 2011) and shared subspace learning for multi-label data (Torralba *et al.*, 2004). BSH improves the hashing efficiency by making each hash function target at and share a portion of labels instead of all labels.

Different from conventional hyper-plane-based hash functions, spherical hashing (Heo *et al.*, 2012) uses hyper-sphere-based hash functions to encode image features. Since hyper-sphere provides much

stronger power than hyper-plane in defining a more tightly closed region in the original feature space, spherical hashing maps more spatially coherent features to the same or similar binary codes compared to hyper-plane-based hash functions. Spherical hashing uses hyper-spheres to partition the feature space and each hyper-sphere is associated with a hash function. Similar to the methods in He *et al.* (2011) and Joly and Buisson (2011), spherical hashing uses an iterative optimization process to achieve balanced partitioning of features for each hash function and independence between hash functions. Besides, a spherical hamming distance is designed to improve the discrimination of the Hamming distance. There is another similar spherical LSH (Terasawa and Tanaka, 2007), which is a specialized technique for all the features that are constrained to lie on the surface of the unit hyper-sphere.

If the binary codes of image features are used directly as indexing keys and inserted into the buckets in hash tables, the retrieval will be inefficient when the binary codes exceed 32 bits. To overcome this drawback, Norouzi *et al.* (2012) proposed a multi-index hashing scheme by building multiple hash tables on binary code substrings. Multi-index hashing enables exact kNN search in the Hamming space. The binary code of an image feature is divided into m disjoint substrings, and each substring is separately indexed into a bucket. Given a query binary code, it is also indexed m times into m buckets, and those binary codes in the m buckets are used as a candidate set of query results. If the binary codes are distributed uniformly, multi-index hashing has sub-linear run-time and is more than 100 times faster than a linear scan baseline.

LSH works well on dense feature vectors, while the bag-of-features (BOF) image descriptor is a sparse vector. In view of this problem, Zhang *et al.* (2009) used an image decomposition model (IDM) to decompose BOF image representation into two vectors, vector of topic distribution and vector of image-specific distribution. As an extension of the LDA model (Blei *et al.*, 2003), IDM was first proposed in Chemudugunta *et al.* (2006) to detect important words of a document. The vector of topic distribution is low-dimensional and dense, and is indexed by LSH (Andoni and Indyk, 2008). For a query image, the candidate set of results is first attained by the indexing structure, and then the vector of topic distribution and the vector of image-specific distribution are combined together for computing the similarity between the query image and images in the candidate set.

4 Visual words based inverted index

The first visual words based inverted index, called bag-of-features (BOF) or bag-of-words (BOW), was introduced from text retrieval systems. Given an image, the local features, such as SIFT, are first extracted, and then quantized into the closest visual words ('codebook'), which are pre-learned on a training dataset. Finally, a high-dimensional vector, called the BOF descriptor, is generated to represent the image. The BOF descriptor is indexed by an inverted file which has an entry for each visual word followed by a list of all the images in which the visual word occurs. The similarity between BOF representations is usually measured by the Euclidian distance or cosine distance.

An image descriptor is essential to content-based image indexing structure and image retrieval; thus, it is of great importance to produce an image descriptor quickly and accurately. Besides, there are another two keys for large scale content-based image retrieval: an efficient inverted indexing structure for quick search, and methods of compressing and encoding image descriptors for storing a huge number of descriptors in computer memory. We will review the technologies in these three aspects respectively in the following.

4.1 Visual words based image descriptor

The visual words based image descriptor was first introduced from text retrieval to content-based image retrieval by Sivic and Zisserman (2003), who proposed the BOF descriptor using the SIFT local feature (Lowe, 1999). The BOF descriptor is a high-dimensional vector of visual word occurrences, weighted by using term frequency inverse document frequency (tf-idf). There are another two kinds of BOF descriptors. One is the frequency BOF vector where the component denotes the frequency of a given visual word occurring in the image, and the other is the binary BOF vector where the component

denotes whether a given visual word occurs in the image. To achieve good accuracy and efficiency, the dimension of BOF should be high, usually up to a million. Thus, it is important to quickly and accurately calculate the BOF descriptor.

4.1.1 Constructing a large visual vocabulary

The high dimensionality of the BOF descriptor brings forth the necessity of learning numerous visual words. Standard BOF (Sivic and Zisserman, 2003) adopts a flat *K*-means algorithm to train visual words. Although *K*-means can obtain good quality, it will be time inefficient if the visual vocabulary is too large. By using a hierarchical *K*-means clustering algorithm, Nister and Stewenius (2006) presented a vocabulary tree, enabling a large visual vocabulary to be trained efficiently. Philbin *et al.* (2007) proposed the approximate *K*-means (AKM) algorithm, which uses a forest of eight randomized KD-trees (Lepetit *et al.*, 2005) built over the cluster centers at the beginning of each iteration. Instead of the exact nearest neighbor computation used in flat *K*-means, AKM uses an approximate nearest neighbor method to increase the speed. Robust approximated *K*-means (RAKM) (Li *et al.*, 2010) is an extension of AKM where the nearest neighbor in one iteration is re-used in the next one. In this context, less effort is spent for new neighbor search. To further improve RAKM, Avrithis and Kalantidis (2012) proposed approximate Gaussian mixture (AGM), which combines the flexibility of Gaussian mixtures with the scaling properties needed, to construct large vocabularies for image retrieval. AGM is a variant of expectation-maximization (EM) that can converge rapidly while dynamically estimating the number of components. Similar to RAKM, approximate nearest neighbor search is employed to speed up the E-step in EM (Biship, 2006). It has been demonstrated that AGM stands in the same line as RAKM in terms of speed while improving RAKM in performance.

To reduce the memory storage requirement and increase the time efficiency of constructing a visual vocabulary, a random locality sensitive vocabulary (RLSV) scheme was proposed by Mu *et al.* (2010). Combining LSH (Andoni and Indyk, 2008) with the random forest (RF), RLSV generates a vocabulary by applying a sequence of random linear bipartitions. To reduce the inherent randomness of LSH, multiple random vocabularies are created independently using the same method, and the final inter-sample distance or similarity is based on the consensus of all vocabularies. Compared with *K*-means related methods, RLSV does not take clustering or training efforts. Besides, Mu *et al.* (2010) developed the kernel version of RLSV, called kernelized RLSV, and the supervised version of RLSV, called discriminative RLSV.

Since the BOF descriptor uses only low-level visual features, it is very difficult to retrieve object images of different viewpoints or lighting conditions. Some researchers considered more information when generating visual words, such as visual constraints (Philbin *et al.*, 2010) and textual information (Wu *et al.*, 2009). However, extra manual information is usually required during supervised learning, which may be a burden for large scale image databases.

Unsupervised auxiliary visual words (AVW) mechanism (Kuo *et al.*, 2010) observes that an image is associated with a BOF descriptor and a textual description. AVW first adopts the algorithm in Elsayed *et al.* (2008) to construct a graph on BOF descriptors and textual description respectively. Then, affinity propagation (AP) (Frey and Dueck, 2007) is used to cluster images in these graphs. Finally, by feature propagation and selection, AVW automatically discovers auxiliary visual words in textual and visual image graphs. By AVW, the discrimination of the BOF descriptor is improved, and more various kinds of similar images can be retrieved.

4.1.2 Assigning local features to visual words

To reduce the quantization error of calculating the BOF descriptor, Cai *et al.* (2012) proposed constrained key-point quantization. When the Euclidian distances from a local feature to all the visual words are greater than the predefined threshold, the local feature is considered as an outlier and not assigned to any visual word. Although this approach is simple, it has been demonstrated that the attained BOF descriptor is superior to the standard BOF descriptor.

In general, hard assignment is used to assign a local feature to its nearest visual word when calculating the BOF descriptor. Since the hard assignment scheme does not consider codeword uncertainty or codeword plausibility, large quantization errors are usually produced when assigning local features to

their neighbor visual words. To improve the image descriptor, some research communities put emphasis on soft assignment, i.e., assigning a local feature to multiple neighbor visual words.

To alleviate the two drawbacks of hard assignment, the approach of kernel-based soft assignment (van Gemert *et al.*, 2008; 2010) first computes the weight between each local feature and every visual word, and then assigns the local features to the visual words if the corresponding weight is larger than a preset threshold. To reduce quantization errors, by descriptor space soft assignment (Philbin *et al.*, 2008) each local feature is assigned to the k nearest visual words. The weights between the local feature and neighbor visual words are computed based on the Gaussian mixture models (GMM) (Biship, 2006). Similarly, given a local feature, the localized soft assignment scheme (Liu *et al.*, 2011) considers only its k nearest visual words and sets its distances to the remaining visual words to infinity. The method of computing weights is similar to that in Philbin *et al.* (2008), but the sum of the weights between a local feature and its neighbor visual words is equal to 1. The spherical soft assignment scheme (Ai *et al.*, 2012) adaptively assigns each local feature to neighbor visual words according to the preset radius. A visual word signifies a cluster which is defined as a hyper-sphere in the feature space. The radius is defined as the distance from the cluster center to the farthest feature in the cluster. According to spherical soft assignment, a local feature is assigned to the visual word whose distance to the local feature is smaller than the corresponding radius.

4.1.3 Other visual words based image descriptors

There are some other image descriptors built on the BOF descriptor but superior to it. Based on BOF and the Fisher kernel (Perronnin and Dance, 2007), Jegou *et al.* (2010a) proposed vectors of locally aggregated descriptors (VLADs). VLAD creates a compact global image representation by aggregating vector residuals of local SIFT (Lowe, 2004) features quantized to a small set of codebooks. VLAD commonly consists of SIFT feature extraction, visual codebook creation, feature assignment, vector residuals aggregation, and residual vectors concentration. VLAD outperforms the BOF descriptor with lower dimensionality. Jegou *et al.* (2012) further improved VLAD based on the Fisher kernel (Perronnin and Dance, 2007), which compresses the kernel vector of high dimensionality and density. To remove outlier features lying close to the boundary between two visual words, Chen *et al.* (2011) discarded all the features whose distances to their nearest visual words are above 90% on the distribution of distances. Thus, the discrimination of VLAD is improved. Hard assignment is used in calculating VLAD, where each local feature is assigned to the nearest visual word.

Torresani *et al.* (2010) proposed an image descriptor of attributes which combines a color GIST descriptor, pyramid of histograms of oriented gradients (PHOG), a PHOG descriptor with oriented edges, a pyramid of self-similarity descriptor, and BOF. It has been proved to be better than the standard BOF. Douze *et al.* (2011) combined an image descriptor of attributes with Fisher vectors (Perronnin and Dance, 2007) to further improve the discrimination of the image descriptor.

Based on BOF, Wengert *et al.* (2011) proposed a bag-of-color (BOC), an image descriptor in the CIE-Lab space. BOC splits an image into m blocks and the most occurring color of the 3D vector is found for each block. Then, like BOF, each color is allocated to the nearest color word pre-learned on a training dataset. Finally, a BOC image descriptor about the occurrence frequency of the most occurring colors is produced. BOC has much lower dimensionality than BOF, and it is superior to BOF based on either SIFT or color SIFT descriptors.

4.2 Compressing and encoding descriptors

Compared with storing many SIFT local descriptors of images, BOF-related visual words based image descriptors can significantly reduce storage requirement. The sparser the BOF, the more efficient it is. Such a BOF approach allows a single machine to handle several million images in computer memory. However, it is hard to scale up to billions of images.

Hashing coding reviewed above can be considered as efficient methods of compressing and encoding descriptors of dense vectors.

Compared with the BOF descriptor, Jegou *et al.* (2008a; 2009b; 2010b) proposed a more precise image descriptor by compressing the BOF descriptor

based on Hamming embedding (HE) and weak geometric consistency (WGC) constraints. HE projects descriptors from the Euclidian space to the Hamming space where the Hamming distance between a descriptor and its nearest neighbor in the Euclidian space is also the smallest, and encodes the descriptor with binary code. HE provides binary signatures that refine the matching based on visual words. Given a query descriptor, WGC is used to filter the matching descriptors that are not consistent in terms of angle and scale.

Based on HE, an approximate descriptor of BOF, called miniBOF (Jegou *et al.*, 2009a), was proposed by projecting the BOF descriptor onto a set of predefined sparse projection functions. First BOF is projected into m low-dimensional descriptors by using m sparse projection matrices, and then HE is used to encode these low-dimensional descriptors respectively. The m miniBOFs are separately indexed by inverted files. For an image, the memory usage of a BOF descriptor would be typically 10 KB, while miniBOF needs only 320 bytes when the BOF descriptor is replaced with 16 miniBOFs.

Derived from Dong *et al.* (2008) and Gordo and Perronnin (2011) who used asymmetric distances in the context of LSH, Jain *et al.* (2011) proposed an asymmetric hamming embedding (AHE) scheme to improve HE for large scale image retrieval. Instead of the HM distance used in HE, AHE uses a vector-to-hyper-plane distance to compute the distance between two descriptors. AHE improves HE with no additional cost in terms of memory but slightly higher complexity.

Descriptor quantization can also be used to encode descriptors to reduce storage requirements. A survey of descriptor quantization can be seen in Gray and Neuhoff (1998). In the following, we will review some typical encoding technologies by quantization for large scale content-based image retrieval in recent years.

Product quantization (PQ) (Jegou *et al.*, 2009c; 2011) partitions a descriptor vector into several distinct sub-vectors and encodes the sub-vectors separately. In the feature space associated with a sub-vector, a quantizer is trained by K-means. If k bits are allocated to the quantizer, there will be 2^k centroids. Given a descriptor divided into m sub-vectors, each sub-vector is quantized by the corresponding quantizer. The quantizers assign each sub-vector to its nearest centroid whose code is used to identify the sub-vector. Then, the m codes are concentrated together to form the global code of the descriptor (km bits). Compared with the global quantizer, PQ uses much fewer centroids, and thus the memory requirement is significantly reduced. A similar research to PQ is the lattice quantizer (Tuytelaars and Schmid, 2007), which allocates a constant number of bits per descriptor component.

PQ can attain good performance based on the assumption that the components of descriptor vectors are statistically independent of each other. However, it is not practical enough for real data. To address this problem, Brandt (2010) proposed a quantization technique that combines transform coding with PQ. According to the eigenvalue in each dimension, transform coding is used to allocate a corresponding number of bits for each component of the descriptor at the training step. The components allocated 0 bit are dropped. According to the number of bits allocated, a 1D quantizer is trained by K-means for each remaining component. PQ is used to encode the descriptors in the database.

To provide efficient quantization for both structured and unstructured descriptors, Chen *et al.* (2010) developed residual vector quantization (RVQ) to encode descriptors. RVQ is a technique to reduce the quantization error with several low complex quantizers. RVQ approximates the quantization error by another quantizer instead of discarding it. RVQ consists of several stage-quantizers, each having the corresponding stage-codebook trained by K-means. These stage-quantizers are connected sequentially. Each stage-quantizer approximates the residual vector in the preceding stage by one of centroids in the stage-codebook and generates a new residual vector for the succeeding quantization stage. The more the stages it has, the more accurate the code it generates. Taking an RVQ of two stages as an example, a descriptor is quantized in the 1-stage quantizer, and the residual vector between the descriptor and its nearest centroid is quantized in the 2-stage quantizer. Consequently, two codes are concentrated together to form the global code of the descriptor. Since the whole vector of the descriptor is used to train the quantizer, the time complexity is a little larger than that of PQ.

4.3 Visual words based inverted indexing structures

In the standard visual words based inverted indexing structure (Sivic and Zisserman, 2003), each visual word is associated with an inverted list, in which the image identification and the frequency of the visual word occurring in the image are stored. Given a query image, after calculating the BOF descriptor, all the inverted lists, associated with the visual words occurring in the query image, are searched. If there are 1000 visual words occurring in a query image, 1000 inverted lists need to be searched. Also, the image will be stored in 1000 inverted list when indexing it. In view of this scenario, to efficiently construct indexing structure, some researchers used a coarse quantizer which includes a small number of global centroids. Of course, the smaller the number of inverted lists, the larger the number of descriptors in the candidate set of query results.

Jegou *et al.* (2011) proposed an inverted file system, IVFADC, which combines an inverted structure with asymmetric distance computation (ADC). By *K*-means, IVFADC trains a coarse quantizer of k centroids. Each centroid is associated with an inverted list in the indexing structure. Every descriptor is allocated to the nearest centroid, and the residual vector between the descriptor and its nearest centroid is quantized and encoded by PQ. The descriptor identification information and its codes are stored in the corresponding inverted list according to its nearest centroid. Given a query descriptor, after quantizing it to the nearest centroid or r nearest centroids, all the descriptors in the corresponding inverted list are used as the candidate set of query results. The ADC method is used to quickly compute the distances from the query to the descriptors in the candidate set, and a ranking procedure is followed.

RVQ-based indexing structure (Chen *et al.*, 2010) directly uses the quantizers of the first L stages as the coarse quantizer. When a stage quantizer includes k centroids, k^L inverted lists can be constructed. This is more efficient than PQ. According to the codes encoded by the quantizers of the first L stages, the descriptor identification information together with its RVQ codes are inserted into the corresponding inverted list. Given a query descriptor, the candidate set of query results is obtained according to its codes of the first L stages. Then similar to PQ, the ADC method is used to quickly compute the distance from the query to the descriptor in the candidate set, and a ranking procedure is followed.

Similar to the construction of RVQ-based indexing structure, Babenko and Lempitsky (2012) proposed an inverted multi-index, which is a multi-dimensional table based on PQ. Typically, the inverted multi-index partitions descriptors into two sub-vectors. PQ is separately adopted to train two quantizers for the two sub-vectors. The centroid pairs from the two quantizers form the indexing structure of a 2D indexing table. Given a descriptor, the pair of quantization codes by PQ is used as the indices, by which the descriptor is inserted into the corresponding inverted list. For very similar retrieval complexity and pre-processing time, the inverted multi-index achieves a much denser subdivision of the search space compared to the inverted indexing structure in Jegou *et al.* (2011), while retaining memory efficiency.

An inverted list is equivalent to a cluster, and the centroid can be thought of as the cluster center. It takes more time to search in a larger cluster. So, the balance of inverted lists is an important factor which will impact the query response time.

Tavenard *et al.* (2011) proposed a balanced cluster scheme to produce clusters of much more even size. The key idea of this approach is to artificially enlarge the distances from the descriptor to the centroids of the heavily filled clusters so as to shrink and slightly drain the loaded cluster. This is realized by designing a penalization term, where the distance between the descriptor and a centroid is the sum of the Euclidian distance and the penalization term. The more heavily is the cluster filled, the larger the penalization term.

5 Conclusions and discussions

High-dimensional indexing structure is the core of retrieving similar images from large scale image collections accurately and quickly. In this paper, the past and current technical achievements in tree-based index, hashing-based index, and visual words based inverted index are reviewed.

Tree-based indexing structure is a conventional indexing technique, which successively partitions the

data space from coarse to fine and forms a tree of hierarchical structure. However, when the dimension of the descriptor exceeds 20, these methods usually suffer from the 'curse of dimensionality'. Thus, the high-dimensional descriptors are usually preprocessed by dimensionality reduction before being indexed by the tree-based indexing structure, or they are divided into several sub-vectors of low dimensionality that is suitable to the tree-based indexing structure.

Hashing-based index can be classified into two categories: E2LSH-related methods on behalf of data-independent hashing schemes and hashing coding methods on behalf of data-dependent hashing schemes. Compared with E2LSH-related methods, the methods of hashing coding have been more popular in recent years. Both E2LSH-ralated methods and hashing coding methods are inefficient on sparse descriptors, so the sparse descriptors should be preprocessed to be dense before being indexed by the hashing-based index.

Introduced from the textual retrieval system, visual words based inverted index has been widely used for large scale content-based image retrieval in recent years. With compressing descriptors and encoding descriptors, a huge number of descriptors can be stored directly in computer memory. However, the features of image are low-level, and features of small distance may not be similar in human perception, so the image descriptors, such as BOF, may not be discriminative enough to represent an image. Since the image descriptor is essential to image indexing structure and image retrieval, how to combine the low-level features with some semantic information to generate a more discriminative image descriptor, should be further researched.

Since a centroid is associated with an inverted list, inverted indexing methods need samples from the dataset to train the centroids. These centroids can reflect only the distribution in the dataset. When a batch of new data outside the dataset needs to be inserted into the inverted indexing structure or a batch of data is deleted from the dataset, the distribution of the dataset is changed and the pre-learned centroids become inaccurate. Thus, the retrieval accuracy would be influenced. In view of this scenario, current inverted indexing methods usually reconstruct the indexing structure. It would take much time for a huge dataset. So, the inverted indexing methods, which are robust to the operations of dynamic insertion or deletion, should be further researched. This is also one of our future research directions, with regard to a large scale content-based soccer video search engine.

Despite various schemes, such as dimensionality reduction and descriptor decomposition, adopted by nearly all the indexing methods, the 'curse of dimensionality' remains overwhelming. Therefore, an efficient indexing structure robust to high dimensionality needs to be further researched and it is also our current research focus.

Thanks to the development of mobile Internet and smart phone, people have been increasingly relying on smart phones. Moreover, with the increasingly powerful CPU (there are already four cores now) and large memory in smart phones, a wealth of pictures would be stored in personal phones. Thus, the demands for getting interested pictures quickly and accurately would inspire a new and popular research on efficient content-based indexing structure and retrieval towards smart phones.

References

Ai, L., Yu, J., Guan, T., 2012. Soft Assignment: Improving Image Representation in Content-Based Image Retrieval. 13th Pacific-Rim Conf. on Multimedia, p.801-810. [doi:10.1007/978-3-642-34778-8_75]

Anan, C.S., Hartley, R., 2008. Optimised KD-Trees for Fast Image Descriptor Matching. IEEE Computer Society Conf. on Computer Vision and Pattern Recognition, p.1-8. [doi:10.1109/CVPR.2008.4587638]

Andoni, A., Indyk, P., 2006. Near-Optimal Hashing Algorithms for Approximate Nearest Neighbor in High Dimensions. IEEE Symp. on Foundations of Computer Science, p.459-468. [doi:10.1109/FOCS.2006.49]

Andoni, A., Indyk, P., 2008. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. *Commun. ACM*, **51**(1):117-122. [doi:10.1145/1327452.1327494]

Arya, S., Mount, D.M., 1993. Algorithm for Fast Vector Quantization. Data Compression Conf., p.381-390. [doi:10.1109/DCC.1993.253111]

Avrithis, Y., Kalantidis, Y., 2012. Approximate Gaussian Mixtures for Large Scale Vocabularies. European Conf. on Computer Vision, p.15-28. [doi:10.1007/978-3-642-33712-3_2]

Babenko, A., Lempitsky, V., 2012. The Inverted Multi-index. IEEE Conf. on Computer Vision and Pattern Recognition, p.3069-3076. [doi:10.1109/CVPR.2012.6248038]

Beckman, N., Kriegel, H.P., Schneider, R., Seeger, B., 1990. The R*-Tree: an Efficient and Robust Access Method for Points and Rectangles. ACM Int. Conf. on Management of Data, p.322-331. [doi:10.1145/93597.98741]

Beis, J.S., Lowe, D.G., 1997. Shape Indexing Using Approximate Nearest-Neighbour Search in High-Dimensional Spaces. IEEE Computer Society Conf. on Computer Vision and Pattern Recognition, p.1000-1006. [doi:10.1109/CVPR.1997.609451]

Belkin, M., Niyogi, P., 2001. Laplacian Eigenmaps and Spectral Techniques for Embedding and Clustering. Advances in Neural Information Processing Systems (NIPS), p.585-591.

Bentley, J.L., 1975. Multidimensional binary search trees used for associative searching. *Commun. ACM*, **18**(9):509-517. [doi:10.1145/361002.361007]

Berchtold, S., Keim, D.A., Kriegel, H.P., 1996. The X-Tree: an Index Structure for High-Dimensional Data. Int. Conf. on Very Large Data Bases, p.28-39.

Biship, C.M., 2006. Pattern Recognition and Machine Learning. Springer, Heidelberg, p.430-455.

Blei, D.M., Ng, A.Y., Jordan, M.I., 2003. Latent Dirichlet allocation. *J. Mach. Learn. Res.*, **3**(3):993-1022.

Brandt, J., 2010. Transform Coding for Fast Approximate Nearest Neighbor Search in High Dimensions. IEEE Conf. on Computer Vision and Pattern Recognition, p.1815-1822. [doi:10.1109/CVPR.2010.5539852]

Bu, J., Tan, S., Chen, C., Wang, C., Wu, H., Zhang, L., He, X., 2010. Music Recommendation by Unified Hypergraph: Combining Social Media Information and Music Content. ACM Int. Conf. on Multimedia, p.391-400. [doi:10.1145/1873951.1874005]

Buhler, J., 2002. Provably Sensitive Indexing Strategies for Biosequence Similarity Search. Annual Int. Conf. on Computational Molecular Biology, p.90-99. [doi:10.1145/565196.565208]

Burkhard, W.A., Keller, R.M., 1973. Some approaches to best-match file searching. *Commun. ACM*, **16**(4):230-236. [doi:10.1145/362003.362025]

Caetano, T.J., Traina, A., Seeger, B., Faloutsos, C., 2000. Slim-Trees: High Performance Metric Trees Minimizing Overlap Between Nodes. Int. Conf. on Extending Database Technology, p.51-65. [doi:10.1007/3-540-46439-5_4]

Cai, Y., Tong, W., Yang, L., Hauptmann, A.G., 2012. Constrained Keypoint Quantization: Towards Better Bag-of-Words Model for Large-Scale Multimedia Retrieval. Int. Conf. on Multimedia Retrieval, p.1-8. [doi:10.1145/2324796.2324816]

Chemudugunta, C., Smyth, P., Steyvers, M., 2006. Modeling General and Specific Aspects of Documents with a Probabilistic Topic Mode. Advances in Neural Information Processing Systems (NIPS), p.1-8.

Chen, D., Tsai, S., Chandrasekhar, V., Takacs, G., Chen, H., Vedantham, R., Grzeszczuk, R., Girod, B., 2011. Residual Enhanced Visual Vectors for On-Device Image Matching. 45th Asilomar Conf. on Signals, Systems and Computers, p.850-854. [doi:10.1109/ACSSC.2011.6190128]

Chen, Y., Guan, T., Wang, C., 2010. Approximate nearest neighbor search by residual vector quantization. *Sensors*, **10**(12):11259-11273. [doi:10.3390/s101211259]

Ciaccia, P., Patella, M., Zezula, P., 1997. M-Tree: an Efficient Access Method for Similarity Search in Metric Spaces. Int. Conf. on Very Large Data Bases, p.426-435.

Conway, J.H., Sloane, N.J.A., 1982. Fast quantizing and decoding algorithm for lattice quantizers and codes. *IEEE Trans. Inf. Theory*, **28**(2):227-232. [doi:10.1109/TIT.1982.1056484]

Datar, M., Immorlica, N., Indyk, P., Mirrokni, V., 2004. Locality-Sensitive Hashing Scheme Based on *p*-Stable Distributions. 20th Annual Symp. on Computational Geometry, p.253-262. [doi:10.1145/997817.997857]

Dong, W., Charikar, M., Li, K., 2008. Asymmetric Distance Estimation with Sketches for Similarity Search in High-Dimensional Spaces. Proc. 31st Annual Int. ACM SIGIR Conf. on Research and Development in Information Retrieval, p.123-130. [doi:10.1145/1390334.1390358]

Douze, M., Ramisa, A., Schmid, C., 2011. Combining Attributes and Fisher Vectors for Efficient Image Retrieval. IEEE Conf. on Computer Vision and Pattern Recognition, p.745-752. [doi:10.1109/CVPR.2011.5995595]

Elsayed, E., Lin, J., Oard, D.W., 2008. Pairwise Document Similarity in Large Collections with MapReduce. 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies, p.265-268.

Fonseca, M.J., Jorge, J.A., 2003. Indexing High-Dimensional Data for Content-Based Retrieval in Large Databases. 8th Conf. on Database Systems for Advanced Applications, p.267-274. [doi:10.1109/DASFAA.2003.1192391]

Frey, B.J., Dueck, D., 2007. Clustering by passing messages between data points. *Science*, **315**(5814):972-976. [doi:10.1126/science.1136800]

Friedman, J.H., Bentley, J.L., Finkel, P.A., 1977. An algorithm for finding best matches in logarithmic expected time. *ACM Trans. Math. Softw.*, **3**(3):206-209. [doi:10.1145/355744.355745]

Georgescu, B., Shimshoni, I., Meer, P., 2003. Mean Shift Based Clustering in High Dimensions: a Texture Classification Example. Proc. 9th IEEE Int. Conf. on Computer Vision, p.456-463. [doi:10.1109/ICCV.2003.1238382]

Gionis, A., Indyk, P., Motwani, R., 1999. Similarity Search in High Dimensions via Hashing. Int. Conf. on Very Large Data Bases, p.518-529.

Gordo, A., Perronnin, F., 2011. Asymmetric Distances for Binary Embeddings. IEEE Conf. on Computer Vision and Pattern Recognition, p.729-736. [doi:10.1109/CVPR.2011.5995505]

Gray, R.M., Neuhoff, D.L., 1998. Quantization. *IEEE Trans. Inf. Theory*, **44**(6):2325-2383. [doi:10.1109/18.720541]

Guttman, A., 1984. R-trees: a dynamic index structure for spatial searching. *ACM SIGMOD Rec.*, **14**(2):47-57. [doi:10.1145/971697.602266]

He, J., Radhakrishnan, R., Chang, S.F., Bauer, C., 2011. Compact Hashing with Joint Optimization of Search Accuracy and Time. IEEE Conf. on Computer Vision and Pattern Recognition, p.753-760. [doi:10.1109/CVPR.2011.5995518]

Heo, J.P., Lee, Y., He, J., Chang, S.F., Yoon, S.E., 2012. Spherical Hashing. IEEE Conf. on Computer Vision and Pattern Recognition, p.2957-2964. [doi:10.1109/CVPR.2012.6248024]

Indyk, P., Motwani, R., 1998. Approximate Nearest Neighbor: Towards Removing the Curse of Dimensionality. 30th Annual ACM Symp. on Theory of Computing, p.604-613. [doi:10.1145/276698.276876]

Jagadish, H.V., Ool, B.C., Tan, K.L., Yu, C., Zhang, R., 2005. iDistance: an adaptive B^+-tree based indexing method for nearest neighbor search. *ACM Trans. Database Syst.*, **30**(2):364-397. [doi:10.1145/1071610.1071612]

Jain, M., Jegou, H., Gros, P., 2011. Asymmetric Hamming Embedding: Taking the Best of Our Bits for Large Scale Image Search. 19th ACM Int. Conf. on Multimedia, p.1441-1444. [doi:10.1145/2072298.2072035]

Jegou, H., Douze, M., Schmid, C., 2008a. Hamming Embedding and Weak Geometric Consistency for Large Scale Image Search. European Conf. on Computer Vision, p.304-317. [doi:10.1007/978-3-540-88682-2_24]

Jegou, H., Amsaleg, L., Schmid, C., Gros, P., 2008b. Query-Adaptative Locality Sensitive Hashing. IEEE Int. Conf. on Acoustics, Speech and Signal Processing, p.825-828. [doi:10.1109/ICASSP.2008.4517737]

Jegou, H., Douze, M., Schmid, C., 2009a. Packing Bag-of-Features. IEEE Int. Conf. on Computer Vision, p.2357-2364. [doi:10.1109/ICCV.2009.5459419]

Jegou, H., Douze, M., Schmid, C., 2009b. Recent advances in large scale image search. emerging trends in visual computing, *LNCS*, **5416**:305-326. [doi:10.1007/978-3-642-00826-9_14]

Jegou, H., Douze, M., Schmid, C., 2009c. Searching with Quantization: Approximate Nearest Neighbor Search Using Short Codes and Distance Estimators. Technical Report RR-7020, INRIA. Available from http://hal.inria.fr/docs/00/41/07/67/PDF/RR-7020.pdf

Jegou, H., Douze, M., Schmid, C., Perez, P., 2010a. Aggregating Local Descriptors into a Compact Image Representation. IEEE Conf. on Computer Vision and Pattern Recognition, p.3304-3311. [doi:10.1109/CVPR.2010.5540039]

Jegou, H., Douze, M., Schmid, C., 2010b. Improving bag-of-feature for large scale image search. *Int. J. Comput. Vis.*, **87**(3):316-336. [doi:10.1007/s11263-009-0285-2]

Jegou, H., Douze, M., Schmid, C., 2011. Product quantization for nearest neighbor search. *IEEE Trans. Pattern Anal. Mach. Intell.*, **33**(1):117-128. [doi:10.1109/TPAMI.2010.57]

Jegou, H., Perronnin, F., Douze, M., Sanchez, J., Perez, P., Schmid, C., 2012. Aggregating local image descriptors into compact codes. *IEEE Trans. Pattern Anal. Mach. Intell.*, **34**(9):1704-1716. [doi:10.1109/TPAMI.2011.235]

Joly, A., Buisson, O., 2011. Random Maximum Margin Hashing. IEEE Conf. on Computer Vision and Pattern Recognition, p.873-880. [doi:10.1109/CVPR.2011.5995709]

Katayama, N., Satoh, S., 1997. The SR-Tree: an Index Structure for High-Dimensional Nearest Neighbor Queries. Int. Conf. on Management of Data, p.369-380. [doi:10. 1145/253262.253347]

Ke, Y., Sukthankar, R., Huston, L., 2004. Efficient Near-Duplicate Detection and Sub-image Retrieval. ACM Int. Conf. on Multimedia, p.869-876. [doi:10.1145/1027527.1027729]

Kulis, B., Darrell, T., 2009. Learning to Hash with Binary Reconstructive Embeddings. Advances in Neural Information Processing Systems (NIPS), p.1-9.

Kulis, B., Grauman, K., 2009. Kernelized Locality-Sensitive Hashing for Scalable Image Search. IEEE 12th Int. Conf. on Computer Vision, p.2130-2137. [doi:10.1109/ICCV.2009.5459466]

Kuo, Y.H., Chen, K.T., Chiang, C.H., Hsu, W.H., 2009. Query Expansion for Hash-Based Image Object Retrieval. 17th Int. Conf. on Multimedia, p.65-74. [doi:10.1145/1631272.1631284]

Kuo, Y.H., Lin, H.T., Cheng, W.H., Yang, Y.H., Hsu, W.H., 2010. Unsupervised Auxiliary Visual Words Discovery for Large-Scale Image Object Retrieval. IEEE Conf. on Computer Vision and Pattern Recognition, p.905-912. [doi:10.1109/CVPR.2011.5995639]

Lepetit, V., Lagger, P., Fua, P., 2005. Randomized Trees for Real-Time Keypoint Recognition. IEEE Conf. on Computer Vision and Pattern Recognition, p.775-781. [doi:10.1109/CVPR.2005.288]

Li, D., Yang, L., Hua, X.S., Zhang, H.J., 2010. Large-Scale Robust Visual Codebook Construction. Int. Conf. on Multimedia, p.1183-1186. [doi:10.1145/1873951.1874182]

Lin, K.I., Jagadish, H.V., Faloutsos, C., 1994. The TV-tree: an index structure for high-dimensional data. *VLDB J.*, **3**(4):517-542. [doi:10.1007/BF01231606]

Liu, L., Wang, L., Liu, X., 2011. In Defense of Soft-Assignment Coding. IEEE Int. Conf. on Computer Vision, p.2486-2493. [doi:10.1109/ICCV.2011.6126534]

Liu, W., Wang, J., Ji, R., Jiang, Y.G., Chang, S.F., 2012. Supervised Hashing with Kernels. IEEE Conf. on Computer Vision and Pattern Recognition, p.2074-2081. [doi:10.1109/CVPR.2012.6247912]

Liu, X., Mu, Y., Lang, B., Chang, S.F., 2012. Compact Hashing for Mixed Image-Keyword Query over Multi-label Images. 2nd ACM Int. Conf. on Multimedia Retrieval, p.1-8. [doi:10.1145/2324796.2324819]

Lowe, D.G., 1999. Object Recognition from Local Scale-Invariant Features. IEEE Int. Conf. on Computer Vision, p.1150-1157. [doi:10.1109/ICCV.1999.790410]

Lowe, D.G., 2004. Distinctive image feature from scale-invariant keypoints. *Int. J. Comput. Vis.*, **60**(2):91-100. [doi:10.1023/B:VISI.0000029664.99615.94]

Lv, Q., Josephson, W., Wang, Z., Charikar, M., Li, K., 2007. Efficient Indexing for High-Dimensional Similarity Search. 33rd Int. Conf. on Very Large Data Bases, p.950-961.

Matei, B., Shan, Y., Sawhney, H.S., Tan, Y., Kumar, R., Huber, D., Hebert, M., 2006. Rapid object indexing using locality sensitive hashing and joint 3D-signature space estimation. *IEEE Trans. Pattern Anal. Mach. Intell.*, **28**(7):1111-1126. [doi:10.1109/TPAMI.2006.148]

Motwani, R., Naor, A., Panigraphy, R., 2008. Lower bounds on locality sensitive hashing. *SIAM J. Discr. Math.*, **21**(4):930-935. [doi:10.1137/050646858]

Mu, Y., Sun, J., Han, T.X., Cheong, L.F., Yan, S., 2010. Randomized Locality Sensitive Vocabularies for Bag-of-Features Model. European Conf. on Computer Vision, p.748-761. [doi:10.1007/978-3-642-15558-1_54]

Mu, Y., Chen, X., Chuan, T.S., Yan, S., 2011. Learning Reconfigurable Hashing for Diverse Semantics. Proc. 1st ACM Int. Conf. on Multimedia Retrieval, p.1-8. [doi:10.1145/1991996.1992003]

Nister, D., Stewenius, H., 2006. Scalable Recognition with a Vocabulary Tree. IEEE Computer Society Conf. on Computer Vision and Pattern Recognition, p.2161-2168. [doi:10.1109/CVPR.2006.264]

Norouzi, M., Fleet, D.J., 2011. Minimal Loss Hashing for Compact Binary Codes. 28th Int. Conf. on Machine Learning, p.353-360.

Norouzi, M., Punjani, A., Fleet, D.J., 2012. Fast Search in Hamming Space with Multi-index Hashing. IEEE Conf. on Computer Vision and Pattern Recognition, p.3108-3115. [doi:10.1109/CVPR.2012.6248043]

Panigraphy, R., 2006. Entropy Based Nearest Neighbor Search in High Dimensions. 17th Annual ACM-SIAM Symp. on Discrete Algorithm, p.1186-1195. [doi:10.1145/1109557.1109688]

Pauleve, L., Jegou, H., Amsaleg, L., 2010. Locality sensitive hashing: a comparison of hash function types and querying mechanisms. *Pattern Recogn. Lett.*, **31**(11):1348-1358. [doi:10.1016/j.patrec.2010.04.004]

Perronnin, F., Dance, C., 2007. Fisher Kernels on Visual Vocabularies for Image Categorization. IEEE Conf. on Computer Vision and Pattern Recognition, p.1-8. [doi:10.1109/CVPR.2007.383266]

Philbin, J., Chum, O., Isard, M., Sivic, J., Zisserman, A., 2007. Object Retrieval with Large Vocabularies and Fast Spatial Matching. IEEE Computer Society Conf. on Computer Vision and Pattern Recognition, p.1-8. [doi:10.1109/CVPR.2007.383172]

Philbin, J., Chum, O., Michael, I., Sivic, J., Zisserman, A., 2008. Lost in Quantization: Improving Particular Object Retrieval in Large Scale Image Databases. IEEE Conf. on Computer Vision and Pattern Recognition, p.1-8. [doi:10.1109/CVPR.2008.4587635]

Philbin, J., Isard, M., Sivic, J., Zisserman, A., 2010. Descriptor Learning for Efficient Retrieval. European Conf. on Computer Vision, p.677-691. [doi:10.1007/978-3-642-15558-1_49]

Robinson, J.T., 1981. The K-D-B-Tree: a Search Structure for Large Multidimensional Dynamic Indexes. ACM SIGMOD Int. Conf. on Management of Data, p.10-18. [doi: 10.1145/582318.582321]

Rui, Y., Huang, T.S., 1999. Image retrieval: current techniques, promising directions, and open issues. *J. Vis. Commun. Image Represent.*, **10**(1):39-62. [doi:10.1006/jvci.1999.0413]

Salakhutdinov, R., Hinton, G., 2007a. Semantic Hashing. SIGIR Workshop on Information Retrieval and Application of Graphical Models, p.1-8.

Salakhutdinov, R., Hinton, G., 2007b. Learning a Nonlinear Embedding by Preserving Class Neighborhood Structure. 11th Int. Conf. on Artificial Intelligence and Statistics, p.412-419.

Salakhutdinov, R., Hinton, G., 2009. Semantic hashing. *Int. J. Approx. Reason.*, **50**(7):969-978. [doi:10.1016/j.ijar.2008.11.006]

Sellis, T.K., Roussopoulos, N., Faloutsos, C., 1987. The R^+-Tree: a Dynamic Index of Multi-dimensional Objects. Int. Conf. on Very Large Data Bases, p.507-518.

Shao, J., Wu, F., Ouyang, C., Zhang, X., 2012. Sparse spectral hashing. *Pattern Recogn. Lett.*, **33**(3):271-277. [doi:10.1016/j.patrec.2011.10.018]

Sivic, J., Zisserman, A., 2003. Video Google: a Text Retrieval Approach to Object Matching in Video. Proc. 9th IEEE Int. Conf. on Computer Vision, p.1470-1477. [doi:10.1109/ICCV.2003.1238663]

Skopal, T., 2004. Pivoting M-Tree: a Metric Access Method for Efficient Similarity Search. Annual Int. Workshop on Databases, Texts, Specifications and Objects (Dateso), p.27-37.

Skopal, T., 2007. Unified framework for fast exact and approximate search in dissimilarity spaces. *ACM Trans. Database Syst.*, **32**(4), Article 29, p.1-47. [doi:10.1145/1292609.1292619]

Skopal, T., Hoksza, D., 2007. Improving the Performance of M-Tree Family by Nearest-Neighbor Graphs. 11th East European Conf. on Advances in Databases and Information Systems, p.172-188. [doi:10.1007/978-3-540-75185-4_14]

Skopal, T., Lokoc, J., 2008. NM-Tree: Flexible Approximate Similarity Search in Metric and Non-metric Spaces. 19th Int. Conf. on Database and Expert Systems Applications, p.312-325. [doi:10.1007/978-3-540-85654-2_30]

Skopal, T., Lokoc, J., 2009. New dynamic construction techniques for M-tree. *J. Discr. Algor.*, **7**(1):62-77. [doi:10.1016/j.jda.2008.09.013]

Strecha, C., Bronstein, A.M., Bronstein, M.M., Fua, P., 2012. LDAHash: improved matching with smaller descriptors. *IEEE Trans. Pattern Anal. Mach. Intell.*, **34**(1):66-78. [doi:10.1109/TPAMI.2011.103]

Tavenard, R., Jegou, H., Amsaleg, L., 2011. Balancing Clusters to Reduce Response Time Variability in Large Scale Image Search. 9th Int. Workshop on Content-Based Multimedia Indexing, p.19-24. [doi:10.1109/CBMI.2011.5972514]

Terasawa, K., Tanaka, Y., 2007. Spherical LSH for approximate nearest neighbor search on unit hypersphere. *LNCS*, **4619**:27-38. [doi:10.1007/978-3-540-73951-7_4]

Torralba, A., Murphy, K.P., Freeman, W.T., 2004. Sharing Features: Efficient Boosting Procedures for Multiclass Object Detection. IEEE Computer Society Conf. on Computer Vision and Pattern Recognition, p.762-769. [doi:10.1109/CVPR.2004.1315241]

Torralba, A., Fergus, R., Weiss, Y., 2008. Small Codes and Large Image Databases for Recognition. IEEE Computer Society Conf. on Computer Vision and Pattern Recognition, p.1-8. [doi:10.1109/CVPR.2008.4587633]

Torresani, L., Szummer, M., Fitzgibbon, A., 2010. Efficient Object Category Recognition Using Classemes. European Conf. on Computer Vision, p.776-789. [doi:10.1007/978-3-642-15549-9_56]

Tuytelaars, T., Schmid, C., 2007. Vector Quantizing Feature Space with a Regular Lattice. IEEE 11th Int. Conf. on Computer Vision, p.1-8. [doi:10.1109/ICCV.2007.4408924]

van Gemert, J.C., Geusebroek, J.M., Veenman, C.J., Smeulders, A.W.M., 2008. Kernel codebooks for scene categorization. *LNCS*, **5304**:696-709. [doi:10.1007/978-3-540-88690-7_52]

van Gemert, J.C., Veenman, C.J., Smeulders, A.W.M., Geusebroek, J.M., 2010. Visual word ambiguity. *IEEE Trans. Pattern Anal. Mach. Intell.*, **32**(7):1271-1283. [doi:10.1109/TPAMI.2009.132]

Wang, J., Kumar, S., Chang, S.F., 2010. Semi-supervised Hashing for Scalable Image Retrieval. IEEE Conf. on Computer Vision and Pattern Recognition, p.3424-3431. [doi:10.1109/CVPR.2010.5539994]

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. [doi:10.1109/TPAMI.2012.48]

Weiss, Y., Torralba, A., Fergus, R., 2009. Spectral Hashing. Advances in Neural Information Processing Systems (NIPS), **21**:1753-1760.

Wengert, C., Douze, M., Jegou, H., 2011. Bag-of-Colors for Improved Image Search. 19th Int. Conf. on Multimedia, p.1437-1440. [doi:10.1145/2072298.2072034]

Wu, L., Hoi, S.C.H., Yu, N., 2009. Semantics-Preserving Bag-of-Words Models for Efficient Image Annotation. 1st ACM Workshop on Large-Scale Multimedia Retrieval and Mining, p.19-26. [doi:10.1145/1631058.1631064]

Zhang, D., Wang, J., Cai, D., Lu, J., 2010. Self-Taught Hashing for Fast Similarity Search. Proc. 33rd Int. ACM SIGIR Conf. on Research and Development in Information Retrieval, p.18-25. [doi:10.1145/1835449.1835455]

Zhang, X., Li, Z., Zhang, L., Ma, W., Shum, H.Y., 2009. Efficient Indexing for Large Scale Visual Search. IEEE 12th Conf. on Computer Vision, p.1103-1110. [doi:10.1109/ICCV.2009.5459354]

Zhou, L., 2011. Research on Local Features Aggregating and Indexing Algorithm in Large-Scale Image Retrieval. Master Thesis, Huazhong University of Science and Technology, Wuhan, China, p.10-15 (in Chinese).

Zhuang, Y., Zhuang, Y., Li, Q., Chen, L., Yu, Y., 2008. Indexing High-Dimensional Data in Dual Distance Spaces: a Symmetrical Encoding Approach. Proc. 11th Int. Conf. on Extending Database Technology, p.241-251. [doi:10.1145/1353343.1353375]

Zhuang, Y., Liu, Y., Wu, F., Zhang, Y., Shao, J., 2011. Hypergraph Spectral Hashing for Similarity Search of Social Image. ACM Int. Conf. on Multimedia, p.1457-1460. [doi:10.1145/2072298.2072039]