

Efficient Indexing for Large Scale Visual Search

Xiao Zhang
Tsinghua University

xiao-zhang03@mails.tsinghua.edu.cn

Zhiwei Li, Lei Zhang, Weiying Ma, Heung-Yeung Shum
Microsoft

{zli, leizhang, wyma}@microsoft.com

Abstract

With the popularity of “bag of visual terms” representations of images, many text indexing techniques have been applied in large-scale image retrieval systems. However, due to a fundamental difference between an image query (e.g. 1500 visual terms) and a text query (e.g. 3-5 terms), the usages of some text indexing techniques, e.g. inverted list, are misleading. In this work, we develop a novel indexing technique for this problem. The basic idea is to decompose a document-like representation of an image into two components, one for dimension reduction and the other for residual information preservation. The computing of similarity of two images can be transferred to measuring similarities of their components. The decomposition has two major merits: 1) these components have good properties which enable them to be efficiently indexed and retrieved; 2) The decomposition has better generalization ability than other dimension reduction algorithms. The decomposition can be achieved by either a graphical model or a matrix factorization approach. Theoretic analysis and extensive experiments over a 2.3 million image database show that this framework is scalable to index large scale image database to support fast and accurate visual search.

1. Introduction

Retrieving similar images to a given image(s) from a huge image database is still an unsolved problem with many applications, e.g. image search, image annotation and object recognition [19]. Similar images are defined as images of the same object or scene viewed under different imaging conditions, e.g. lighting, viewpoint, etc. Due to good invariant properties to some imaging conditions, local features (e.g. SIFT[14]) are usually adopted to represent images [6]. However, by local features, an image is represented by a set of features, which is difficult to use. Thus, in state-of-the-art local feature-based image retrieval systems, “bag of visual terms” (BOV) model is widely adopted to convert local features of an image to a fixed-length histogram [18]. We start our discussion from a brief review of existing systems.

1.1. Overview of existing systems

A typical local feature-based large-scale image retrieval system could be divided into three parts: feature extraction and quantization, indexing and postprocessing.

Feature extraction and quantization is to convert an image into BOV representation. An image is usually represented by 500-5000 local descriptors. *k-means* clustering is the most popular approach to get BOV representations. [15] and [16] demonstrated the accuracy of retrieval increases with the increasing of the number of clusters (i.e. vocabulary size). However, the time to train a vocabulary with 1 million words by *k-means* is extremely long. [15] proposed to train the lexicon by a top-down hierarchical clustering manner. [16] proposed to use an approximate *k-means* to achieve this goal. Their following works used soft assignment[17] and hamming embedding[11] to alleviate information loss suffered by hard quantization.

Indexing: To speed up the process of finding relevant images for a query image, indexing techniques are necessarily adopted to organize images. In state-of-the-art systems, a popular indexing technique is the so called *inverted list* structure [4, 18, 16, 6], which has an entry for each visual term followed by a list of all the images in which the visual term occurs. Based on an index structure, we expect the relevance scores can be efficiently computed for images in the database and a small number of candidate images can be proposed. As in this step we need to find relevant images from a database with millions of images, the indexing design is very critical for a large-scale system.

However, a fundamental difference between an image query (e.g. 1500 visual terms) and a text query (e.g. 3 terms) is largely ignored in existing index design. This difference makes the inverted list inappropriate to index images. For example, given a query with three terms, all images containing them can be obtained by intersecting three rows in the inverted list. However, given a real image query which consists of 1500 visual terms, we have to intersect 1500 rows in the list. Thus, when inverted list is applied to index images, both its storage cost and time complexity to respond a query are unacceptably high. In this case, the performance of inverted list may even worse than linear

scan (say, without index). In a state-of-the-art image retrieval system which index about 1 million images [16, 6], the memory cost to host its index is 4.3GB which exceeds the maximum physical memory of a 32-bit machine. Thus, the average time to answer a query is 15-35 seconds. In contrast, it is reported that a commercial web search engine can index more than 5 million web pages by a 32-bit machine with the same data structure, and answer a query in a few milliseconds [2]. This creates a strong need to an efficient index structure specifically designed for visual search.

postprocessing: After an initial candidate set of images is obtained, which usually contains several hundred images, many existing computer vision techniques can be applied to rerank images in it. For examples, RANSAC-based fast image matching [16, 6], and query expansion-based high recall retrieval [6].

Despite of the heavy investigation of the first and third component, efficient indexing design has been given little attention since its first introduction in [18]. An accurate and fast image retrieval system is a precondition for many successive applications. Therefore, in this paper, we focus on the indexing design for large-scale visual search. Other approaches proposed for feature extraction and postprocessing can be easily combined with the proposed index solution.

1.2. High-dimensional indexing techniques

The BOV model represents each image with a fixed-length high-dimensional feature vector, which may contain more than 1 million components [16]. Although the dimensionality of the new image representation is very high, most of its components are zero. That is, the image feature is extremely high-dimensional and sparse. These characteristics bring difficulties to existing high-dimensional indexing techniques, e.g. Locality Sensitive Hashing (LSH) [1] and *KD*-tree[13]. LSH is known as an effective technique to index dense features, but not a good one to index sparse features because the L2 distance of sparse features is not as reliable as that of dense features [10]. *KD*-tree can only be applied when the dimensionality of the feature space is about a dozen [13], otherwise its performance may be even worse than linear scan.

It is worth noting that dimension reduction algorithms could be a solution to address this problem because they are capable of finding compact and dense representations of images in a low-dimensional space. Dimension reduction algorithms have two major merits. First, by capturing high level semantics, the projected low dimensional space could facilitate information retrieval. Second, high-dimensional indexing technique such as *KD*-tree or LSH performs better in the projected space, because image feature vectors in this space are compact and dense. However, retrieving images in a low dimensional space has a critical drawback because dimension reduction will inevitably result in the

lost of some important information. That is, some image specific information (say, residual) which cannot be effectively represented in the low-dimensional space. For example, when a user inputs a query image which contains a building facade, the projected low dimensional space may capture the general aspects of this image such as windows, pillars, roofs, etc. but may lose some specific information about this image like company logo or some special artifacts. However, from a user’s perspective, this kind of specific information is important. This problem will further affect the performance when the type of information contained in the query image is not contained in the training corpus used for dimension reduction algorithms. That is, the query image could not be well represented in the low dimensional space and the information lost is large.

1.3. Overview of our approach

The proposed solution consists of a feature decomposition model and a specifically designed indexing scheme.

1. We compute a compound representation for each image by a feature decomposition model. Both general and special aspects of an image can be extracted by the model. Beyond traditional dimension reduction algorithms, this model enables us to take the residual information into account. In the model, we can explicitly know which visual terms in an image cause most of the re-construction error. That is, we can identify and preserve the most special/important local features for an image. Thus, an image is represented by a low-dimensional vector and a few important key terms.

2. We design an appropriate indexing scheme by fully considering the physical meanings and storage characteristics of the two components of the compound representations. The low-dimensional representations of images can be efficiently indexed by LSH [1]. By indexing only those most important visual terms (say, 30 terms in our experiments), which cause the most of the reconstruction error, and taking them into account in the ranking algorithm, we achieve significant performance improvement.

We evaluate the proposed solution on some benchmark datasets as well as a dataset with 2.3 million images. With a postprocessing technique (RANSAC based image matching [16]), we achieve comparable performance with state-of-the-art approaches [16], but both the time to answer a query and memory cost to host index are significantly reduced.

2. Image Decomposition

2.1. Basic Idea

Our image decomposition idea is motivated from dimension reduction approaches because once we obtain low-dimensional and dense representations of images, many ex-

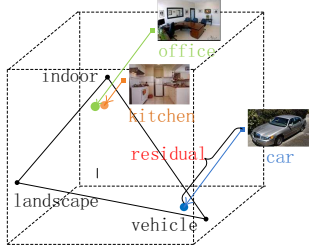


Figure 1. Illustration of the image decomposition idea. The cube denotes a vector space of images represented by bag of words, the triangle denotes a simplex spanned by three bases, a rectangle point denotes an image, a circle point denotes an embedding of an image in the subspace, and the distance of an image to the subspace is its residual

isting indexing techniques can be used to index images, e.g. LSH [1, 10]. A common objective function for dimension reduction algorithm is

$$\min \sum_{i=1}^N \|p_i - V \times h_i\|$$

where p_i is the high-dimensional histogram representation of an image [18, 16]¹, V is a matrix whose column vectors span a low-dimensional space, and h_i is the coefficients to reconstruct the image by column vectors of V . h_i is the compact representation of the original histogram feature p_i in the low-dimensional subspace V .

However, as we have discussed in Section 1.2, h_i does not contain complete semantics of an image. The lost semantics of an image in the dimension reduction process is likely to be very discriminative. Therefore, we propose a compound image representation which consists of a low-dimensional vector h_i and some important residual information lost in dimension reduction. Actually, an image p_i can be mathematically represented as follows

$$p_i = V \times h_i + \epsilon_i \quad (1)$$

where ϵ_i is residual of the reconstruction. ϵ_i has the same dimensionality as p_i . However, only components with the largest absolute values in ϵ_i , which are more important than other components in terms of reconstructing the image, need to be preserved. We will show in Section 4 that this compound representation has a very natural indexing scheme.

We further illustrate in Fig. 1 the key observation motivating the image decomposition idea. The two images about kitchen and office are projected to almost the same position(indoor) in the subspace, while the image about car is projected to a position(vehicle) far from them. Thus, it's easy for us to distinguish kitchen and car or office and car in the low dimensional subspace. However, only from their

¹We assume that the means of feature vectors are removed

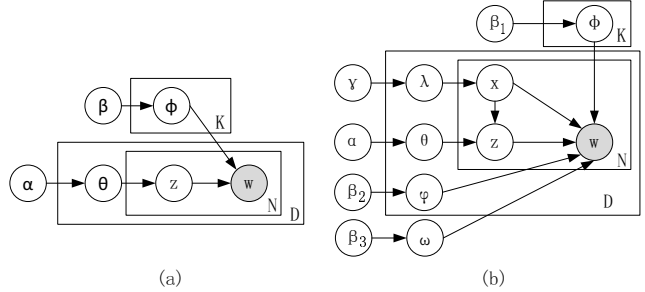


Figure 2. Probabilistic topic models. (a) latent Dirichlet allocation(LDA), (b) Image decomposition model (IDM)

coefficients in the subspace, we may not be able to distinguish the two indoor images(kitchen and office). Fortunately, the two images can be distinguished by their distances to the subspace, i.e. their residual (ϵ_i).

Two kinds of approaches could be considered for implementing the proposed idea. One is matrix factorization approaches, the other is probabilistic topic models. Most of matrix factorization approaches, e.g. PCA and non-negative matrix factorization (NMF) [12] could be adopted for this purpose. Although the computations of topic models are more complex than matrix factorization approaches, they offer better ways to understand problems and easy to be generalized to unseen data. Therefore, in this paper, we will mainly implement the image decomposition idea in a topic model way.

2.2. A Probabilistic Decomposition Model

Represented by a BOV model, an image can be deemed as a document. A way to implement the proposed decomposition idea is to use an Image Decomposition Model(IDM), which is an extension of the popular LDA model [3], which was first proposed to model documents, as shown in Figure 2(b). IDM was first proposed in [5] to detect important words of a document. To generate a visual word, a random variable x is first drawn from $Multinomial(\lambda)$, then

- if $x = 1$, draw $z \sim Multinomial(\theta)$, and then draw a visual word $w \sim Multinomial(\phi_z)$
- if $x = 2$, draw a visual word $w \sim Multinomial(\psi)$
- if $x = 3$, draw a visual word $w \sim Multinomial(\omega)$

The conditional probability of a visual word w given an image d in IDM could be written as

$$p(w|d) = \sum_z p(w|z)p(z|x=1, d)p(x=1|d) + p(w|x=2, d)p(x=2|d) + p(w|x=3, d)p(x=3|d)$$

The key idea of IDM is to use a switch variable x to control the generation of visual words. x can take value in 1,

2, 3, which controls that a visual word is generated from either a topic distribution (ϕ_z), an image-specific distribution (ψ_d), or a corpus background distribution (ω). It provides a natural way to partition an image into three types of visual words. Intuitively, a visual word which appears in almost all images is likely to be a background/stop word; a visual word which only appears in a few images but seldom appears in other images is likely to be an image-specific word; a visual word which widely appears in images with a common semantic but seldom appears in other images is likely to be a topic related word.

Mapping to the matrix factorization formulation, the work of IDM can be understood as simultaneously finding a group of basis vectors (ϕ_z , a set of distributions over vocabulary), coefficients (θ_d , topic mixture proportion), and residual (ψ_d , an image-specific distribution), which correspond to V , h_i and ϵ_i in Eq. 1, respectively. The visual terms generated from the background distribution correspond to the mean of image histograms. The compound representation of an image obtained by IDM is a pair $\langle \theta_d, \psi_{large} \rangle$, where ψ_{large} means part of ψ components which have the largest values.

2.2.1 Parameter Estimation

Because the maximum likelihood estimation of IDM is intractable [3, 5], we adopt the Monte Carlo EM algorithm to estimate parameters [7]. MCEM consists of two steps. In E-step, instead of computing the posterior of latent variables, we draw samples from it. Gibbs sampling is a popular method to sample complex posterior [8]. We adopt the collapsed Gibbs sampling algorithm proposed in [5], in which x and z are jointly sampled while other hidden variables are integrated out. The Gibbs sampling equations are listed as follows

$$\begin{aligned}
p(x_i = 1, z_i = k | x_{-i}, z_{-i}, w, \alpha, \gamma, \beta_1, \beta_2, \beta_3) \\
&= \frac{\gamma_1 + n_{d,-i}^1}{\sum_{j=1}^3 \gamma_j + n_{d,-i}} \times \frac{\alpha_1 + n_{d,-i}^{1,(.)}}{\sum_{j=1}^K \gamma_j + n_{d,-i}^{1,(.)}} \times \frac{\beta_1 + n_{d,-i}^{1,k,w_i}}{\sum_{j=1}^W \beta_j + n_{d,-i}^{1,k,w_j}} \\
p(x_i = 2 | x_{-i}, z_{-i}, w, \alpha, \gamma, \beta_1, \beta_2, \beta_3) \\
&= \frac{\gamma_2 + n_{d,-i}^2}{\sum_{j=1}^3 \gamma_j + n_{d,-i}} \times \frac{\beta_2 + n_{d,-i}^{2,w_i}}{\sum_{j=1}^W \beta_{2,j} + n_{d,-i}^{2,w_j}} \\
p(x_i = 3 | x_{-i}, z_{-i}, w, \alpha, \gamma, \beta_1, \beta_2, \beta_3) \\
&= \frac{\gamma_3 + n_{d,-i}^3}{\sum_{j=1}^3 \gamma_j + n_{d,-i}} \times \frac{\beta_3 + n_{d,-i}^{3,w_i}}{\sum_{j=1}^W \beta_{3,j} + n_{d,-i}^{3,w_j}}
\end{aligned}$$

where K is the number of topics, W is the size of visual vocabulary, $-i$ in subscript means whole variables excluding the i -th variable; $n_{d,-i}^1$, $n_{d,-i}^2$ and $n_{d,-i}^3$ mean the numbers of words generated from topics, image-specific distribution and background distribution in image d ; $n_{d,-i}^{1,k}$ means the number of words assigned to topic k in image d ; $n_{d,-i}^{1,k,w_i}$ means the number of times of word w_i assigned to topic k in the whole corpus; $n_{d,-i}^{2,w_i}$ means the number of times of

word w_i assigned to image-specific distribution in image d ; and $n_{d,-i}^{3,w_i}$ means the number of times of word w_i assigned to background distribution in the whole corpus.

In M-step, we maximize the expectation with respect to model parameters. The expectation can be approximated by

$$\arg \max_{\alpha, \gamma, \beta_{1,2,3}} \frac{1}{M} \sum_{j=1}^M \log(p(w, x_j, z_j | \alpha, \gamma, \beta_{1,2,3}))$$

where M is the number of samples drawn in E-step. Because all hidden variables are observed, the joint probability can be decomposed to a product of some simple probabilities, which can be estimated separately. We take the Newton-Raphson algorithm to estimate Dirichlet hyperparameters [3].

2.2.2 Inference of Unseen Images

After we estimate a model from training images, we need to infer latent variables for images to be indexed in the database as well as query images. For IDM, the inference algorithm is the same as the estimation. However, part of model parameters will be fixed to be those obtained in the estimation step. The two terms $\frac{\beta_1 + n_{d,-i}^{1,k,w_i}}{\sum_{j=1}^W \beta_j + n_{d,-i}^{1,k,w_j}}$ and

$\frac{\beta_3 + n_{d,-i}^{3,w_i}}{\sum_{j=1}^W \beta_{3,j} + n_{d,-i}^{3,w_j}}$ are replaced by ϕ_{k,w_i} and ω_{w_i} .

It is remarkable that the inference of different images are independent to each other. This fact can be easily verified by checking the Gibbs sampling equations for inference: the factors in these equations only depend on variables related to one image. That is, we can distribute images to different processors/machines and decompose them in a parallel way. Thus, the scalability of IDM algorithm is not a problem.

2.2.3 Similarity Measure

Given a query image, we use a heuristic method to compute its similarity to each image in the database. The similarity function is a linear combination of two evidences

$$sim(d, q) = \gamma_{d1} \gamma_{q1} sim(\theta_d, \theta_q) + \gamma_{d2} \gamma_{q2} sim(\psi_d, \psi_q) \quad (2)$$

where θ_d, θ_q stand for the topic mixture proportion of the two images, ψ_d, ψ_q stand for image-specific distributions ($p(w|x=2, d)$), and $\gamma_{.1}$ and $\gamma_{.2}$ stands for the word ratios of topic related terms and image-specific terms. These variables can be obtained by the inference algorithm. An advantage of this simple ranking function is that it does not introduce additional parameters. Actually, we can use any similarity function to compute $sim(\theta_d, \theta_q)$ and $sim(\psi_d, \psi_q)$. In our experiments, we adopted *cosine* similarity to compute them.

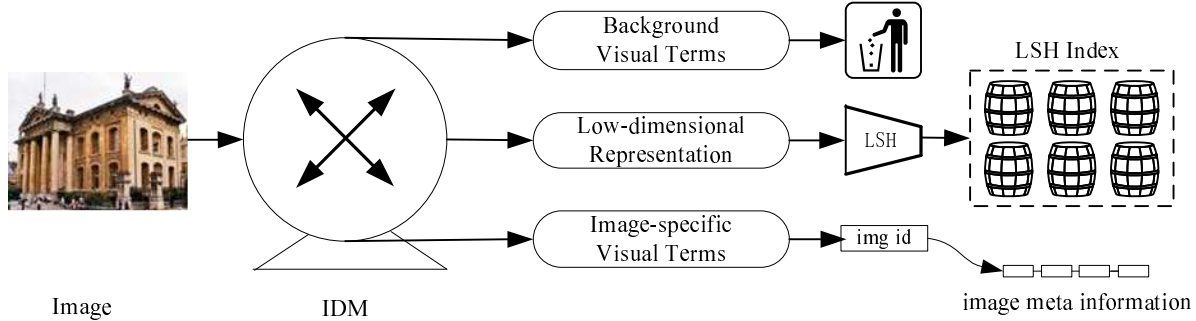


Figure 3. Work flow of our indexing framework. IDM is a centrifugal machine which decomposes an image to three components which are indexed in appropriate ways.

3. Indexing and Ranking

If we rank images according Eq. 2, we have to linearly scan all images in the database. In case the database is huge, the time complexity is unacceptable. To speed up the ranking process, we need a specifically designed indexing and fast ranking solution. The overall work flow of the proposed indexing solution is shown in Figure 3.

3.1. Indexing Scheme

As we have illustrated in Figure 1, different kinds of image representations have different functionalities and storage characteristics. Thus, we choose different approaches to index them. It is worth noting that background visual terms are common for all images in a database, that is, they cannot discriminate images with different semantics. Thus, they are discarded from index.

Indexing Low-dimensional Representations. We choose Locality Sensitive Hashing (LSH) [1] to index the low-dimensional representations of images (i.e. θ). Previous research [10] and our experiments demonstrated that the LSH is an effective approach to indexing compact and dense feature vectors. LSH will assign a group of hash values to an image. These hash values can be deemed as a new discrete representation of this image. Therefore, the memory requirement of LSH index is only $O(L * D)$, where L (e.g. 60 in our experiments) is the number of hash functions of LSH and D is the number of images.

Indexing Image-Specific Words. As we have mentioned, we can preserve only a small amount of visual words of an image as its image-specific words. These words can be selected according to the image-specific distribution ψ . In our experiments, we demonstrated the number of image-specific visual words could be very small. In our experiments, we obtained satisfied retrieval accuracies with only 30 image-specific words. We organize the image-specific words of an image as a fixed-length array, in which each component is a $\langle wordid, weight \rangle$ pair. This array is termed as image meta information. The most advantage of

the fixed-length array is that we can randomly access meta information of any image efficiently. The memory cost of this index in our implementation is only $O(30 * D)$.

3.2. Ranking Scheme

Considering the functionalities (in terms of representing semantics of an image) of the two components of the compound representation of an image, we develop an efficient ranking scheme. The similarity $sim(\theta_d, \theta_q)$ in Eq. 2 can distinguish images belonging to different visual categories, while $sim(\psi_d, \psi_q)$ is a complement to the first similarity and it can further distinguish images belonging to different sub visual categories. This observation motivates a two-step re-ranking approach for querying an image. A query image q is processed as follows

- Extract θ_q (topic mixture proportion) and ψ_q (an image-specific word distribution) from the query image.
- Generate an initial candidate set of relevant images (e.g. 200 images in our experiments) by approximately computing $cosine(\theta_d, \theta_q)$ using LSH.
- Assign final scores to images in the candidate set by using Eq. 2.
- Rank all candidate images according to the final scores and return the most relevant images to the user.

As shown above, in the first step, we use $sim(\theta_d, \theta_q)$ to select k images as a candidate set. In our implementation, θ_d is indexed by LSH. Thus, we can get an approximately accurate set efficiently. In the second step, we combine information from topic related words and image-specific words to re-rank images in this set. Our experimental results show that this approach is capable of obtaining almost the same results as the “accurate algorithm” (i.e. Eq. 2).

It is worth noting that, using the ranking algorithms based on the proposed index scheme, we cannot get the

same results as the accurate algorithm (Eq. 2) because: 1) LSH is an approximate nearest-neighbor algorithm; 2) only a few most salient image-specific visual terms are kept in index. Although this ranking scheme is only an approximation, its retrieval results are almost as good as the accurate algorithm in our experiments. The little sacrifice of accuracy significantly reduce the response time to queries and save memory cost. The two ranking approaches are termed as “*accurate*” and “*index*” in our experiments.

4. Experiments

In this section, we report the performance comparison of the proposed solution with several state-of-the-art approaches [17, 16, 6, 11] over a 2.3 million image database. To the best of our knowledge, this database is the largest one in this research direction.

4.1. Datasets

Four datasets (including two benchmark databases) were used in our experiments.

Oxford5K (5062 images, 18M descriptors, 55 queries) This dataset consists of 5062 images with extensively associated groundtruth for 55 queries: 5 queries for each of 11 Oxford landmarks [16]. This dataset served as the groundtruth in all experiments.

Paris (6300 images, 19M descriptors) This dataset consists of 6300 images obtained from Flickr [17]. Similar as the *Oxford5K* dataset, most of images in this dataset are buildings.

PhotoForum2M (2.3M images, 4165M descriptors). We collected these images from some professional photo forums, which cover various topics such as people, buildings and natural scenes. Some sample images of the two datasets are shown in Fig. 4. To test the impact of vocabularies constructed on different types of images, we construct a subset of *PhotoForum2M*, **PhotoForum7K** (7769 images, 22M descriptors).

4.2. Image Features

SIFT features were chosen as local features to represent images. We used the binary code shared by David Lowe to detect salient points and generate descriptors [14]. In feature extraction, we used default parameters of the tool. Visual vocabulary was generated by *k-means*, which was accelerated by the *KD-forest* algorithm proposed in [16]. In all probabilistic model-based approaches, we constructed vocabularies with 50k words, while in other approaches, we constructed vocabularies with 1 million words. We tested baseline approaches under their best settings [16].



Figure 4. Sample images from PhotoForum2M

4.3. Experiment Design

All experiments were carried out over a dataset constructed by mixing images from *Oxford5K* and *PhotoForum2M*. Images in *PhotoForum2M* served as distractors. The combined dataset contains 2,277,584 images. As we have summarized in Section 1.1, a typical image retrieval system consists of three major components: feature extraction and quantization, indexing, and postprocessing. To evaluate the performance of the proposed indexing solution, we varied settings of each component. These variations include:

- Construct visual vocabularies based on different types of images, i.e. *Oxford5K* and *PhotoForum7K*.
- Train IDM on both groundtruth dataset, i.e. *Oxford5K*, and a similar dataset, i.e. *Paris*.
- Change the number of image-specific words in index.
- Perform postprocessing or not.

We implemented two baseline approaches. The first one is the approach proposed in [16], in which images are represented by BOV models, weighted by TF-IDF scheme, and indexed by inverted list. It has the state-of-the-art performance of large-scale visual search. The second one is the approach proposed in [9], in which images are projected to a latent space obtained by training a latent Dirichlet allocation (LDA) model, and image similarities are measured in the latent space.

In retrieval applications, a popular and simple accuracy measure is the precision of top search results. In all experiments, we use the precision of top 10 search results ($p@10$), to measure the performances of all approaches.

4.4. Accuracy Comparison

Results of all approaches on two vocabularies are summarized in Table 1 and 2. Overall, the proposed approach

achieved even better accuracies than state-of-the-art approaches.

4.4.1 Impact of vocabulary and models

The two vocabularies were constructed over *PhotoForum7K* and *Oxford5K*. All topic model based approaches are trained with 250 latent topics. Only 30 image-specific words are preserved in index in our approaches. From these tables, we can see: 1) The proposed indexing solution performs better than state of the art approaches when vocabularies are trained either on *PhotoForum7K* or *Oxford5K*. 2) The model trained on *Oxford5K* achieved slightly better performance than the model trained on *Paris*, and the performances of LDA-based approaches dropped dramatically. When we decompose some kind of images, e.g. *Oxford5K*, using model trained on a different dataset, e.g. *PhotoForum7K/Paris*, more information will be preserved in the residual (image-specific words) instead of the low dimensional representation. This is the reason why the proposed index method, which combines information from both image-specific words and the low-dimensional representations obtained by dimension reduction, has better generalization ability. This is very important for real word applications.

To further verify this explanation, we reported the performance of a ranking approach which only depends on low-dimensional representations obtained by IDM, which is termed as “*Low-dim only*” in these tables. We found its performance is almost the same as the LDA based approach. However, once taking the image-specific words into account, its performance (say, the proposed indexing and ranking scheme) was significantly improved. Especially when the model or vocabulary cannot well explain images to be indexed, image-specific words must be included in index.

4.4.2 Improvement with postprocessing

We rerank a short list of images obtained by our indexing scheme based on the estimation of an affine transformation with an implementation of [16]. Table 1 and 2 also shows that the results obtained by reranking a short list of 100 images, which is termed as “index with reranking” in these tables. We observed further performance improvement. The performance gain of our approach is larger than the TF-IDF based approach. This results indicates that the proposed indexing and ranking scheme has better recall rate than the TF-IDF based approach.

4.4.3 The number of image-specific words

Table 3 shows the performances of the proposed approach under different numbers of image-specific words kept in

Approaches	Training Data for IDM	
	Oxford5K	Paris
Accurate	0.4227	0.3542
Low-dim Only	0.3632	0.2315
Index	0.3901	0.3387
Index With reranking	0.6304	0.5571
LDA	0.3927	0.2405
TFIDF	0.4116	
TFIDF with reranking	0.6047	

Table 1. Performance comparison for some variations of the proposed algorithm, LDA-based approach and TF-IDF approach. The vocabulary was constructed over *PhotoForum7k*.

Approaches	Training Data for IDM	
	Oxford5K	Paris
Accurate	0.6371	0.5774
Low-dim Only	0.5713	0.4105
Index	0.6207	0.5654
Index With reranking	0.8727	0.7803
LDA	0.6091	0.4674
TFIDF	0.6274	
TFIDF with reranking	0.8451	

Table 2. Performance comparison for some variations of the proposed algorithm, LDA-based approach and TF-IDF approach. The vocabulary was constructed over *Oxford5K*.

Approaches	The Number of Image Specific Words				
	30	60	90	120	150
Accurate	0.4227	0.4306	0.4369	0.4428	0.4475
Index	0.3901	0.3957	0.3996	0.4013	0.4072

Table 3. Performances of the proposed approaches under different numbers of image-specific words in index with visual vocabulary trained on *PhotoForum7K*.

the indexing structure, with vocabulary trained on *PhotoForum7K* and IDM trained on *Oxford5K*. No obvious improvement could be observed after the number of image-specific words kept in the indexing structure is more than 30 in the index scheme. This result demonstrates that a small number of image-specific words is enough for the proposed indexing scheme.

4.5. Cost Comparison

We compared the memory cost and time cost of the proposed approach with the inverted list based approach [16]. The results are summarized in Table 4.

Memory Cost: In a real system, index should be able to be loaded in memory to enable fast search. For the proposed index algorithm, the memory cost consists of two parts, memory cost of indexing low-dimensional representations and indexing image-specific words. The low-dimensional representations are indexed by LSH, whose size (#topics=250, LSH parameters L=60 and k=6 [16, 17]) is about 0.5869 GB for 2.3 million images. We use <word id, weight>pair to represent an image-specific word in image meta information. Assuming 30 visual words are se-

	Memory Cost(GB)	Time Cost(s)
Low-dim Representation	0.5869	
Image-specific Words	0.4688	0.873
Total	1.0557	
TFIDF	31.252	25

Table 4. memory and time cost for different methods on 2.3M image dataset

lected as image-specific words, the size of image meta information of 2.3 million images is about 0.4688 GB. Thus, all index data can be loaded in memory. In contrast, assuming each image contains 2000 visual words, then the index made by inverted list for 2.3 million images costs about 31.252 GB and could not be loaded into memory for a typical 32-bit workstation. In theory, a common back-end server with 8GB memory is capable of indexing 10 Million images using proposed indexing scheme.

Time Cost: The time cost of proposed indexing algorithm consists of two parts, time to decompose an query image to get its topic distribution (θ) and image-specific words, and time to query the database. As we use LSH to index the topic related word, querying image database is very fast. The average time to inferring an image is about 4.2 milliseconds. The average time for querying an image on the 2.3 Million database costs about 0.87 seconds (not including the RANSAC based spatial re-ranking step [16]).

5. Conclusion

In this paper, we have presented a novel indexing solution for local feature-based large-scale image retrieval systems. The solution consists of two key components: an image decomposition model, and an indexing and ranking scheme. The major contributions can be highlighted as follows. Based on the image decomposition model, the representation of an image can be decomposed to a compact vector as well as a few key visual terms. Retrieval accuracy based on the new representations is comparable to or even better than existing approaches in our experiments. Moreover, the proposed indexing and ranking scheme, which fully considers the characteristics of the new representations of images, is effective and efficient. With this scheme, both the computational and memory cost of online search service are significantly reduced, while the search results are almost as good as the accurate algorithm.

Both theoretic cost analysis and experiments over a 2.3 million image database demonstrate that this solution can be scaled up to index large-scale image databases for fast and accurate visual search.

References

[1] A. Andoni and P. Indyk. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. *Commun. ACM*, 2008. 2, 3, 5

[2] L. A. Barroso, J. Dean, and U. Hölzle. Web search for a planet: The google cluster architecture. *IEEE Micro*, 23(2):22–28, 2003. 2

[3] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 2003. 3, 4

[4] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. In *WWW*, 1998. 1

[5] C. Chemudugunta, P. Smyth, and M. Steyvers. Modeling general and specific aspects of documents with a probabilistic topic model. In *NIPS*, 2006. 3, 4

[6] O. Chum, J. Philbin, J. Sivic, M. Isard, and A. Zisserman. Total recall: Automatic query expansion with a generative feature model for object retrieval. In *Proceedings of the 11th International Conference on Computer Vision, Rio de Janeiro, Brazil*, 2007. 1, 2, 6

[7] W. Gang and M. Tanner. A monte carlo implementation of the em algorithm and the poor man’s data augmentation algorithms. *Journal of American Statistics Association*, 1990. 4

[8] T. L. Griffiths, M. Steyvers, D. M. Blei, and J. B. Tenenbaum. Integrating topics and syntax. In *NIPS*, 2004. 4

[9] E. Hörster, R. Lienhart, and M. Slaney. Image retrieval on large-scale image databases. In *CIVR*, pages 17–24, 2007. 7

[10] P. Indyk and N. Thaper. Fast image retrieval via embeddings. In *3rd International Workshop on Statistical and Computational Theories of Vision*, 2003. 2, 3, 5

[11] H. Jegou, M. Douze, and C. Schmid. Hamming embedding and weak geometric consistency for large scale image search. In *ECCV (1)*, pages 304–317, 2008. 1, 6

[12] D. D. Lee and H. S. Seung. Algorithms for non-negative matrix factorization. In *NIPS*, pages 556–562, 2000. 3

[13] T. Liu, A. W. Moore, A. Gray, and K. Yang. An investigation of practical approximate nearest neighbor algorithms. In *NIPS*, 2004. 2

[14] D. Lowe. Distinctive image features from scale-invariant keypoints. In *International Journal of Computer Vision*, 2004. 1, 6

[15] D. Nister and H. Stewenius. Scalable recognition with a vocabulary tree. In *CVPR*, 2006. 1

[16] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *CVPR*, 2007. 1, 2, 3, 6, 7, 8

[17] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Lost in quantization: Improving particular object retrieval in large scale image databases. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2008. 1, 6

[18] J. Sivic and A. Zisserman. Video google: A text retrieval approach to object matching in videos. In *ICCV*, pages 1470–1477, 2003. 1, 2, 3

[19] A. Torralba, R. Fergus, and W. T. Freeman. 80 million tiny images: a large dataset for non-parametric object and scene recognition. In *MIT-CSAIL-TR-2007-024*, 2007. 1