Learning to Distribute Vocabulary Indexing for Scalable Visual Search

Rongrong Ji, *Member, IEEE*, Ling-Yu Duan, *Member, IEEE*, Jie Chen, Lexing Xie, *Senior Member, IEEE*, Hongxun Yao, *Member, IEEE*, and Wen Gao, *Fellow, IEEE*

***Abstract*—In recent years, there is an ever-increasing research focus on Bag-of-Words based near duplicate visual search paradigm with inverted indexing. One fundamental yet unexploited challenge is how to maintain the large indexing structures within a single server subject to its memory constraint, which is extremely hard to scale up to millions or even billions of images. In this paper, we propose to parallelize the near duplicate visual search architecture to index millions of images over multiple servers, including the distribution of both visual vocabulary and the corresponding indexing structure. We optimize the distribution of vocabulary indexing from a machine learning perspective, which provides a "memory light" search paradigm that leverages the computational power across multiple servers to reduce the search latency. Especially, our solution addresses two essential issues: "What to distribute" and "How to distribute". "What to distribute" is addressed by a "lossy" vocabulary Boosting, which discards both frequent and indiscriminating words prior to distribution. "How to distribute" is addressed by learning an optimal distribution function, which maximizes the uniformity of assigning the words of a given query to multiple servers. We validate the distributed vocabulary indexing scheme in a real world location search system over 10 million landmark images. Comparing to the state-of-the-art alternatives of single-server search [5], [6], [16] and distributed search [23], our scheme has yielded a significant gain of about 200% speedup at comparable precision by distributing only 5% words. We also report excellent robustness even when partial servers crash.**

***Index Terms*—Distributed search, inverted indexing, parallel computing, visual search, visual vocabulary.**

I. Introduction

COMING with the popularity of local feature representations [1]–[3], recent years have witnessed an ever-increasing research focus on near duplicate visual search, with numerous applications in mobile location search, mobile product search, video copy detection and web image retrieval etc. In general, state-of-the-art visual search systems are built based upon a visual vocabulary model with an inverted indexing structure [4]–[7], which quantizes local features [1], [2] of query and reference images into visual words. Each reference image is then represented as a Bag-of-Words histogram and is invert indexed by quantized words of local features in the image. The Bag-of-Words representation provides good robustness against photographing variances in occlusion, viewpoint, illumination, scale and background. The problem of image search is then reformulated from a document retrieval perspective. Promising techniques include TF-IDF [8], pLSA [9] and LDA [10] have been reported. Approximate search techniques such as visual vocabulary and hashing have been well exploited in recent literatures to improve search efficiency in large image collections, e.g., Vocabulary Tree [5], Approximate K-means [6], Hamming Embedding [12], Locality Sensitive Hashing [11] and their variances [6], [13]–[15]. As a typical pipeline, visual vocabulary based search systems work in a client-server paradigm as follows: the client end (e.g., mobile devices or web browsers) sends a query image[1] to the server. At the server end, searching for similar reference images works in three phases: (1). Extracting local features from a query image (if compact visual descriptors are delivered rather than the query image, this step is skipped); (2). Quantizing local features into a Bag-of-Words histogram; (3). Ranking similar images in the inverted indexing of all non-empty words, where the linear scanning of all reference images can be avoided in similarity ranking.

To the best of our knowledge, existing visual search systems are often deployed over a single server. However, targeting scalable visual search applications, huge amounts of reference images (say millions or billions) need to be indexed and searched. To deal with massive data, it is almost impossible to maintain a scalable visual search architecture solely in one regular server. Taking our mobile location search system for instance, we need to search 10 million reference images online. Together with local features (for spatial verification and re-ranking, e.g., RANSAC), the visual vocabulary and the inverted indexing files, would easily cost up to 1 TB. Clearly, such a scale is beyond the storage capability of a single server, in terms of main memory or hard disks.

Nevertheless, substantially improving the storage capability of a server, e.g., by larger main memory and hard disks, is a straightforward alternative. Indeed, this is what big companies

Manuscript received August 20, 2011; revised January 03, 2012 and April 12, 2012; accepted May 01, 2012. Date of publication October 16, 2012; date of current version December 12, 2012. This work was supported by the National Science Foundation of China (60902057, 61271311), in part by the National Basic Research Program ("973") of China (2009CB320902). The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Shin'ichi Satoh.

R. Ji, L.-Y. Duan (corresponding author), J. Chen and W. Gao are with the Institute of Digital Media, Peking University, Beijing 100871, China (e-mail: lingyu@pku.edu.cn).

L. Xie is with the School of Computer Science, Australian National University, Canberra, ACT 0200, Australia.

H. Yao is with the Department of Computer Science, Harbin Institute of Technology, Harbin 150001, China (e-mail: yhx@vilab.hit.edu.cn). R. Ji and W. Gao are also affiliated with the Harbin Institute of Technology.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TMM.2012.2225035

[1]Alternatively, in recent mobile visual search systems [16], [17], sending compact descriptors of a query image may significantly reduce the latency of query delivery.

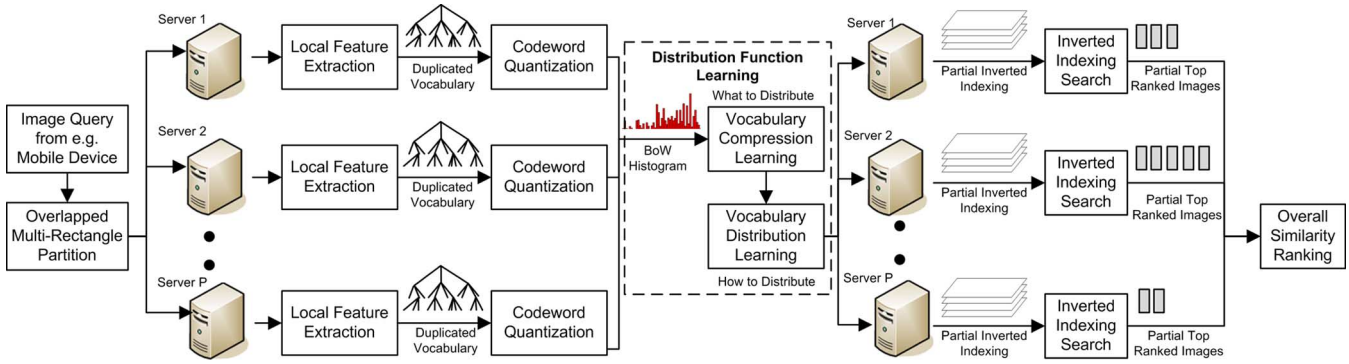


Fig. 1. The proposed learning based vocabulary indexing distribution scheme for scalable visual search on multiple servers.

TABLE I

TIME COST COMPARISON AT DIFFERENT PHASES OF THE DISTRIBUTED/NON DISTRIBUTED SEARCH SYSTEMS (MS). COMMUNICATION I: (TRANSFER WORDS TO DISTRIBUTED SERVERS); COMMUNICATION II: (MERGE THE MULTIPLE RANKING LISTS FROM DISTRIBUTED SERVERS). MAXIMUM SEARCH: MAXIMUM SEARCH COST (TIME) AMONG 20 DISTRIBUTED SERVERS. SIFT EXTRACTION IS PARALLELED BY DIVIDING THE QUERY IMAGE INTO P OVERLAPPING RECTANGLES, AS SHOWN IN FIG. 2. DUPLICATE DETECTED LOCAL FEATURE ARE FILTERED OUT BY COORDINATE DE-DUPLICATIONS, AS SHOWN IN FIG. 2. NOTE THAT THE COMPLEXITY OF SIFT EXTRACTION IS INDEPENDENT OF DATASET SCALE

	SIFT	Quantization	Communication I	Maximum Search	Communication II
Single Server (Oxford Buildings)	745.6 ms	387.2 ms	–	45 ms	–
20 Distribute Servers (Oxford Buildings)	37.28 ms	19.36 ms	≈ 0ms	3.32 ms	≈ 0ms
Single Server (10-Million Landmarks)	1036.85 ms	538.45 ms	–	–	–
20 Distribute Servers (10-Million Landmarks)	51.8425 ms	26.9255 ms	≈ 0ms	1171.88 ms	30 ms

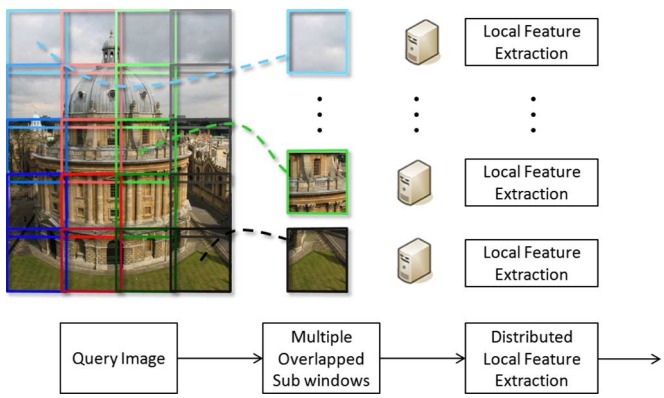


Fig. 2. Distribute the process of local feature extraction by partitioning a query image (left) into multiple overlapping rectangles followed by individual rectangles based feature extraction. Note that the slightly overlapping of nearby rectangles is to overcome missed detection.

like Google and Yahoo do relying on a super computer to deal with massive data in online search.[2] However, the explosive growth of web images renders the above solution less beneficial, especially for academia institutions as well as startup companies. For example, imagining when we scale up 10 million location image dataset by a factor of 1000, there would be at least 1 PB (1000 TB) data to be maintained in a single server, by assuming a linear increasing storage of features and indexing files versus the image volume. To the best of our knowledge, how to organize, index and search a huge image dataset in parallel is left unexploited, at least for both multimedia and computer vision communities.

From the perspective of distributed information retrieval, research in literatures can be basically categorized into either "local" or "global" distribution schemes, which are revisited latter in Section II. In this paper, we argue that for scalable visual search, directly applying previous distributed information retrieval techniques is unsuitable. This argument will be qualitatively explained in Section III and quantitatively validated in Section IV. Different from traditional distribution paradigms, we employ visual feature statistics, such as visual word concurrence and redundancy, for an optimal distribution strategy. Our goal is to significantly reduce the computational load via a distributed visual search architecture, so that massive reference images (say millions or billions) can be searched based on a group of laptops or regular servers, or on cloud computing resources.

To establish a visual search architecture over multiple servers, we will investigate the distribution of three phases of local feature extraction, quantization and inverted indexing, respectively:

First, we distribute the local feature extraction process by partitioning a query photo into multiple rectangles, which are subsequently executed in a parallel manner as shown in Fig. 2.[3]

Second, we perform local feature quantization in parallel by duplicating visual vocabularies in each server except inverted indexing, as storing a visual vocabulary is much more affordable than indexing files, as shown in Table II.

Third, we address the challenging problem of designing core indexing architecture, namely, how to distribute the inverted indexing structure. In our visual search scenario, an optimal dis-

[2]The "quasi-supercomputer" is Google's search engine system with an estimated processing power of 126 316 teraflops, as of April 2004. In June 2006 the New York Times estimated that the Googleplex and its server farms contain 450 000 servers. According to 2008 estimates, the processing power of Google's cluster might reach 20 100 petaflops. (http://en.wikipedia.org/wiki/Supercomputer).

[3]To avoid missed detection at the rectangle boundaries, the pairs of nearby rectangles are set slightly overlapped.

TABLE II
OVERALL STORAGE COST AND THE DATA VOLUME OF OUR EVALUATION VISUAL SEARCH SYSTEM

Database	Image Num.	Feature Num.	Feature Size	Index Size
UKBench	10K	4955K	2.36GB	0.037GB
Ox. Build.	5K	1994K	0.95GB	0.015GB
Landmarks	10M	4496M	2144GB	33.498GB

tribution scheme is expected to distribute the local descriptors extracted from a query image in a uniform manner as much as possible. So any server would not be assigned a large proportion of local features that could delay the process of entire search. Search latency can be largely reduced as multiple servers are fully involved to accomplish ranking. Meanwhile, the memory cost of inverted indexing in each server is affordable. Towards an optimal distribution of vocabulary indexing, there are two essential questions to be answered: "**What to distribute**" and "**How to distribute**", both are essential to parallel existing visual search schemes like [4]–[6], [16], [17] to satisfy large-scale real-world applications: "What to distribute" exploits the redundancy statistics of visual words, aiming to distribute partial and discriminative words from a given query into multiple servers. A selective distribution is to minimize the subsequent similarity ranking latency in inverted indexing, while maintaining comparable search precision.

"How to distribute" refers to carefully designing the distribution of visual words such that an online query can be processed uniformly over all machines, which alleviates the problem of one or several machines awaits the others to complete a query. This is achieved by exploiting the word concurrence statistics of reference images to predict a uniform word distribution of each online query. Ideally, each machine has an almost identical computational load; hence the query latency is minimized.

In this paper, we propose a learning based vocabulary distribution framework to address the above two issues, which enables very scalable search over millions of images using standard PCs or laptops. We brief our distributed search flowchart in Fig. 1 as follows: First, we partition a query image into multiple overlapped rectangles as shown in Fig. 2, so that local feature extraction is completed over multiple servers. Second, we quantize the extracted local features from a query image into visual words using the duplicate vocabulary in each server respectively (without inverted indexing). Third, we tackle the issues of both "What to distribute" and "How to distribute" from a machine learning perspective, minimizing the overall search latency as well as the computational cost in traversing the inverted indexing files in each server. Finally, the partial list of ranking images and the ranking scores from each individual server are combined to generate the final (or so-called global) image ranking list and scores.

To the best of our knowledge, this work serves as the first to distribute the visual vocabulary indexing for near duplicate visual search, especially from a machine learning perspective. The proposed distribution strategy may accommodate the existing visual vocabulary models such as [4]–[7], [15]. In summary, our contributions are three-fold:

- Towards "What to distribute", we propose a "lossy" vocabulary distribution scheme in Section III-B, which distributes a compact subset of discriminative visual words to accelerate online search while maintaining comparable search precision. We propose to learn the compact subset from a supervised dimension reduction perspective, which applies boosting to the original high dimensional vocabulary. The boosting criterion is to minimize the ranking distortion of "conjunctive" image queries (or so-called pseudo queries) when replacing the original high-dimensional Bag-of-Words histogram with its boosted sub-histogram.
- Towards "How to distribute", different from previous works in distributed information retrieval [19], [20], [22], [24], we propose to apply visual word concurrence to the distribution objective function (which partitions/distributes words into different servers) in Section III-C. The learning process is to maximize the possibility of uniformly partitioning local features from a given query into multiple servers. Consequently, each server processes a moderate number of words for similarity ranking based on partial inverted indexing files, which may reduce the latency of awaiting other servers.
- Finally, our learning based distribution can be further enhanced by frequent queries, such as user query logs from web image search engines, or user query images of popular landmarks in mobile location search systems, e.g., by emphasizing the uniform distribution of their visual words.

II. RELATED WORK

Visual Vocabulary Construction: Building visual vocabulary usually resorts to unsupervised vector quantization, which subdivides the local feature space into discrete regions each corresponds to a visual word. An image is represented as a Bag-of-Words (BoW) histogram, where each word bin counts how many local features of this image fall in the corresponding feature space partition of this word. To this end, many vector quantization schemes are proposed to build visual vocabulary, such as K-means [4], Hierarchical K-means (Vocabulary Tree) [5], Approximate K-means [6], and their variances [6], [13]–[15], [31], [32], [34]. Meanwhile, hashing local features into a discrete set of bins and indexed subsequently is an alternative choice, for which methods like Locality Sensitive Hashing (LSH) [25], Kernalized LSH [11], and Spectral Hashing [18] are also exploited in the literature. The visual word uncertainty and ambiguity are also investigated in [6], [12], [26], [27], using methods such as Hamming Embedding [12], Soft Assignments [6] and Kernelized Codebook [27].

Stepping forward from unsupervised vector quantization, semantic or category labels are also exploited [28]–[30] to supervise the vocabulary construction, which learns the vocabulary to be more suitable for the subsequent classifier training, e.g., the images in the same category are more likely to produce similar BoW histograms and vice versa. However, few work attempts to handle the storage complexity to maintain the vocabulary and its inverted files accordingly, which however is the key challenge to deal with the ever growing data corpus.[4] In particular,

[4]For instance, given a 100 million image collection, a single server with 10-20 GB memory and 1T-2T hard disk cannot maintain the vocabulary and inverted files, neither in its memory nor in its hard disks.

to the best of our knowledge, the parallel visual search strategy is rarely investigated in the literature.

Indexing File Pruning: Blanco *et al.* [37] proposed a method to prune the inverted files based on the Probability Ranking Principle as a decision criterion for posting list entries' pruning. Rather than pruning in a single machine, in our work we care more about which subset of terms (words) to be distributed into multiple servers, based on their visual word concurrence, different from using the probability ranking principle as in Blanco *et al.* [37]. Moreover, rather than considering every term in the lexicon as a single-term query to guide their Bayesian decision based term pruning, our Boosting based pruning is a combinational query formulation. In this sense, the next boosted word depends on the set of the previously boosted words, as detailed in Section III-B. Buttcher and Clarke [38] presented an alternative solution to prune document indices from the inverted indexing structures to achieve efficient retrieval on a single machine. In addition, Ntoulas and Cho [39] gave the pruning policies for two-tiered inverted index with a theoretical guarantee about their correctness.

Distributed Indexing for Scalable Information Retrieval: Undoubtedly, many scalable information retrieval systems heavily involve distributing the search and indexing architectures across multiple machines, due to processing the massive data is far beyond the computational limit of a single machine. For instance, Couvreur *et al.* proposed to distribute the vocabulary indexing structure into multiprocessor machines [19] with distinct hardware architectures, e.g., mainframes and RISC processors with LAN network connection. Stanfill *et al.* [21] proposed a distributed retrieval system with a fine-grained, massively parallel and memory-distributed SIMD architecture to achieve efficient retrieval. In [24], Barroso *et al.* further reported a distributed web search architecture developed in Google, which maintains a cluster of machines with duplicate vocabularies. Each machine serves partial documents or web pages. The cluster replication mechanism in [24] results in a very short online query response, meanwhile ensures a high network throughput.

In terms of functionality, the existing distributed indexing architectures can be categorized into either "local indexing" or "global indexing" alternatives. Both are well exploited in the literature [20], [22], [23].

- "Local indexing" refers to maintaining a duplicate vocabulary in each individual machine, where only partial documents are indexed. Hence, each document is indexed only in one of the machines.
- "Global indexing" refers to indexing all documents through different factions of a vocabulary, partitioning a vocabulary into multiple distinct parts, assigning inverted indexing accordingly, and finally distributing the fractional vocabularies and their corresponding indexing cross machines. Hence, each document could be indexed in multiple machines. As reported in [23], the "global indexing" schemes generally outperform the "local indexing" schemes in the TREC evaluations.

Moffat07 *et al.* [40] introduced two alternatives named "document partition" and "term partition" to design the distributed information retrieval system, which correspond to both "global indexing" and "local indexing" as illustrated in this paper. Martin and Gil-Costar [41] proposed a high-performance distributed inverted file system based on a round-robin query processing model. Again, both works [40], [41] tackled the distribution design still based upon heuristic criteria, without regard to the predicted distribution of word partitions from a machine learning perspective. In terms of query log learning, there are also previous works in information retrieval, which leverages the query logs to optimize the indexing partition for parallel web search systems, such as refining the objective function of term partition, based on the frequent patterns extracted from the past query logs [42].

We summarize the key difference between the previous works in inverted indexing file pruning and distribution retrieval in the traditional information retrieval research:

(1) Neither "global indexing" nor "local indexing" schemes tackles the issue of "What to distribute" to reduce the overall computational load. In other words, the distribution of limited but discriminative subset of words retains as our unique contribution after carefully comparing to the existing distributed IR research.
(2) "How to distribute" the vocabulary indexing from a machine learning perspective is left unexploited. Note that while there are several previous works on distributed IR for either "global indexing" (term partition) or "local indexing" (document partition), the existing works [38]–[41] are yet still based on heuristic criteria.
(3) In the visual search scenario, previous distributed IR works in both indexing file pruning and distributed information retrieval cannot be directly deployed, due to the specific requirement and unique characteristics of the visual word redundancy and concurrence. Correspondingly, in this paper we conduct the first investigation about how to prune the inverted file in visual search systems as well as how to distribute visual words distribution from a machine learning perspective. This statement is qualitatively explained in Section III and quantitatively validated in Section IV: In Section III, we show that one of our key differences is that we do not need to distribute all words (refer to the investigation of "What to distribute"), and we can optimize the distribution design to be related to the visual word concurrence (refer to the investigation of "How to distribute"), rather than simply based on uniform distribution or frequency distribution of words, both of which are typical solutions in the distributed IR research.

Distributed Visual Search: Limited research is devoted to scalable visual search, especially to distribute the visual vocabulary indexing structures. To the best of our knowledge, Maree *et al.* [35] reported a pioneer work on distributed image search with incremental indexing of reference images, i.e., incrementally index additional reference images into the search system. It adopted a "global indexing" scheme to partition the reference image collection into multiple groups, each of which is fully indexed using a randomized vocabulary in each machine. Yan *et al.* [36] also proposed a distributed image search architecture [35] for camera sensor networks. However, the search scalability of both works [35], [36] is left unexploited in million

scale image collections. In addition, both works [35], [36] need to quantize all features in every server, and hence the quantization step cannot be paralleled. Furthermore, the inverted indexing search of [35], [36] is accomplished in a rough manner, where the local ranking is performed over the partial image collection in each server, rather than in the entire dataset. Moreover, the network traffic of [35], [36] is heavy, as a bunch of local features from different machines has to be transmitted to all servers for quantization and ranking.

In our consideration, the above issues arise from using the local indexing architecture in [35], [36]. By contrast, the "global indexing" is more efficient in communication as the quantized words are only transmitted to its corresponding machine(s). In this paper, we prefer the "global indexing" scheme to deploy our distributed visual search system. And as quantitatively shown in Section IV, we can achieve better performance over the "local indexing" schemes in [35], [36]. Yet in a more generic sense, the proposed solution to "What to distribute" can benefit both "global indexing" and "local indexing".[5] Based on our empirical study, we report that the "global indexing" outperforms the "local indexing" for the scalable search of over 10 million reference images.

Likewise, topic models like pLSA and LDA [9], [10] can also group visual words into discrete subsets, which can be considered as another sort of vocabulary distribution strategy. The main difference lies in that topic models [9], [10] work on the word similarity rather than the word concurrence, the later of which is our focus. Comparing to topic models in abstracting a compact topical level image representation, our aim is to partition concurrent words into multiple machines as uniform as possible, so as to minimize the parallel computing time and memory cost in each individual machine.

III. Learning to Distribute Vocabulary Indexing

A. Vocabulary Indexing Model

We adopt the Vocabulary Tree (VT) model [5], [7], [16], [33] to build our initial vocabulary, upon which we deploy the inverted indexing structure. It is an efficient vector quantization scheme that adopts hierarchical k-means to partition local features into visual words.[6] In general, an H-depth VT with B-branch produces $\mathrm{D} = \mathrm{B}^{\mathrm{H}}$ words. Dealing with millions of reference images, the previous work [5] typically sets $\mathrm{H} = 6$ and $\mathrm{B} = 10$. We further denote the vocabulary as $\mathbf{V} = [\mathrm{V}_1, \ldots, \mathrm{V}_\mathrm{D}]$, where V_i is the ith visual word.

Given a query image $\mathrm{I_q}$ with in total J local features $\mathbf{S}_\mathrm{q} = [\mathrm{S}_1^\mathrm{q}, \ldots, \mathrm{S}_\mathrm{J}^\mathrm{q}]$, the VT quantizes $\mathbf{S}_\mathrm{q}$ by traversing the vocabulary hierarchy to find the nearest word. This procedure converts $\mathbf{S}_\mathrm{q}$ into a BoW histogram $\mathbf{V}_\mathrm{q} = [\mathrm{V}_1^\mathrm{q}, \ldots, \mathrm{V}_\mathrm{D}^\mathrm{q}]$.

Suppose there are N reference images in total. The online search gives a ranking order $\mathrm{R(i)}$ to each reference image $\mathrm{I_i}$ $(\mathrm{i} \in [1, \mathrm{N}])$. An optimal ranking should minimize the following ranking loss with respect to their BoW similarities:

$$Loss_{Rank} = \sum_{i=1}^{N} Rank(i) \| \mathbf{W}_i^T \mathbf{V}_q, \mathbf{W}_i^T \mathbf{V}_i \|_{Cosine} \quad (1)$$

where $\mathrm{Rank(i)}$ is inverse to the ranking position of image $\mathrm{I_i}$, e.g., $\exp(-\mathrm{rankingposition(i)})$. $\mathbf{W_i}$ denotes the TF-IDF weighting calculated via:

$$\mathbf{W}_i = \left[\frac{n_1^i}{n^i} \times log\left(\frac{N}{N_{V_1}}\right), \ldots, \frac{n_D^i}{n^i} \times log\left(\frac{N}{N_{V_D}}\right) \right] \quad (2)$$

where $\mathrm{n^i}$ denotes the number of local descriptors in $\mathrm{I_i}$, $\mathrm{n^i_{V_j}}$ $(\mathrm{j} \in [1, \mathrm{D}])$ denotes the number of local descriptors in $\mathrm{I_i}$ that are quantized into $\mathrm{V_j}$, N is the total number of images in the database, $\mathrm{N_{V_j}}$ $(\mathrm{j} \in [1, \mathrm{D}])$ is the number of images containing $\mathrm{V_j}$, $\mathrm{n_j^i/n^i}$ is the Term Frequency (TF) [8] of $\mathrm{V_j}$ in $\mathrm{I_i}$, and $\mathrm{\log(N/N_{V_j})}$ is the Inverted Document Frequency (IDF) [8] of $\mathrm{V_j}$ in the entire image collection.

B. Learning a Lossy Vocabulary Distribution

To answer "What to distribute", we propose to distribute only partial (not all) visual words to P servers. In other words, the word distribution $\mathrm{D(V_1, \ldots, V_D)}$ is "lossy". To this end, we discard the not discriminative words and their inverted indices before distributing them to P servers.

More specifically, we aim to learn a dimension reduction matrix $\mathbf{M}_{\mathrm{D \times K}}$ to transform $\mathbf{V} \in \mathbb{R}_\mathrm{D}$ into $\mathbf{U} \in \mathbb{R}_\mathrm{K}$:

$$\mathbf{U} = \mathbf{M}^T \mathbf{V}. \quad (3)$$

We simplify the learning of $\mathbf{M}_\mathrm{D} \times \mathrm{K}$ as a feature (word) selection by Boosting. The weak learner is each single word, and the objective function is to minimize the ranking loss of transforming $\mathbf{V}$ to $\mathbf{U}$. To this end, we set $\mathbf{MM}^\mathrm{T}$ as a diagonal matrix, where each diagonal position is either 0 or 1 to denote a word selection or non-selection.

To quantize the "ranking loss" of transforming $\mathbf{V}$ to $\mathbf{U}$, we sample $\mathrm{n_{Sample}}$ queries $\{\mathrm{I'_i}\}_{\mathrm{i=1}}^{\mathrm{n_{Sample}}}$ from the database as conjunctive query. For each $\mathrm{I'_j}$, we search the top R ranked images using $\mathbf{V}$, which returns a list of ranked images $[\mathrm{T_1^j}, \mathrm{T_2^j}, \ldots, \mathrm{T_R^j}]$, where $\mathrm{T_i^j}$ is the ith ranked image:

$$\begin{aligned} Query(I'_1) &= [T_1^1, T_2^1, \ldots, T_R^1] \\ &\cdots = \ldots \\ Query(I'_{n_{sample}}) &= [T_1^{n_{sample}}, T_2^{n_{sample}}, \ldots, T_R^{n_{sample}}]. \end{aligned} \quad (4)$$

For these conjunctive queries, we aim to boost a word subset $\mathbf{U}$ from $\mathbf{V}$, by using which the new ranking list can approximate the original ranking list $[\mathrm{T_1^j}, \ldots, \mathrm{T_R^j}]$ of each jth conjunctive query. We treat (4) as the ground truth and define an error weighting vector $[\mathrm{w_1}, \ldots, \mathrm{w_{n_{sample}}}]$ to $\{\mathrm{I'_j}\}_{\mathrm{j=1}}^{\mathrm{n_{sample}}}$, subsequently we iteratively select the best word from $\mathbf{U}$. At the tth iteration, we have the current $(\mathrm{t-1})$ non-zero diagonal elements in $\mathbf{MM}^\mathrm{T}$. To select the next tth non-zero element (corresponds

[5]For comparison, we provide quantitative evaluation of our learning based distribution scheme for both "global indexing" and "local indexing" latter in Section IV.

[6]Here, we show the exemplar solution using the VT model. But our solution is general enough for other vocabulary models such as K-means [4], Hierarchical K-means (Vocabulary Tree) [5], Approximate K-Means [6], and their variances [6], [13], [14], [32].

to the tth boosted word), we estimate the ranking preservation of the current $\mathbf{MM}^{\mathrm{T}}$ as:

$$Loss(I'_i) = w_i^{t-1} \sum_{r=1}^{R} Rank(T_r^i) \times \|\mathbf{W}_{T_r^i}^T \mathbf{M}^{t-1} \mathbf{U}_{I'_i}, \mathbf{W}_{T_r^i}^T \mathbf{V}_{T_r^i}\|_{Cosine}. \quad (5)$$

The ranking loss of query I'_{i} relates to the originally top R ranked images (from (4)) and their current positions, by using $\mathbf{U}_{\mathrm{i}}$ to search the reference dataset instead of $\mathbf{V}_{\mathrm{i}}$ ($\mathrm{i} \in [1, \mathrm{n_{sample}}]$); here $\mathrm{Rank}(\mathrm{T}^{\mathrm{i}}_{\mathrm{r}})$ is inverse to the current ranking position of the originally ith ranked image of I'_{i}, similar to that in (1); $[\mathrm{w}_1^{\mathrm{t}-1}, \ldots, \mathrm{w}_{\mathrm{n_{sample}}}^{\mathrm{t}-1}]$ is the $(t-1)$th error weighting to measure the ranking loss of the $\mathrm{n_{sample}}$ queries, which applies similar updating rule as the sampling weighting in AdaBoost. Subsequently, the overall ranking loss is:

$$Loss_{Rank} = \sum_{i=1}^{n_{sample}} Loss(I'_i). \quad (6)$$

And the best new word $\mathrm{U_t}$ at the tth iteration is selected by minimizing an overall ranking distortion:

$$U_t = \arg\min_j \sum_{i=1}^{n_{sample}} w_i^{t-1} \sum_{r=1}^{R} Rank(T_r^i) \times \|\mathbf{W}_{T_r^i}^T \mathbf{V}_{T_r^i}, \mathbf{W}_{T_r^i}^T \left[\mathbf{M}^{t-1} + \mathbf{M}_{additive}(j)\right] \mathbf{U}_{I'_i}\|_{Cosine} \quad (7)$$

where $\mathbf{M}_{\mathrm{additive}}(\mathrm{j})$ is an additive selection matrix that selects one word from $\mathbf{U}$ to be added into the new vocabulary $\mathbf{U}$ as:

$$\mathbf{M}_{additive}(j) = [0, \ldots, pos(j), .., 0]_D [0, \ldots, pos(t), .., 0]_K^T \quad (8)$$

where $[0, \ldots, \mathrm{pos(j)}, .., 0]_{\mathrm{D}}$ is a $\mathrm{D} \times 1$ selection vector to select the jth word; $[0, \ldots, \mathrm{pos(t)}, .., 0]_{\mathrm{K}}$ is a $\mathrm{K} \times 1$ position vector to map $\mathrm{V_j}$ into the new word $\mathrm{U_t}$.

We then update $\mathbf{M}$ and the error weighting of each $\mathrm{w}_{\mathrm{i}}^{\mathrm{t}-1}$ respectively as:

$$w_i^t = \sum_{r=1}^{R} Rank(T_r^i) \|\mathbf{W}_{T_r^i}^T \mathbf{V}_{T_r^i}, \mathbf{W}_{T_r^i}^T \left[\mathbf{M}^{t-1} + \mathbf{M}_{additive}(j)\right] \mathbf{U}_{I'_i}\|_{Cosine} \quad (9)$$

$$\mathbf{M}^t = \mathbf{M}^{t-1} + \mathbf{M}_{additive}(j). \quad (10)$$

The entire process is stopped if and only if $\sum_{\mathrm{i}=1}^{\mathrm{n_{sample}}} \mathrm{w}_{\mathrm{i}}^{\mathrm{t}} \leq \tau$, which defines an adaptive length based on the ranking discriminability of the boosted visual word subset. We summarize our vocabulary Boosting procedure in Algorithm 1.

Algorithm 1 Building Lossy Vocabulary by Boosting

1 **Input**: Bag-of-Words signatures $\mathbf{V} = \{\mathbf{V}_{\mathrm{i}}\}_{\mathrm{i}=1}^{\mathrm{n}}$; the sampled images $\{\mathrm{I}'_{\mathrm{i}}\}_{\mathrm{i}=1}^{\mathrm{n_{sample}}}$ with their original ranking list $\{\mathrm{Query}(\mathrm{I}'_{\mathrm{r}})\}_{\mathrm{r}=1}^{\mathrm{R}}$; Boosting threshold τ; error weighting vector $[\mathrm{w}_1, \ldots, \mathrm{w}_{\mathrm{n_{sample}}}]$; and Boosting iteration $\mathrm{t} = 0$;

2 **Pre-Computing**: Calculate $\mathrm{Loss_{Rank}}$ by **(6)**, calculate $\sum_{\mathrm{i}=1}^{\mathrm{n_{sample}}} \mathrm{w}_{\mathrm{i}}^{\mathrm{t}}$;

3 **while** $\{\sum_{\mathrm{i}=1}^{\mathrm{n_{sample}}} \mathrm{w}_{\mathrm{i}}^{\mathrm{t}} \leq \tau\}$ **do**

4 **Loss Estimation**: Calculate $\mathrm{Loss_{Rank}}$ by **(6)**;

5 **Codeword Selection**: Select $\mathrm{U_t}$ by **(7)**;

6 **Error Weighting**: Update $[\mathrm{w}_1, \ldots, \mathrm{w}_{\mathrm{n_{sample}}}]$ by **(9)**;

7 **Transformation Update**: Update $\mathbf{M}^{\mathrm{t}-1}$ by **(10)**;

8 $\mathrm{t}++$;

9 **end**

10 **Output**: The boosted vocabulary $\mathbf{U} = \mathbf{M}^{\mathrm{T}}\mathbf{V}$.

C. Learning a Uniform Distribution

To answer "How to distribute", we propose to learn a distribution function to uniformly partition $\mathbf{V}$ into (in total) P servers:

$$D(V_1, \ldots V_D) = [L_1, \ldots, L_D] \quad (11)$$

where $\mathrm{L_i} = \mathrm{p}$ denotes the ith word is assigned to the pth server. Equivalently, we aim for a minimal-cost partition of a *D*-Node graph into P subgraphs, based upon the visual word concurrence statistics, in which "cost" measures how uniform $\mathrm{D}()$ can partition local features extracted from a given query into these P severs.

While the real-world query is hard to obtain (we will revisit the usage of user query logs latter in this section), we leverage (17) to learn "How to distribute" this "lossy" vocabulary, where the distribution of kth server at the tth iteration is minimized by:

$$L_k = \arg\min_p Cost(U_k, S_p) \quad (12)$$

where $\mathrm{U_k}$ is a renewed word in the new vocabulary, $\mathrm{S_p}$ denotes the pth distributed server, $\mathrm{Cost}(\mathrm{U_k}, \mathrm{S_p})$ is the cost to distribute each renewed word $\mathrm{U_k}$ to the pth server, which replaces $\mathrm{V_k}$ with $\mathrm{U_k}$ in (18). The concurrence of $\mathrm{U_k}$ and $\mathrm{U_t}$ in $\mathrm{S_p}$ is estimated by counting images that contain both $\mathrm{U_k}$ and $\mathrm{U_t}$. Similarly to Section III, we use EM to learn the new distribution $\mathrm{D}(\mathrm{U_1}, \ldots, \mathrm{U_D})$.

In particular, we treat the N images $\mathbf{I} = \{\mathrm{I_i}\}_{\mathrm{i}=1}^{\mathrm{N}}$ (or its sampled subset) as the training data to guide the above graph partition in (11). Accordingly, the distribution cost of (11) with respect to $\{\mathrm{I_i}\}_{\mathrm{i}=1}^{\mathrm{N}}$ is defined as:

$$Cost = \sum_{i=1}^{N} Div(I_i) \quad (13)$$

$$Div(I_i) = \sum_{p=1}^{P} \sum_{p' \neq p} |\log \frac{n_{p'}^i}{n_p^i}| \quad (14)$$

$$n_p^i = |V_d | V_d \in I_i,\ V_d \in S_p| \quad (15)$$

where $\mathrm{S_p}$ is the pth server; $\mathrm{n_p^i}$ is the number of local descriptors that come from $\mathrm{I_i}$ and are distributed into $\mathrm{S_p}$.

Subsequently, the cost of a given distribution $\mathrm{D(V_1, \ldots, V_D)}$ is the sum of the costs to distribute each $\mathrm{I_i}$ into its corresponding servers,[7] which is measured by their divergence $\mathrm{Div(I_i)}$ to reveal its summarized (overall) load imbalance. For each $\mathrm{I_i}$, we expect $\mathrm{Div(I_i)}$ as small as possible. Ideally, a uniform distribution has $\forall_{\mathrm{p'}} \mathrm{n_p^i} = \mathrm{n_{p'}^i}$, which results in $\mathrm{Div(I_i)} = 0$. Note that we use $\log$ operator to punish the distribution that leads to a high load imbalance, which hinders the efficient search based on inverted indexing files. When each image is distributed across all servers uniformly, (13) outputs zero, which indicates that such a distribution configuration is optimal.

Instead of summing up the costs of individual images as shown in (13), we may estimate the overall distribution cost from the perspective of servers as:

$$Cost = \sum_{p=1}^{P} \left\{ \sum_{p' \neq p} \sum_{i \in N} n_p \log \frac{n_p^i}{n_{p'}^i} \right\} \quad (16)$$

which accounts for the overall cost of in total P servers. Each is measured by the number of concurrent local descriptors from an identical image but being used to search over other servers. Ideally, each server is assigned by the same amount of words from the given query. So that (16) would yield zero output, which means all servers are well running in a distributed search process.

We apply an iterative process to learn the optimal $\mathrm{D(V_1, \ldots, V_D)}$. Firstly, words are assigned to the P servers to minimize the cost in (13). Secondly, the divergence of each $\mathrm{I_i}$ is updated to renew this cost. We iterate the above two steps in an Expectation Maximization manner as follows:

Expectation: Given the word assignment at the $(t-1)$th iteration $\mathrm{D_{t-1}(V_1, \ldots, V_D)} = [\mathrm{L_1^{t-1}, \ldots, L_D^{t-1}}]$, at the tth iteration we assign a given $\mathrm{V_d}$ to the pth server based on:

$$L_d^t = \arg\min_p Cost(V_d, S_p) \quad (17)$$

where $\mathrm{Cost(V_d, S_p)}$ defines the cost from assigning word $\mathrm{V_d}$ to server $\mathrm{S_p}$ based on the $(t-1)$th concurrence between word $\mathrm{V_d}$ and the existing words in $\mathrm{S_p}$:

$$\begin{aligned} Cost(V_d, S_p) &= \sum_{V_t \in S_p} Concur_{t-1}(V_d, V_t) \\ &= \sum_{V_t \in S_p} |I_i | i \in [1, N],\ V_d, V_t \in I_i| \end{aligned} \quad (18)$$

where $\mathrm{Concur_{t-1}(V_d, V_t)}$ is measured by going through each word $\mathrm{V_t}$ in the pth server, counting the number of images each contains both word $\mathrm{V_d}$ and word $\mathrm{V_t}$.

[7]It is possible to distribute each image $\mathrm{I_i}$ into multiple servers in the current "global indexing" setting, since we distribute the visual word subset instead of the image subset.

Maximization: How to assign words from $\mathrm{I_i}$ to its corresponding servers depends on the expectation in (17) at the $(t-1)$th iteration. As a result, we update the word concurrence in each server $\mathrm{S_p}$ at the tth iteration as:

$$Concur_t(V_d, V_t) = \sum_{V_t \in S_p} |I_i | i \in [1, N] \wedge V_d, V_t \in I_i| \quad (19)$$

where $\mathrm{Concur_t(V_d, V_t)}$ is calculated likewise as described in (18).

D. Learning With User Query Logs

In practice, we may easily access to the user query logs (i.e., the collection of query photos) or the more frequent (or most probable queries). For instance, many online web image search engines can provide frequent user query logs. And more concretely, in mobile landmark search applications evaluated in this paper, the frequent queries can be accessed by collecting and parsing the most representative landmark photos.

To investigate the most frequent queries, we perform empirical study by collecting a set of M user query photos $\mathbf{Q} = \{\mathrm{Q_i}\}_{\mathrm{i=1}}^{\mathrm{M}}$ (from our real world 10-million landmark search system as detailed in Section IV) to supervise the above distribution learning. We inject this frequent query set into the distribution function learning in two aspects:

First, we plug $\mathbf{Q}$ in the EM estimation, where the concurrence $\mathrm{Concur(V_i, V_j)}$ between $\mathrm{V_i}$ and $\mathrm{V_j}$ in both (18) and (19) are updated as:

$$Concur(V_i, V_j) = \sum_{n=1}^{N} w_n Concur(V_i, V_j, I_n) \quad (20)$$

where $\mathrm{Concur(V_i, V_j, I_n)} = 1$ when both $\mathrm{V_i}$ and $\mathrm{V_j}$ are $\in \mathrm{I_n}$ and 0 otherwise. The $\mathrm{w_n}$ denotes the weight assigned to $\mathrm{I_n}$:

$$w_n = \begin{cases} 1, & \mathrm{I_n} \in \mathbf{N} \wedge \mathrm{I_n} \notin \mathbf{Q} \\ \omega, & \mathrm{I_n} \in \mathbf{N} \wedge \mathrm{I_n} \in \mathbf{Q} \end{cases} \quad (21)$$

where ω is a pre-defined integer $(\omega > 1)$ to reward a given images falling into the query log $\mathbf{Q}$.

Second, to supervise the lossy distribution learning with user query logs, $\mathbf{Q}$ is included in the sampled queries $\{\mathrm{I_i}\}_{\mathrm{i=1}}^{\mathrm{n_{sample}}}$ in the Boosting of $\mathbf{U}$ in Section III-B. Hence, we can simply replace the $\mathrm{n_{sample}}$ conjunctive queries with the query collection $\mathbf{Q}$.

E. Final Result Scoring

For the pth server of the in total P servers, we denote its output ranking list as $[\mathrm{S_1^p, S_2^p, \ldots, S_R(p)^p}]$ where $\mathrm{R(p)}$ is the number of non-zero scored images selected based on the inverted indexing based voting, and $\mathrm{S_j^p}$ is the weighted score (weighted vote) of the jth ranked image in the pth server, where the weight is calculated from TF-IDF weighting [8]. We fuse the P ranking lists from P servers, based on which the score of each kth image is calculated as follows:

$$S_k = \sum_{p=1}^{P} \{S_i^p | I(I, p) = I(k)\} \quad (22)$$

where $\mathrm{I(I, p) = I(k)}$ indicates that the $k\mathrm{th}$ ranked image in the final list is the $i\mathrm{th}$ ranked image from the $p\mathrm{th}$ server.

IV. Quantitative Analysis

Databases: To quantize the advantages of our learning based distributed vocabulary indexing, we evaluate on both UKBench and Oxford Building benchmarks, as well as a 10-million landmark image database collected from the web. In particular, both UKBench and Oxford building provide quantitative evaluations over the state-of-the-art alternatives, while our 10-million landmark dataset provides a large-scale real-world evaluation. We detail the above datasets as follows:

UKBench [5] contains 10 200 images with 2550 objects, including CD-covers, flowers and indoor scenes, etc. There are 4 images per object involving sufficient variances in viewpoints, lightings, occlusions and affine transforms. And the objective of this benchmark is to recognize different views of a particular object.

Oxford Buildings [6] contains 5062 images collected from Flickr by searching for particular Oxford landmarks. This collection has been manually annotated to generate a comprehensive ground truth for 11 different landmarks, each of which includes 5 possible queries. In total 55 queries are used to evaluate the retrieval performance. For each image of a landmark, a label is given by: (1). Good: A nice, clear picture of the object/building. (2). OK: More than 25% of the object is clearly visible. (3). Bad: The object is not present. (4). Junk: Less than 25% of the object is visible, or there are very high levels of occlusion or distortion.

10-Million Landmarks Dataset: We have collected over 10 million geo-tagged photos from photo sharing websites of Flickr[8] and Panoramio.[9] The data covers typical areas including Beijing, New York City, Lhasa, Singapore and Florence.

- **Training Query Set**: We partition photos in each city into multiple geographical regions using their geographical tags by k-means. Then, our system randomly selects $\mathrm{n_{sample}}$ photos from every geographical region as the conjunctive queries to learn "What to distribute". Subsequently, the top R returning photos are collected for each query using the original BoW histogram. Finally, assembling all queries and their ranking list, we form a conjunctive query set for the subsequent learning.
- **Ground Truth Labeling for Evaluation**: We invite volunteers to label landmark queries and their correct matching: For each city, we select the top 30 densest regions as well as 30 random regions based on the geographical partition. Since labeling the complete list of reference images of each query is intensive, in practice we identify one or more dominant landmark views from each of these 60 regions.
  Then, the near duplicate photos of each query are manually labeled in the query's belonging and nearby regions. Finally, 300 queries as well as their ground truth ranking lists[10] are yielded for each city. In total, we have 300×5 queries for the overall evaluation over these five cities.

Evaluation Criteria: We evaluate the proposed scheme from four aspects:

(1) *Processing Time*, which measures how a distributed indexing scheme outperforms the traditional single server strategy as well as alternative distribution strategies in terms of the search latency in seconds.
(2) *Speedup*, which measures the gain of search acceleration independent of the server configuration and network connection settings.
(3) *Load imbalance*, which indicates how well each server is fully utilized in a more balanced way.
(4) *mean Average Precision (mAP)*, which is to show the search precision loss when using our proposed lossy distribution scheme.

Processing Time: We measure the overall time cost as the maximum cost of a single machine among P servers. Given the extracted local features from a query, each server $\mathrm{S_p}$ $(\mathrm{p} \in \mathrm{P})$ utilizes its partial vocabulary and the corresponding inverted indexing to rank all co-indexed images to the subset of local features, whose time is denoted as $\mathrm{T_{S_p}}$. The overall processing time is the maximal cost among all servers, denoted as:

$$T_{Overall} = \max_{S_p} T_{S_p}. \tag{23}$$

Speedup: To make our measurement more independent to the different computational capabilities of different machines, we also measure the speedup before and after distributed indexing based on a proportional criterion:

$$Speedup = \frac{T_{Single}}{T_{Distributed}} \tag{24}$$

where $\mathrm{T_{Distributed}}$ and $\mathrm{T_{Single}}$ are processing time in distributed servers and single server (without distribution) respectively.

Load Imbalance: This measure calculates the ratio between the maximum and the averaged processing time costs among all machines as follows:

$$Loadimbalance = \frac{\max_{S_p} T_{S_p}}{\frac{1}{P}\sum_{p} T_{S_p}}. \tag{25}$$

A similar measure is also employed to evaluate the distributed IR system performance [23], which reflects the balance between servers, e.g., whether all servers are sufficiently running or whether partial servers are overloaded.

mAP: We use mean Average Precision (mAP) to compute the position-sensitive ranking precision of a set of queries based on the returning list:

$$mAP@N = \frac{1}{N_q}\sum_{i=1}^{N_q}\left(\frac{\sum_{r=1}^{N} P(r)rel(r)}{\min(N, \#-relevant-images)}\right) \tag{26}$$

where $\mathrm{N_q}$ is the number of queries, r is the rank, N is the number of reference images for query i, $\mathrm{rel(r)}$ is a binary function on the relevance of r, and $\mathrm{P(r)}$ is precision at the cut-off rank of

[8]http://www.Flickr.com

[9]http://www.Panoramio.com

[10]Here "ground truth" means the ranked image lists obtained by using the Bag-of-Words histograms in search.

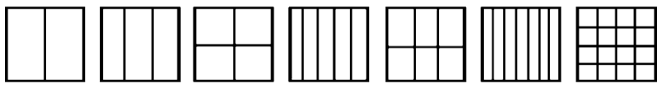


Fig. 3. Illustration of the spatial partition of P ($\mathrm{P} = 1$ to $\mathrm{P} = 20$). With the similar idea as shown in the above figure, we design the partition for the cases of 9–20 servers. In principle, when the image can be subdivided into $\mathrm{m} \times \mathrm{n}$, we will employ an $\mathrm{m} \times \mathrm{n}$ grids based partition; otherwise we will apply a uniform left-to-right partition.

r. Here, we have a min operation between the top N returning and #-relevant images. In a large scale search system, there are always over hundreds of ground truth relevant images to each query. Therefore, dividing by #-relevant-images would result in a very small mAP. A better alternative to be divided by the number of returned images. We use min(N, #-relevant-images) to calculate MAP@N. As N is at most 20 in our evaluation and always smaller than the number of labeled ground truth, we simply replace min(N, #-relevant-images) with N in subsequent calculation.[11]

Parameter Setting and Storage Cost: We extract SIFT features [1] for each image in each reference dataset. Based on the local features, we employ Vocabulary Tree model [5] to build the initial vocabulary $\mathbf{V}$ used before the lossy distribution. The quantization division stops once there are less than 500 SIFT points in a word node. We denote the hierarchical level as H, which decides the layers to run the hierarchical clustering. We denote the branching factor as B, which decides the number of clusters when a given parent cluster at the higher layer is allowed to be split into subsets. This setting produces $\mathrm{D} = \mathrm{B}^{\mathrm{H}}$ words in the finest level at most. In a typical setting, we have $\mathrm{H} = 5$ and $\mathrm{B} = 10$ for both UKBench and Oxford Buildings benchmarks, which produces approximately 10 000 words at the finest granularity as visual words. For our 10-million image database, we have $\mathrm{H} = 7$ to have approximate 1 million words. Table II estimates the storage cost in both memory and hard disk, that involves vocabulary, indexing and reference data set. The adaptive lossy learning is stopped once the ranking preserving of the boosted words satisfies $\sum_{\mathrm{i}=1}^{\mathrm{n}_{\mathrm{sample}}} \mathrm{w}_{\mathrm{i}}^{\mathrm{t}} \leq \tau$.

For the rectangle partition, note that we do not unify the aspect ratios among different input images in both offline training and online query. The height and width of each rectangle are settled adaptively based on the height and width of the input image, as well as the number of P (the number of servers). For instance, if $\mathrm{P} = 20$, we use the regular division as in Fig. 2. The overlapping ratio is set as 5% between each pair of rectangles.

For different numbers of P, we have different partition designs. For each case, we ensure that the partition is as unify as possible in terms of the spatial area of each rectangle. To facilitate our explanation, we draw the following visualization for partition 1–8 as in Fig. 3:

Distributed Servers and Concurrent Query Configuration: We have successfully deployed a real world distributed search system in LAN environment that consists of 20 personal computers (PCs). Each PC is with an $\mathrm{Intel\text{-}Xeon5620} \times 2$ with a 2.4 GHz processor, 16 GB memory, and 2TB hard disk. Each PC runs Linux kernel 2.6.35 and is connected to each other by a 10 M Ethernet connection via a 32 port switch. To simulate the real world search scenario where hundreds of web users might upload query image simultaneously, we perform quantitative evaluations by running 500 concurrent queries each time. For both UKBench and Oxford Building datasets, the concurrent queries are set identical as the original queries [5], [6]. And for our 10-Million Landmark dataset, the concurrent queries are selected as described in Section IV.

Baselines: We carry out five groups of baseline experiments including single-server indexing, "global" indexing and "local" indexing, detailed as follows:

"Local Indexing": A straightforward solution is to maintain the entire vocabulary in each server, keeping partial image collection indexed, e.g., the proportion of 1/P of the entire image collection. This is a sort of local indexing mode.

"Local Indexing with Boosting": Learning "What to distribute" is employed to perform the local indexing. As a result, the boosted words are maintained instead of the original vocabulary in each server. Likewise, only partial images of the entire reference data set are (locally) indexed on each server.

"Direct Distribute" and "Frequency Distribution": Another two straightforward distributed indexing schemes are to directly distribute words in equal partitions, or further based on the word frequency. For the latter, the most frequent words are iteratively distributed to each server, which ensures that most frequent words are distributed more or less uniformly across P servers.

"Lossless Distribution": Without applying vocabulary Boosting, learning $\mathrm{D}(\mathrm{V}_1, \ldots, \mathrm{V}_{\mathrm{D}})$ is lossless as none of words is discarded in the distribution. It is worth to note that each server indexes all images that have local descriptors falling into the word subset assigned to this server. It is a "global indexing" scheme.

"Lossy Distribution": This scheme combines the distribution learning of $\mathrm{D}(\mathrm{V}_1, \ldots, \mathrm{V}_{\mathrm{D}})$ and the Boosting based word selection. As a sort of "global indexing", it addresses both "What to distribute" (by vocabulary Boosting) and "How to distribute" (by learning distribution function).

"Lossy Distribution with User Query Log": We inject the user query logs to learn the distribution function $\mathrm{D}(\mathrm{V}_1, \ldots, \mathrm{V}_{\mathrm{D}})$ ("How to distribute") as well as the vocabulary Boosting ("What to distribute").

"Lossy Distribution with PCA": As an alternative to Boosting based lossy distribution, we also test the feasibility of other BoW feature space compression schemes, for which PCA is adopted as an alternative approach.

In particular, to evaluate the mAP maintaining capability of our lossy distribution (Method(5)), we perform comprehensive comparison with the state-of-the-art vocabulary indexing model [5], [6], [16] deployed in a single server, including Vocabulary Tree [5], Approximate K Means [6], and Tree Histogram Coding [16].

Visual Word Concurrence: We first validate the motivation of our concurrence based visual word distribution scheme. This fact demonstrates that the straightforward solution (e.g., Method (3)) cannot work well in our application scenario. Fig. 4 shows the visual word concurrence in UKBench. The left to right subfigures show the cases of different vocabulary sizes from 100

[11]The min(N, #-relevant-images) operation is a common evaluation in the TRECVID evaluation (http://trecvid.nist.gov/).

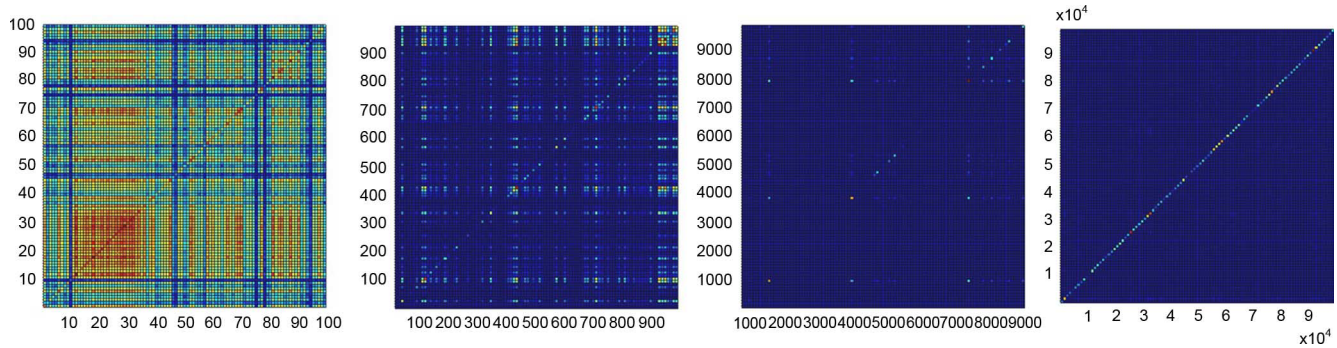


Fig. 4. The visual word concurrence matrix at different vocabulary sizes in the UKBench dataset. Each subfigure is a co-occurrence table, with hierarchical levels $\mathrm{H} = 2, 3, 4, 5$ to produce 100, 1,000, 10 000 and 100 000 words (red-blue: maximal-minimal concurrence, best view in color). It is obvious that visual words are highly concurrent and do not show a uniform distribution with the increasing number of words (The less heat colors are due to the effect of normalized distribution over more visual words.) Even with 100 000 visual words, the distribution map is not smooth. In the above subfigures, the concurrence is based on the entire dataset, rather than within a single image. For the image-level concurrence, the word distribution would be even highly concurrent.

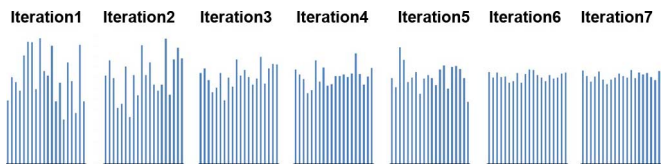


Fig. 5. The histogram based visualization of learning convergence process, which shows the convergence of learning the distribution of a test query image over 20 servers. The volume in each histogram bin indicates the number of local features falling to each server respectively. Histogram bins reflect the visual word distribution over 20 servers at different iterations.

to 100 000 respectively. As shown, these concurrence matrices are really sparse, where the minority of words are highly concurrent with each other, while the majority of words are rarely concurrent. Such concurrence observation indicates that words are fairly separable across groups by nature, which supports the improvement of speed and load balance in the scenario of distributed image search.

Visualized Learning Convergence: Fig. 5 visualizes the convergence procedure of our lossless distribution learning. In this example, each subfigure corresponds to a certain learning iteration, and the contours of histogram bins show how the local features from a given query image (over 300) are distributed across P servers $(\mathrm{P} = 20)$. Note that each bin counts the number of assigned local features in a given server, which serves as a sort of estimating the processing time unbalance between servers. Therefore, more uniform histogram distribution means that the corresponding search can be performed more efficiently, as all servers are fully exploited in this search task. As shown in Fig. 5, the uniform characteristics are incrementally improved from Iterations 1 to 7 by using our distribution function learning.

Practical Time Comparison: Table I compares the practical time costs before and after our lossless distribution learning. While our scheme requires two additional network communications (e.g., Communication I to transfer words into distributed servers and Communication II to merge multiple rankings from P servers), the overall time is largely reduced, since we greatly reduce the load of the most time-consuming parts, including both local feature extraction and inverted indexing based search. Table III shows the time cost improvements of our query log learning in 20 servers, with a large margin over learning schemes without query logs. In this evaluation, for both lossless and lossy distribution learning (4), the user queries are collected from the most frequent queries uploaded to our landmark search system to replace the original $\mathrm{n}'_{\mathrm{sample}}$ images that are randomly selected. Note that in Table III some scores for single server based implementation is left empty, as it is impossible to maintain a 10-million image indexing structure within a single server. Fig. 6 further visualizes their distribution of words for several exemplar query images.

Speedup with Respect to Server Number: We further evaluate the speedup of our vocabulary distribution learning by increasing the number of distributed servers. Fig. 7 shows that our speedup is linear to the server number, when the server number is less than 20. The possible degenerations mainly come from the costs of inter-server communications. As shown in the comparisons of Fig. 7, the proposed scheme largely outperforms the local indexing, direct distribution and frequency based distribution schemes (Methods (1)(3)) (Note that all learning schemes in this paper perform better than both direct distribution and frequency based distribution.) In particular, our lossy distribution learning has achieved the highest speedup, comparing to both lossless distribution. Finally, incorporating the query logs can further improve the speedup of the proposed distribution scheme.

Load Imbalance Improvement: Fig. 8 shows the load imbalance improvement of the proposed distribution learning scheme. Comparing to other alternatives, our scheme can more uniformly partition words based on their concurrence statistics to ensure an extremely low load imbalance. Given a query image, the extracted local features are less possible to be quantized and dispatched to an identical server. In such a case, the choice of either lossy or lossless distribution might not bias the performance of load imbalance. It is worth mentioning that, although learning based distribution is tackled in the paradigm of global indexing, it can be also extended into the paradigm of local indexing, e.g., distributing image subsets rather than word subsets into multiple servers with fully indexing vocabularies.

Fig. 6. Case study of log based learning. Each circle indicates the words' distribution of a given query in each server, where the length of each radius proportionally corresponds to the assigned words' volume on each server (red: the word distribution prior to lossless log learning as of Baseline (6); blue: the word distribution after lossless log learning as of Baseline (6)).

TABLE III
ONLINE SEARCH TIME COSTS (MS), I.E., THE SEARCH TIME OF 500 CONCURRENT QUERIES ON AVERAGE OVER A DISTRIBUTED INDEXING SYSTEM WITH 20 SEVERS

	Ox. Build.	10-Million Landmarks
Single Server	43.50 ms	N.A.
Lossless Distrib.	3.26 ms	1154.56 ms
Lossy Distrib.	1.95 ms	623.62 ms
Lossless Distrib. with Log	3.21 ms	1156.33 ms
Lossy Distrib. with Log	1.92 ms	608.08 ms
Local Indexing	3.35 ms	1409.12 ms

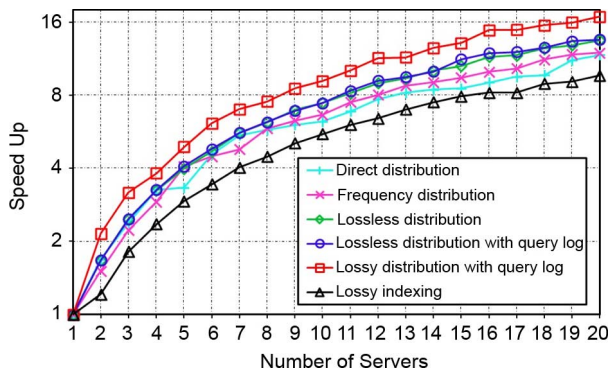


Fig. 7. Speedup with respect to the number of servers in 10-Million Landmarks. x-axis: the number of servers; y-axis: speedup. Lines: 1). Direct distribution (Method (3)); 2). Frequency distribution (Method (3)); 3). Lossless distribution (Method (4)); 4). Lossy distribution (Method (5)); 5). Lossy distribution with log (Method (6)); 6). Local indexing (Method (1)). The performance is based on 500 concurrent queries on average, with different settings of 1 to 20 servers.

Search Accuracy vs. Lossy Distribution: Fig. 9 shows the search accuracy distortion with respect to the vocabulary compression rate in lossy distribution. Fig. 10 shows the gain of time saving. The search accuracy is measured by mAP, and the lossy distribution is measured by the vocabulary size, as shown in the subfigures of Fig. 9. In both Oxford Buildings and 10-Million Landmarks, we have achieved comparable mAP with less than 10% visual words. As shown in the right subfigures of Fig. 9, by incorporating query log learning and lossy distribution learning, the overall search speed is further improved with limited mAP distortion but a relatively small vocabulary. And comparing to the state-of-the-arts [5], [6], [16], our lossy distribution achieves comparable or even better performance, due to our Boosting based word selection can keep the most discriminative words that effectively hinder a big distortion of image ranking results.

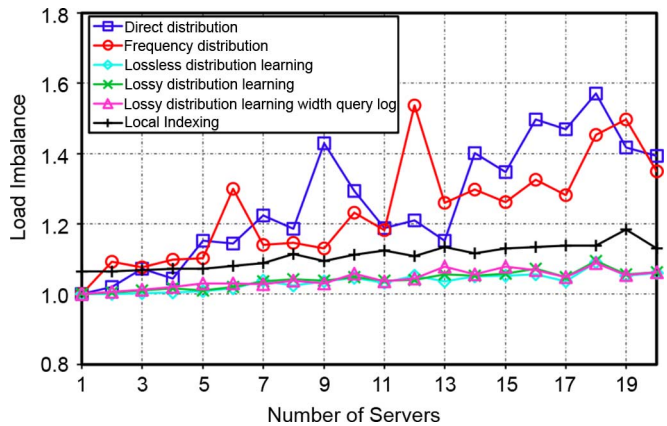
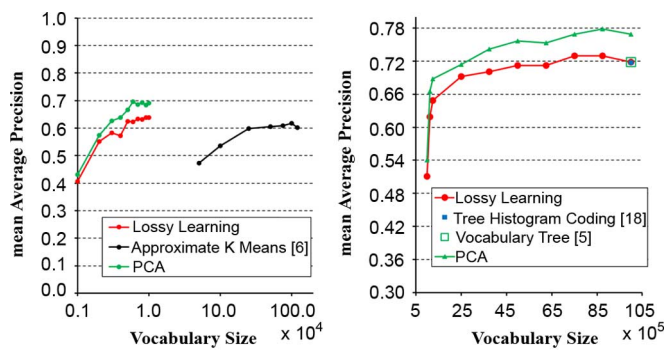


Fig. 8. Load imbalance comparisons in the 10-Million Landmarks dataset with respect to the number of servers in distributing vocabulary indexing. The comparing methods include: 1). Direct distribution (Method (3)), 2). Frequency distribution (Method (3)), 3). Lossless distribution (Method (4)); 4). Lossy distribution (Method (5)); 5). Lossy distribution with log (Method (6)); 6). Local Indexing (Method (1)). The performance is based on 500 concurrent queries on average, with different settings of 1 to 20 servers.



(a) (b)

Fig. 9. Retrieval mAP vs. Vocabulary sizes in (a) Oxford Building data set and (b) 10-Million Landmark dataset. Comparing methods in the above subfigures include: 1. Lossy Learning (Method (5)); 2. Vocabulary Tree [5]; 3. Approximate K Means [6]; 4. Tree Histogram Coding [16]; 5. PCA based unsupervised BoW feature space compression. Note that Approximate K Means on Oxford Buildings reported in [6] is also included for comparison. In our 10-Million Landmarks dataset, the state-of-the-art compact vocabulary scheme based on Tree Histogram Coding in [16] is also compared, as an alternative solution to "What to distribute".

In addition, as can be seen from the Fig. 9, PCA based selection is not as good as our boosting based word selection. In addition, dealing with million-scale visual vocabulary (containing 0.1 million visual words), PCA is much less

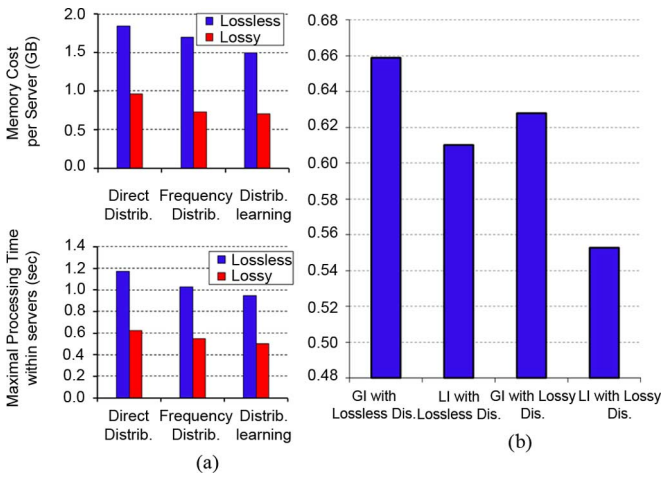


Fig. 10. (a) Comparison of maximal processing time over all servers. (b) mAP@10 when crashing 20% servers for both global indexing (GI) and local indexing (LI) Comparing methods in the above subfigures include: 1. (Lossless) Direct Distribution (Method (3)); 2. (Lossless) Frequency Distribution (Method (3)); 3. (Lossless) Learning based Distribution learning (Method (4)); 4. Direct Distribution + Lossy Boosting; 5. Frequency Distribution + Lossy Boosting; 6. Learning based Distribution + Lossy Boosting (Method (5)); The above experiments are conducted over our 10-Million Landmarks data set, in which the queries are selected according to the rule introduced at the beginning of Section IV. (For more details on how different methods work, please refer to the "Baseline" description in Page 9).

efficient than Boosting. And meanwhile, our solution is a supervised, linear feature selection while PCA is an unsupervised, non-linear feature compression.

Global Indexing vs. Local Indexing: Quantitative comparisons in Figs. 7 and 8 and Table I show that the "global indexing" performs better than the "local indexing". In Fig. 8, our global indexing achieves comparable performance of load imbalance to local indexing. Note that the local indexing is claimed to exhibit better performance of "load imbalance" in the traditional information retrieval systems [23]. However, as local indexing needs to dispatch all words into each server to complete search, it is less effective to handle the concurrent queries. In addition, by maintaining the entire one million vocabulary in each server, local indexing incurs more memory cost. Finally, it is worth mentioning that despite the comparisons in Figs. 7 and 8, and Table I, our "What to distribute" can be employed in both local and global indexing, as shown in Fig. 10.

When Partial Servers Crash: Finally, we validate the robustness of our learning based distribution. An empirical finding is that our distribution can also improve the robustness when partial servers were disconnected from the network due to crashing or other reasons. We simulate the scenario of crashing partial servers to test the performance of both our learning based distribution scheme and other alternatives. As shown in Fig. 9, our scheme achieves comparable performance to the case that all servers work well. This is attributed to that our scheme distributes words uniformly to all servers, and hence can essentially deal with the case of server crashing from a partial matching perspective (as one of the advantages of BoW based search). Unfortunately, for local indexing [23], to retrieve images in the crashed servers would definitely fail, since those images are not indexed in any other servers.

V. Conclusions and Future Work

We propose a novel learning based distributed vocabulary indexing architecture. As qualitatively evaluated, this architecture is suitable for scalable visual search by using standard PCs. To tackle the challenges in distributed indexing, we propose two essential issues, i.e., "What to distribute" and "How to distribute". We have addressed both issues from a machine learning perspective. We formulate "What to distribute" as the problem of lossy vocabulary Boosting, which discards "less meaningful" visual words prior to distributing them to multiple servers. We formulate "How to distribute" as the problem of learning a uniform distribution function, which exploits the concurrence statistics of visual words to distribute words of a given query as uniformly as possible, minimizing the overall search latency through multiple servers. Furthermore, to simulate the practice in online image search, we propose to impose the query history into the distribution function learning, which greatly improve search accuracy over popular image search engines with rich log of frequent queries. We have successfully deployed a real-world distributed image search system over a 10-million landmark image collection. Extensive comparisons have demonstrated the advantages in speedup and precision over the state-of-the-art vocabulary indexing [5], [6], [16] in either a single server or multiple servers [23]. More importantly, we have shown the robustness of global indexing; that is, our global indexing approach normally would not lead to the weakness of completely missing relevant images in "local indexing". Finally, our proposed learning based distributed indexing strategy is actually open to most existing visual vocabularies, e.g., [4]–[6], [16], [17], [34] to cope with the massive web image collections.

References

[1] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, 2004.
[2] K. Mikolajczyk, B. Leibe, and B. Schiele, "Local features for object class recognition," in *IEEE Conf. Computer Vision and Pattern Recognition*, 2005.
[3] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. Van Gool, "A comparison of affine region detectors," *Int. J. Comput. Vis.*, vol. 65, no. 1/2, pp. 43–72, 2006.
[4] J. Sivic and A. Zisserman, "Video Google: A text retrieval approach to object matching in videos," in *Proc. Int. Conf. Computer Vision*, 2003.
[5] D. Nister and H. Stewenius, "Scalable recognition with a vocabulary tree," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2006.
[6] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman, "Object retrieval with large vocabulary and fast spatial matching," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2007.
[7] G. Schindler and M. Brown, "City-scale location recognition," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2007.
[8] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Inf. Process. Manage.*, 1988.
[9] T. Hofmann, "Unsupervised learning by probabilistic latent semantic analysis," *Mach. Learn. J.*, 2001.
[10] D. Blei, A. Ng, and M. Jordan, "Latent dirichlet allocation," *J. Mach. Learn. Res.*, vol. 3, no. 4/5, pp. 993–1022, 2003.
[11] B. Kulis and K. Grauman, "Kernelized locality sensitive hashing for scalable image search," in *Proc. Int. Conf. Computer Vision*, 2009.
[12] H. Jegou, M. Douze, and C. Schmid, "Hamming embedding and weak geometric consistency for large scale image search," in *Proc. Eur. Conf. Computer Vision*, 2008.
[13] F. Jurie and B. Triggs, "Creating efficient codebooks for visual recognition," in *Proc. Int. Conf. Computer Vision*, 2005.

[14] J. Yang, Y.-G. Jiang, A. Hauptmann, and C.-W. Ngo, "Evaluating bag-of-words representations in scene classification," in *Proc. ACM Conf. Multimedia Information Retrieval*, 2007.
[15] H. Jegou, M. Douze, C. Schmid, and P. Perez, "Aggregating local descriptors into a compact image representation," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2010.
[16] D. Chen, S. Tsai, and V. Chandrasekhar, "Tree histogram coding for mobile image matching," in *Proc. Data Compression Conf.*, 2009.
[17] V. Chandrasekhar, G. Takacs, D. Chen, S. Tsai, R. Grzeszczuk, and B. Girod, "CHoG: Compressed histogram of gradients a low bit-rate feature descriptor," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2009.
[18] Y. Weiss, A. Torralba, and R. Fergus, "Spectral hashing," *Advances in Neural Inf. Process. Syst.*, 2008.
[19] T. Couvreur, R. Benzel, S. Miller, D. Zeitler, D. Lee, M. Singhai, N. Shivaratri, and W. Wong, "An analysis of performance and cost factors in searching large text databases using parallel search systems," *J. Amer. Soc.r Inf. Sci.*, vol. 45, no. 7, pp. 443–467, 1994.
[20] B.-S. Jeong and E. Omiecinski, "Inverted file partitioning schemes in multiple disk systems," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 6, no. 2, pp. 142–153, 1995.
[21] C. Stanfill, "Partitioned posting files: A parallel inverted file structure for information retrieval," in *Proc. ACM SIGIR Conf. Research and Development in Information Retrieval*, 1990.
[22] A. Moffat, W. Webber, and J. Zobel, "Load balancing for term-distributed parallel retrieval," in *Proc. ACM SIGIR Conf. Research and Development in Information Retrieval*, 2006.
[23] C. Badue and R. Yates, "Distributed query processing using partitioned inverted files," *Signal Process. Inf. Retrieval*, pp. 10–20, 2001.
[24] L. A. Barroso, J. Dean, and U. Hlzle, "Web search for a planet: The Google cluster architecture," *IEEE Micro*, vol. 23, no. 2, pp. 22–28, 2003.
[25] B. Kulis and K. Grauman, "Kernelized locality-sensitive hashing for scalable image search," in *Proc. Int. Conf. Computer Vision*, 1999.
[26] Y.-G. Jiang, C.-W. Ngo, and J. Yang, "Towards optimal bag-of-features for object categorization," in *Proc. ACM Conf. Content Based Image and Video Retrieval*, 2007.
[27] J. Gemert, C. Veenman, A. Smeulders, and J. Geusebroek, "Visual word ambiguity," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 7, pp. 1271–1283, 2009.
[28] F. Moosmann, B. Triggs, and F. Jurie, "Fast discriminative visual codebooks using randomized clustering forests," *Adv. Neural Inf. Process. Syst.*, 2006.
[29] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman, "Supervised dictionary learning," *Adv. Neural Inf. Process. Syst.*, 2008.
[30] S. Lazebnik and M. Raginsky, "Supervised learning of quantizer codebooks by information losss minimization," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 7, pp. 1294–1309, 2009.
[31] R. Ji, X. Xie, H. Yao, Y. Wu, and W.-Y. Ma, "Vocabulary tree incremental indexing for scalable location recognition," in *Proc. ICME*, 2009.
[32] R. Ji, H. Yao, X. Sun, B. Zhong, and W. Gao, "Towards semantic embedding in visual vocabulary," in *Proc. CVPR*, 2010.
[33] R. Ji, L.-Y. Duan, J. Chen, H. Yao, Y. Rui, S.-F. Chang, and W. Gao, "Towards low bit rate mobile visual search with multiple-channel coding," in *Proc. ACM Multimedia*, 2011.
[34] R. Ji, L.-Y. Duan, J. Chen, H. Yao, J. Yuan, Y. Rui, and W. Gao, "Location discriminative vocabulary coding for mobile landmark search," *Int. J. Comput. Vis.*, 2012.
[35] R. Maree, P. Denis, L. Wehenkel, and P. Geurts, "Incremental indexing and distributed image search using shared randomized vocabularies," in *Proc. ACM Conf. Multimedia Information Retrieval*, 2010.
[36] T. Yan, D. Ganesan, and R. Manmatha, "Distributed image search in camera sensor networks," in *Proc. ACM Conf. Embedded Network Sensor Systems*, 2008.
[37] R. Blanco and A. Barreiro, "Probabilistic static pruning of inverted files," *ACM Trans. Inf. Syst.*, vol. 28, no. 1, pp. 1–31, 2010.
[38] S. Buttcher and C. L. A. Clarke, "A document-centric approach to static index pruning in text retrieval systems," in *Proc. Annu. Int. ACM SIGIR Conf. Research and Development in Information Retrieval*, 2006.
[39] A. Ntoulas and J. Cho, "Pruning policies for two-tiered inverted index with correctness guarantee," in *Proc. Annu. Int. ACM SIGIR Conf. Research and Development in Information Retrieval*, 2007.
[40] A. Moffat, W. Webber, J. Zobel, and R. Baeza-Yates, "A pipelined architecture for distributed text query evaluation," *Inf. Retrieval*, vol. 10, pp. 205–231, 2007.
[41] M. Marin and V. Gil-Costa, "High-performance distributed inverted files," in *Proc. ACM Conf. Information and Knowledge Management*, 2007.
[42] C. Lucchese, S. Orlando, R. Perego, and F. Silvestri, "Mining query logs to optimize index partitioning in parallel web search engines," in *Proc. Int. Conf. Scalable Information Systems*, 2007.

Rongrong Ji (M'11) is currently a PostDoc Researcher at Electronic Engineering Department, Columbia University. He obtained his Ph.D. from the Computer Science Department, Harbin Institute of Technology. His research interests include: image retrieval and annotation, video retrieval and understanding. During 2007–2008, he was a research intern at Microsoft Research Asia, where he received Microsoft Fellowship 2007. During July 2010–November 2010, he was a visiting student at the Institute of Digital Media, Peking University. He has published papers in over 50 referred journals and conferences. He serves as a reviewer for IEEE Signal Processing Magazine, IEEE Transactions on Multimedia, SMC, TKDE, and ACM Multimedia conference *et al.* He is a member of the ACM.

Ling-Yu Duan (M'06) received the Ph.D. degree in Information Technology from The University of Newcastle, Australia, in 2007, the M.Sc. degree in Computer Science from the National University of Singapore, Singapore, and the M.Sc. degree in Automation from the University of Science and Technology of China, Hefei, China, in 2002 and 1999, respectively. Since 2008, he has been with Peking University, Beijing, China, where he is currently an Associate Professor with the School of Electrical Engineering and Computer Science. Dr. Duan is leading the group of visual search in the Institute of Digital Media, Peking University. Before that, he was a Research Scientist in the Institute for Infocomm Research, Singapore, from 2003 to 2008. His interests are in the areas of visual search and augmented reality, multimedia content analysis, and mobile media computing. He has authored more than 80 publications in these areas. He is a member of the ACM.

Jie Chen has been a Ph.D. candidate at the School of Electronics Engineering and Computer Science, Peking University, since 2010. He is working with Prof. Ling-Yu Duan in the Institute of Digital Media. His research topics include data compression and visual search, focusing on compact visual descriptors for large scale mobile visual search.

Lexing Xie (SM'11) received the B.S. degree from Tsinghua University, Beijing, China, in 2000, and the M.S. and Ph.D. degrees from Columbia University, in 2002 and 2005, respectively, all in electrical engineering. She is a Lecturer in the Research School of Computer Science at the Australian National University; she was with the IBM T.J. Watson Research Center, Hawthorne, NY from 2005 to 2010. Her recent research interests are in multimedia mining, machine learning and social media analysis. Dr. Xie has won several awards: the best conference paper award in IEEE SOLI 2011, the best student paper awards at JCDL 2007, ICIP 2004, ACM Multimedia 2005 and ACM Multimedia 2002.

Hongxun Yao (M'03) received the B.S. and M.S. degrees in computer science from the Harbin Shipbuilding Engineering Institute, Harbin, China, in 1987 and in 1990, respectively, and received the Ph.D. degree in computer science from Harbin Institute of Technology in 2003. Currently, she is a professor with the School of Computer Science and Technology, Harbin Institute of Technology. Her research interests include pattern recognition, multimedia processing, and digital watermarking. She has published 5 books and over 200 scientific papers.

Wen Gao (M'92–SM'05–F'09) received the Ph.D. degree in electronics engineering from the University of Tokyo, Tokyo, Japan, in 1991. He is currently a Professor of computer science with the Institute of Digital Media, School of Electronic Engineering and Computer Science, Peking University, Beijing, China. Before joining Peking University, he was a Professor of computer science with the Harbin Institute of Technology, Harbin, China, from 1991 to 1995, and a Professor with the Institute of Computing Technology, Chinese Academy of Sciences, Beijing.

Dr. Gao served or serves on the editorial boards for several journals, such as the IEEE Transactions on Circuits and Systems for Video Technology, IEEE Transactions on Multimedia, IEEE Transactions on Autonomous Mental Development, EURASIP Journal of Image Communications, and Journal of Visual Communication and Image Representation. He has chaired a number of prestigious international conferences on multimedia and video signal processing, such as IEEE ICME and ACM Multimedia, and also served on the advisory and technical committees of numerous professional organizations.