MOBILE VISUAL SEARCH VIA HIEVARCHICAL SPARSE CODING

Xiyu Yang[1], Lianli Liu[1], Xueming Qian[1*], Tao Mei[2], Jialie Shen[3], Qi Tian[4]

[1]SMILES LAB, Xi'an Jiaotong University, [2]Microsoft Research, Beijing, China,
[3]Signapore Management University, [4]University of Texas at San Antonia
qianxm@mail.xjtu.edu.cn; tmei@microsoft.com; jlshen@smu.edu.sg; qitian@cs.utsa.edu

ABSTRACT

Mobile visual search is attracting much research attention recently. Existing works focus on addressing the limited capacity of wireless channel yet overlook its instability, thus is not adaptive to the change of channel capacity. In this paper, a novel image retrieval algorithm that is scalable to various channel condition is proposed. The proposed algorithm contains three contributions: (1) to achieve instant retrieval under various channel capacity, we adjust transmission load by sparseness instead of codebook size; (2) we introduce hierarchical sparse coding into our retrieval workflow, where original codebook is transformed into a tree-structured dictionary which implies elements' priority; (3) we propose transmission priority ranking schemes that is adaptive to specific query. Experiment results show that the proposed algorithm outperforms BoW and Lasso based algorithm under different parameter settings. Retrieval results under different channel limitation validate the scalability of our method.

Index Terms- image retrieval, mobile, hierarchical, sparse coding, visual search

1. INTRODUCTION

With the fast development of mobile device, mobile visual search is receiving more and more attention. Typically, mobile visual search follows client-server architecture. By processing reference images, a codebook is generated and stored both in the server end and client end. BoW histogram of query image is coded using the codebook. Codebook entries are sent through wireless channel to the server end. The server end then reconstructs BoW histogram and proceeds with similarity search between query image and reference images as illustrated in Fig.1.

One big challenge of mobile visual search is the poor condition of wireless channel. By contrast to wired channel, wireless channel has smaller bandwidth and is more vulnerable to interference. Mobile visual search often suffers from unpredicted latency due to such instability, which degenerates customers' experience a lot.

To address the above problem, we aim to design a channel capacity self-adaptive algorithm that makes visual search scalable to various channel conditions. The proposed algorithm has to satisfy two requirements: 1) it should be able to retrieve image instantly given any bits of data; 2) to ensure satisfactory performance in low bit rate, it should be able to decide which part of data is most relevant to retrieval task and to be transmitted first.

In this paper, a novel image retrieval algorithm is proposed that meets the above two requirements. It includes three main contributions. Firstly, we reduce the amount of nonzero entries of codebook instead of the size of codebook to satisfy requirement of low bit rates. Secondly, we introduce hierarchical sparse coding into image retrieval for better performance and scalability. In addition, we propose ranking schemes for codebook entries to achieve scalable retrieval.

The rest of this paper is organized as follows. Section 2 briefly reviews related work in mobile visual search and scalable video coding. Section 3 formulates the problem of hierarchical sparse coding. Section 4 elaborates the method to decide entries' transmission priority. Experiment results are given in section 5 and section 6 concludes the paper.

2. RELATED WORK

Research on mobile visual search flourishes recently, e.g. [1] helped to refine the query, and [17]-[18] devoted to improving performance. To achieve low bit rate visual search, usually local descriptors is compressed such as CHoG [2], or the vocabulary codebook is processed as in [5]. To achieve satisfactory performance, the dimension of BoW histogram has to be rather high (usually 10K above). Therefore it is desirable to compress the BoW histogram.

*Corresponding author. The first two authors equally contribute to this paper. They are co-first authors.

This work is supported in part by National Natural Science Foundation of China (NSFC) Project No. 61173109, No. 60903121, No. 61128007, No. 61332018, and Microsoft Research. This work was supported in part to Dr. Qi Tian by ARO grant W911NF-12-1-0057 and Faculty Research Awards by NEC Laboratories of America.

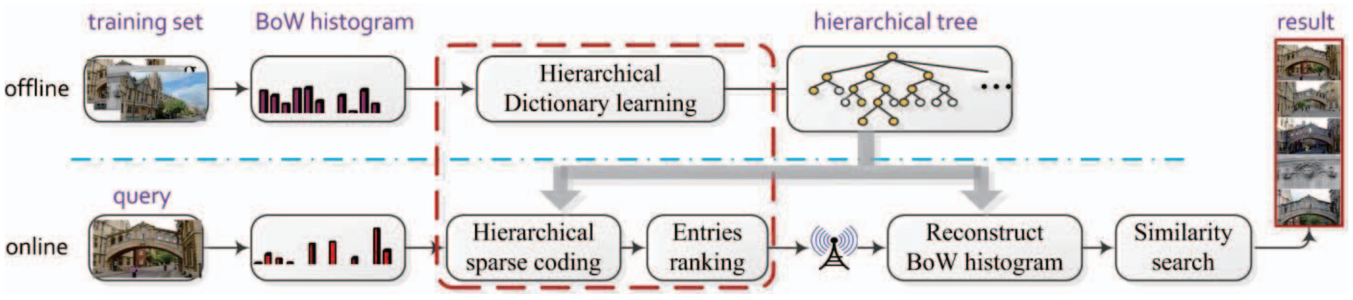


Fig.1. Work flow of scalable mobile visual search

Recent methods for BoW histogram compression fall into two categories: (1) Tree histogram based: transmitting the BoW histogram in a compact form. In [3], BoW histogram is encoded by the length of zero-runs between positive-count nodes of SVT [4]. And Chen et al. [5] prunes some trivial branches to shrink the scale of SVT; (2) sparse coding based: taking a post processing measure on BoW histogram by representing it as linear combination of a compact dictionary elements. Sparse coding schemes, e.g. Lasso [6] or elastic-net lasso [7], can learn the dictionary from original BoW codebook. Kinds of dictionary are proposed, e.g. Lin et al. generate a query-specific-codebook online by weighting a multiple codebooks [8] learned offline, and Ji et al. [9] present a Location Discriminative Vocabulary Coding framework for mobile landmark search.

Current works in mobile visual search, as described above, focus on promoting the performance within the limited capacity of mobile channel, e.g. Ji et al. introduce LCK [10] into Lasso regression, which improves the performance to some extent. However, most of them overlook the fact that mobile channel is unstable. Due to various factors, the capacity of mobile channel may vary a lot. Existing work achieves low bit rate visual search by reducing the dimension of codebook, while the fixed size of data makes image retrieval algorithm not self-adaptive to variant channel. When capacity of channel changes, a new codebook has to be generated for feature coding, otherwise, either latency or waste of channel resource will occur.

The idea of scalability is applied widely, e.g. scalable clustering is adopted for GPS estimation in [15] and [16], and the well-known SVT for scalable recognition [4]. Scalable Video Coding (SVC) has been prevailed for tens of years, such as spatial and quality scalability in H.262| MPEG-2 Video, H.263 and MPEG-4 Visual. H.264 includes temporal scalability besides. The SVC technology makes the length of video stream variable to satisfy the users' need in the condition of current channel condition. Inspired by SVC, we proposed a method that considers low bit rate visual search as reducing the number of nonzero entries of codebook. By ranking the contribution of entries, entries with latter rank are set to zero and not transmitted. Such scheme guarantees instant retrieval given arbitrary data size as well as good performance under low bit rate.

3. PROBLEM FORMULATION

We intend to compress the BoW histogram in a way basing on hierarchical sparse coding. As traditional sparse coding has done, we need learning a dictionary at first. Supposing there are N images in training set, for each image i, we extract the SIFT [11], and quantify the SIFT through a trained SVT which contains M leaf nodes, so the image i can be represented as a BOW histogram $x_i \in \mathbb{R}^M$. Then the N images in training set are presented as $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N] \in \mathbb{R}^{M\times N}$. Sparse coding based codebook compression learns a compression transform, i.e. a sparse representation $\boldsymbol{\alpha} \in \mathbb{R}^{p\times N}$ for $\mathbf{X}$ from dictionary $\mathbf{D} \in \mathbb{R}^{M\times p}$ by solving the following optimization problem:

$$(\hat{\mathbf{D}}, \hat{\boldsymbol{\alpha}}) = \arg\min_{\mathbf{D},\boldsymbol{\alpha}} L(\mathbf{X}, \mathbf{D}\boldsymbol{\alpha}) + \lambda T(\boldsymbol{\alpha}) \qquad (1)$$

where $T(\boldsymbol{\alpha})$ is regularization term, usually a norm, λ is the regularization parameter [12]. The definite equation of $T(\boldsymbol{\alpha})$ will be given in Section 4. $\hat{\mathbf{D}}$ and $\hat{\boldsymbol{\alpha}}$ denote the final trained $\mathbf{D}$ and $\boldsymbol{\alpha}$. $L(\mathbf{X}, \mathbf{D}\boldsymbol{\alpha})$ measures the reconstructing error which usually has the square form

$$L(\mathbf{X}, \mathbf{D}\boldsymbol{\alpha}) = \sum_{i=1}^{N} \|\mathbf{x}_i - \mathbf{D}\boldsymbol{\alpha}_i\|_2^2 \qquad (2)$$

where $\boldsymbol{\alpha}_i \in \mathbb{R}^p$ is a p dimensional compressed BoW histogram of image i, and $\|\bullet\|_2^2$ is the square of l_2 norm.

The second item $T(\boldsymbol{\alpha})$ introduces sparse penalty. With the increase of penalty parameter λ, the number of nonzero entries will decrease.

Given a query image q with its feature $\mathbf{x}_q$, $\mathbf{x}_q$ will be coded into a more compact descriptor $\boldsymbol{\alpha}_q$ by solving

$$\hat{\boldsymbol{\alpha}}_q = \arg\min_{\boldsymbol{\alpha}} L(\mathbf{x}_q, \mathbf{D}\boldsymbol{\alpha}_q) + \lambda T(\boldsymbol{\alpha}_q) \qquad (3)$$

Usually, $\boldsymbol{\alpha}_q$ will be sent through wireless channel to the server end. To achieve scalability, however, we need to rank the contribution of dictionary entries to decide which entries should be transmitted with priority. When channel capacity is limited, those nonzero entries not transmitted will be set to zero, thus we have a coding vector $\boldsymbol{\alpha}_s$ sent to the server end with smaller data size. Ideally, $\boldsymbol{\alpha}_s$ should be the solution of Eq. (3),

$$\hat{\boldsymbol{\alpha}}_s = \arg\min_{\boldsymbol{\alpha}} L(\mathbf{X}, \mathbf{D}\boldsymbol{\alpha}_s) + \lambda' T(\boldsymbol{\alpha}_s), \text{ where } \lambda' > \lambda \qquad (4)$$

To assure scalability, we cannot resolve Eq. (3) for a narrow channel. Instead, we need to "predict" its solution for Eq. (4) by analyzing the property of $\boldsymbol{\alpha}$.

Traditional sparse coding algorithm, such as Lasso, formulates $T(\boldsymbol{\alpha})$ as L_1 regularization. However, L_1 regularization regards each dictionary element as independent, while BoW code words are often correlated. Such independence assumption makes it infeasible to analysis the contribution of each dictionary element for retrieval task. If a group of elements are highly correlated, L_1 regularization only selects some randomly. To achieve scalable visual search, we need to incorporate side information into sparse coding to guide priority determination.

4. SCALABLE RETRIEVAL VIA HIERARCHY

4.1 Hierarchical sparse coding

As pointed out in Section 1, achieving low bit rate search by reducing the size of codebook is not scalable to various channel conditions. Sparse coding based methods have proved the sparseness of code words. Therefore, it is justifiable to transmit part of nonzero entries of $\boldsymbol{\alpha}_i$ only.

To model dependencies between dictionary elements, the idea of hierarchical sparse coding has been proposed and achieves impressive performance in various tasks such as image processing and text model [12]. Hierarchical sparse coding embeds a tree structure in a dictionary, as shown in Fig. 2. The nodes in higher level contribute more than that in lower level, i.e. the father nodes contain principal information, and the child nodes characterize the details. So the entries of $\boldsymbol{\alpha}_i$ corresponding to the higher level should have the priority to be transmitted. In addition, the nodes in same group, namely in common sub-tree, correlate tighter. E.g. in Fig. 2, the node 3 has closer relationship with node 4 than with 6, and it just depend on its ancestors 2 and 1 rather than 5.

Dictionary elements form overlapped groups according to the tree structure: given a node (element) j in tree F, node j forms a group g_j that contains itself and all its descendants, as illustrated in Fig. 2.

Given grouping information, hierarchical sparse coding penalizes elements in groups. Following the notations in [12], penalty function $T(\boldsymbol{\alpha})$ in Eq. (1) can be written as:

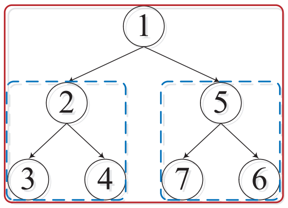


Fig. 2. Illustration of grouping in tree-structured dictionary. Node 1-7 form a large group g_1, 2-4 and 5-7 compose small groups g_2 and g_5 respectively which are overlapped with g_1. In fact, each of the nodes 3,4,6,7 can also be a group.

$$T(\boldsymbol{\alpha}) = \sum_{g \in G} \left\| \alpha_{|g} \right\| \qquad (5)$$

where G is the set of groups g, $\alpha_{|g}$ indicates the entries of codebook, namely dictionary, that belong to group g, and $\|.\|$ is often set as L_2 norm.

For image retrieval task, similar with the work in [10], we embed visual discriminability of BoW code words into coding. The visual discriminability $\mathbf{w} = \{w_1, \cdots, w_M\}$ is determined by the term frequency and inverted file frequency of each code word, as given in Eq.(6).

$$w_i = \frac{n_i'}{n'} \times \log\left(\frac{N}{n_i}\right) \qquad (6)$$

where n_i' is the number of features(SIFT) quantized into visual word c_i, n' is the total number of features in training images. N is the size of training set identical to N in Eq. (2), and n_i is the number of images that contain word c_i. Thus Eq. (2) should be redefined as Eq. (7)

$$L(\mathbf{X}, \mathbf{D}\boldsymbol{\alpha}) = \sum_{i=1}^{N} \left\| \mathbf{w}\mathbf{x}_i - \mathbf{D}\boldsymbol{\alpha}_i \right\|_2^2 \qquad (7)$$

Solving Eq. (1) generally requires iterative update of $\mathbf{D}$ and $\boldsymbol{\alpha}$. Following the work in [13], the process is described in **Algorithm 1**. Since it's unviable to solve the jointly convex optimization of $\mathbf{D}$ and $\boldsymbol{\alpha}$ at the same time, we update them alternately. Firstly, we make $\mathbf{D}_{t-1}$ constant, optimize $\boldsymbol{\alpha}$ by minimizing the objective function Eq. (1). Then keep $\boldsymbol{\alpha}$ fixed, compute $\mathbf{D}_t$. We reiterate updating $\mathbf{D}$ and $\boldsymbol{\alpha}$ until iteration times reaches the maximum iterations T_{iter} (T_{iter} is about 1000 empirically) or $|\mathbf{D}_t - \mathbf{D}_{t-1}|$ is less than the threshold *th* (*th*=0.01 in this paper) .

By setting $\|\bullet\|$ as L_2 norm, hierarchical sparse coding has one important property suitable for scalable visual search: for an element j and its descendants j_d, entries of descendants are penalized more times than that of ancestors, i.e. the entries of j_d will be set to zero ahead of entries of j. Hence j should be given transmission priority over j_d. Denoting their respective ranking as R_j and R_{jd}, we have $R_j < R_{jd}$ .Therefore, by incorporating tree structure information into coding process, hierarchical sparse coding provides a direct guidance for priority determination.

```
Algorithm 1 Hierarchical Sparse Coding
1. Input: BoW histogram of training images
      $\mathbf{X} = [\mathbf{x}_1, \dots \mathbf{x}_n]$ ,maximum iterations $T_{iter}$ , stop
      criterion $th$ , tree structure $G$
2. Initialization: dictionary $\boldsymbol{D}$ is initialized by
      randomly selected $k$ columns from $\mathbf{X}$, parameter
      initialization $\mathbf{A} = \mathbf{0}, \mathbf{B} = \mathbf{0},\ t = 1$
3. while $|\mathbf{D}_t - \mathbf{D}_{t-1}| < th$ or $t < T_{\max}$ do
4.    Sparse coding:
      $\boldsymbol{\alpha}_t = \arg\min_{\boldsymbol{\alpha}} \|\mathbf{wX} - \mathbf{D}_{t-1}\boldsymbol{\alpha}\|_2 + \lambda \sum_{g \in G} \|\boldsymbol{\alpha}_{|g}\|_2$
5.    Parameter Update:
      $\mathbf{A}_t \leftarrow \mathbf{A}_{t-1} + \alpha_t \alpha_t^T, \mathbf{B}_t \leftarrow \mathbf{B}_{t-1} + x_t \beta_t^T$
6.    Dictionary Update
      $\mathbf{D}_t = \arg\min_{\mathbf{D}} \left(1/2\mathrm{Tr}\left(\mathbf{D}^T \mathbf{D} \mathbf{A}_t\right) - \mathrm{Tr}\left(\mathbf{D}^T \mathbf{B}_t\right)\right)$
7.    $t = t + 1$
8. end
9. Output: dictionary $\mathbf{D}$, coding result $\boldsymbol{\alpha}$
```

4.2 Codebook entry selection via hierarchy

Our hierarchical dictionary is constructed by multiple parallel groups. The group mentioned in this section refers to the whole tree structure as enclosed in outmost red box in Fig.2. According to the tree structure, two strategies, level-based ranking and group-based ranking schemes, can be applied to select the entries to be transmitted.

In level-based ranking algorithm, we transmit elements level by level from top to bottom no matter which group they are in. In group-based ranking algorithm, elements are sent over group by group. We firstly visit group with highest priority from top to bottom. Then the algorithm returns to the top and starts the visit of group with second priority. Group's priority is decided by the num of nonzero elements in the group, more nonzero elements, higher priority. Consider two elements, element j and element k that are two ancestor elements, each belongs to a group g_j and g_k , their children are s_j and s_k. If both j and k are selected, the penalty of s_j and s_k are in the same magnitude. Therefore the selection of s_j and s_k depends on their contribution to the first item of Eq. (3). When s_j is selected while s_k is kept zero, which means s_j contributes more than s_k, we can infer that g_j is more important to reconstruct the original BoW Histogram $\mathbf{x}_q$ than g_k considering the correlation between group elements. More generally, when a group of elements tend to be selected a lot, it means that this group is highly relevant to the search task. Therefore this group should be given priority over other groups.

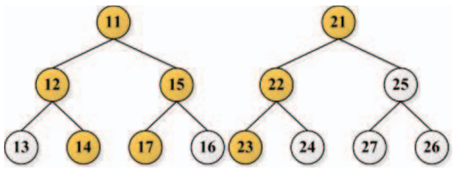


Fig.3. Ranking for elements without ancestor-descendant relationship: dark part indicates selected elements. As element 11's group has 5 elements selected while element 21's has only 3, in group-based ranking method, group g_{11} will be selected before group g_{21}; in level-based ranking scheme, element 12 and 15 will be prior over 22.

Denoting n_{g_j} and n_{g_k} as the number of nonzero elements in g_j and g_k, if $n_{g_j} > n_{g_k}$, we have $R_{g_j} < R_{g_k}$.

Tree structure offers a direct way to rank entries that are in an ancestor-descendant relation. For elements in same group, it's explicit that ancestor elements rank before descendants. As for elements in the same level in level-based ranking, two cases need to be taken into account. On the one hand, for elements in the same level not in same group we rank them in terms of the number of nonzero entries in their respective large group surrounded by the outermost red frame like in Fig.2. Assuming that j and k are two elements in this situation, as elements in the same group are strongly related, determining the significance of j and k can be treated as ranking g_j and g_k. If $R_{g_j} < R_{g_k}$, element j should be ranked before k. On the other hand, the elements in the same level as well as same group are put on an equal footing, so their rankings are just same as the traversal order. An illustration of the two schemes is given in Fig.3. Performance of these two methods will be compared and discussed in the experiment section.

Whether in level-based ranking or group-based ranking scheme, two logical vectors $\mathbf{F}_L$ and $\mathbf{F_G}$ mark which groups and levels are transmitted separately. $\mathbf{F}_L \in \mathbb{R}^l$, l is the num of layers in a whole group, and $\mathbf{F}_G \in \mathbb{R}^c$, c is the num of groups that the dictionary contains. $\mathbf{F}_L(i) = 1, i = 1, 2, \cdots, l$, indicates that elements in level i are transmitted. Similarly, $\mathbf{F}_G(i) = 1, i = 1, 2, \cdots, c$ denotes that entries in group i are sent to server end. Associating $\mathbf{F}_L$ with $\mathbf{F_G}$, we can know the exact entries' position available in server end.

Therefore, given a query image coded by a tree-structured dictionary, our algorithm ranks the codebook entries according to the following rules:

Given two elements j and k,

- If k is the decedent of j, then $R_j < R_k$.
- If k and j are in the same level, and $n_{g_j} > n_{g_k}$, then $R_j < R_k$.
- If k and j are neither in the same group nor the same level, for group-based scheme, ranking will be given

according to n_{g_j} and n_{g_k} ; for level-based scheme, ranking will be given according to the levels they are in.

Nonzero entries are transmitted according to their rankings until channel limitation is reached, while those not sent are set to zero. The server end reconstructs the BoW histogram and carries out similarity search. When channel capacity grows, the mobile end simply needs to transmit more nonzero entries and the server can perform retrieval again to refresh results. There is no need to learn new dictionary or coding again, thus instant retrieval scalable to channel band-width is achieved.

5. EXPERIMENT

We conduct our experiments on The Oxford Buildings Dataset. The scalable vocabulary tree (SVT) is learned on the dataset, including 4 layers and 8623 leaf nodes in total. The SIFT feature is extracted and then quantized to the 8623 visual words. Thus every image is represented as a 8623 dimensional BoW histogram. We select 3795 images' BoW histograms as training set randomly. The hierarchical dictionary is trained on the training set with **Algorithm 1**. In this paper, the dictionary comprises 100 parallel groups, and each group is 4 layers' binary tree architecture with 15 nodes, i.e. the dictionary have 1501 elements, appended a common ancestor node for all the groups.

With the trained dictionary, compact descriptor $\boldsymbol{\alpha}_{\mathrm{q}}$ can be attained by solving the Eq. (3). Group-based ranking and level-based ranking schemes are utilized to decide which elements of $\boldsymbol{\alpha}_{\mathrm{q}}$ will be transmitted. The rate of nonzero entries varies from 10% to 100% with an interval of 10%.

5.1 Dataset

The Oxford Buildings Dataset consists of 5062 images collected from Flickr by searching for particular Oxford landmarks, 11 landmarks in total. For each landmark, 5 possible queries, and relevant images in three different quality grades signifying whether the object is clearly visible, good, ok and junk are given.

5.2 Evaluation Criterion

Two evaluation criterions, MAP and Precision at K documents retrieved (P@K) assess the retrieval performance. We use MAP because it considers the rank of relevant documents retrieved. MAP is defined following [14]. For a single query, AP is defined as:

$$AP = \frac{\sum_{j=1}^{N} P(j) \cdot R(j)}{\sum_{j=1}^{N} R(j)} \tag{8}$$

where, j is the rank, $R(j)$=1 only if the j-th retrieved image is relevant result, $R(j)$=0 otherwise. N is the size of dataset.

$$P(j) = P@j = \frac{num\ of\ relevant\ results\ up\ to\ j}{j} \tag{9}$$

The MAP is mean average precision denoted as:

$$MAP = \frac{1}{K}\sum_{k=1}^{K} AP(k) \tag{10}$$

where K denotes the num of queries. We select 308 images from good and ok categories in Oxford Buildings Dataset as queries, so K=308.

5.3 Baselines

1) BoW based image retrieval: We compare the performance of using our 1501 dimensional compressed BoW histogram with using the original 8623 dimensional BoW histogram. The query's and dataset images' BoW histograms are all be normalized before similarity search.

2) Sparse coding with Lasso regression [10]: The dictionary trained via Lasso regression has the identical size with our dictionary embedded a tree structure in. Owing to its irregular distribution of nonzero elements, it is difficult to judge which entries are more important. To compare it with our scalable transmission, two methods can be used to rank the elements in Lasso dictionary. Firstly, IDF-based ranking, in which elements with high IDF rank top. IDF is computed over the training set as follows:

$$IDF(i) = \log\frac{N}{en_i}, i = 1, 2, \cdots, p \tag{11}$$

where, en_i is the num of images whose i-th entry to the dictionary is nonzero. p=1501 in this paper, denotes the dimension of compressed BoW histogram. The second manner is random ranking. To be fair, these two schemes will be applied to our hierarchical dictionary as well.

5.4 Performance show

Comparing with the original BoW histogram, we use all the 1501 entries of hierarchical dictionary and Lasso dictionary (λ=0.002) to reconstruct the 8623 dimensional BoW histogram for retrieval in the server end. MAP is the standard to evaluate the retrieval performance shown in Tab. 1, and the precision from top 10 to top 100 is given in Fig. 4. It's obvious that sparse coding performs better, for compressed histogram can eliminate inherent redundancy inside the original histogram.

Next, we compare our proposal with Lasso changing the rate of entries transmitted from 10% to 100%. Fig. 5 (a) shows that embedding tree structure in the dictionary performs slightly over that learned via Lasso regression when the parameter λ that controls the sparseness is set as 0.002. Then, we alternate λ to 0.02, Fig. 5 (b) demonstrates that, the retrieval precision will decline when λ increases, because more entries are compelled to be zero. That means the term $L(\mathbf{x_q}, \mathbf{D}\boldsymbol{\alpha}_\mathbf{q})$ in Eq. 3 increases, i.e. there is bigger loss in $\mathbf{x}_q'$. Nevertheless, hierarchical sparse coding still acts better than Lasso based sparse coding. Especially, when MAP of Lasso method declines seriously, MAP of our hierarchical method just decreases a little.

Tab. 1. MAP of using compact BoW histogram by hierarchical, Lasso sparse coding and uncompressed original BoW histogram

method	hierarchical	Lasso	uncompressed
MAP	0.4155	0.3833	0.3235

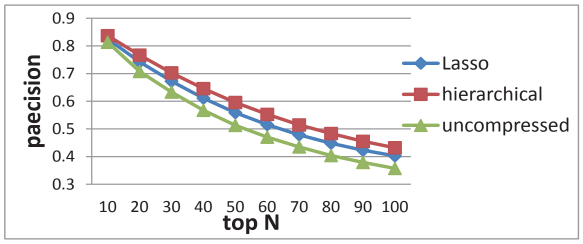
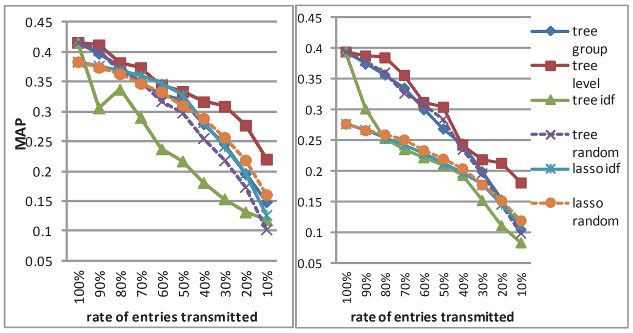


Fig.4. Precision at top N



(a) (b)

Fig. 5. MAP corresponding to ratio of entries ranging from 100% to 10%. In (a) λ=0.002 and in (b) λ=0.02. The legend "tree" and "lasso" denote hierarchical and Lasso based sparse coding respectively, and "group", "level", "idf", and "random" stand for the four aforementioned ranking schemes in the paper.

Tree-structure defines a kind of ancestor-descendant relationship between elements in same group, which determines the importance of element simultaneously. Entries are no more set to zero at random like in Lasso regression, therefore the level-based ranking and group-based ranking schemes are more rational than IDF-based and random ranking on a whole. The experiment results in Fig.5 (b) and Fig.5 (b) can manifest the inference. And it's obvious level-based ranking performs superior than group-based ranking scheme, which verifies that ancestor elements contribute more than descendants no matter which groups they are in.

6. CONCLUSION

We have proposed a novel hierarchical sparse coding for mobile image search, which can not only reduce the data size to be transmitted, but also remove part redundant code words to improve the retrieval precision. Moreover, on the basis of tree-structured dictionary, we further achieved scalable search that can adjust the data size to the channel capacity for better user experience. More experiments on larger scale vocabulary and image dataset will be conducted in our future work. And we will seek a better representation of query to improve performance like in [19].

7. REFERENCES

[1] H. Li, Y. Wang, T. Mei, J. Wang, S. Li, "Interactive Multimodal Visual Search on Mobile Device," IEEE Trans. on Multimedia, vol.15, no.3, pp. 594-607, 2013.

[2] V. Chandrasekhar, G. Takacs, D. Chen and S. Tsai, "CHoG: Compressed histogram of gradients A low bit-rate feature descriptor," CVPR, 2009.

[3] D. Chen, S. Tsai, V. Chandrasekhar and G. Takacs, "Tree histogram coding for mobile image matching," DCC, pp. 143-152, 2009.

[4] D. Nistér and H. Stewénius, "Scalable Recognition with a Vocabulary Tree", CVPR, vol. 2, pp. 2161-2168, 2006.

[5] J. Chen, L. Y. Duan, R. Ji, W. Gao, "Pruning Tree-Structured Vector Quantizer Towards Low Bit Rate Mobile Visual Search," ICASSP, pp. 965-968, 2012.

[6] R. Tibshirani, "Regression shrinkage and selection via the Lasso," J. Roy. Stat. Soc., vol. 58, no. 1, pp. 267–288, 1996.

[7] H. Zou and T. Hastie, "Regularization and variable selection via the elastic net," J. Roy. Stat. Soc., vol. 67, no. 2, pp. 301-320, Apr. 2005.

[8] J. Lin, L. Y. Duan, J. Chen and R. Ji, "Learning multiple codebooks for low bit rate mobile visual Search," ICASSP, pp. 933-936, Mar. 2012.

[9] R. Ji, L. Y. Duan, J. Chen, H. Yao, Y. Rui and W. Gao, "Location discriminative vocabulary coding for mobile landmark search," Int. J. Comput. Vis., vol. 96, pp. 290-314, Feb. 2012.

[10] R. Ji, H. Yao, W. Liu and X. Sun, "Task-Dependent Visual Codebook Compression," IEEE Transaction on Image Processing, vol. 21, no. 4, pp. 2282-2293, 2012.

[11] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," Int. J. Comput. Vis., vol. 60, no. 2, pp. 91-110, 2004.

[12] R. Jenatton, J. Mairal, G. Obozinski and F. Bach, "Proximal methods for sparse hierarchical dictionary learning," ICML, 2010.

[13] J. Mairal, F. Bach, J. Ponce and G. Sapiro, "Online dictionary learning for sparse coding," ICML, pp. 689-696, 2009.

[14] A. Turpin, F. Scholer, "User Performance versus Precision Measures for Simple Search Tasks," SIGIR, pp. 11-18, 2006.

[15] J. Li, X. Qian, Y. Tang, L. Yang, and T. Mei, "GPS estimation for places of interest from social users' uploaded photos," IEEE Trans. Multimedia 2013, vol. 15, no. 8, pp. 2058-2071.

[16] J. Li, X. Qian, Y. Yan Tang, L. Yang, and C. Liu, "GPS estimation from users' photos", In Proc. MMM 2013, pp. 118-129

[17] Z. Cheng, J. Ren, J. Shen, H. Miao, "Building a Large Scale Test Collection for Effective Benchmarking of Mobile Landmark Search," MMM (2) 2013, pp. 36-46

[18] Y. Xue, X. Qian, B. Zhang, "Mobile image retrieval using multi-photos as query," ICMEW, 2013.

[19] J. Shen, D. Tao, X. Li, "QUC-Tree: Integrating Query Context Information for Efficient Music Retrieval," IEEE Trans. on Multimedia 2009, vol. 11, no. 2, pp. 313-323