Vocabulary Hierarchy Optimization for Effective and Transferable Retrieval

Rongrong Ji[1], Xing Xie[2], Hongxun Yao[1], Wei-Ying Ma[2]
[1]Harbin Institute of Technology
No.92, West Dazhi Street, Harbin, 150001, Heilongjiang, P. R. China
{rrji,yhx}@hit.edu.cn
[2]Microsoft Research Asia
No.49, Zhichun Road, Haidian Distinct,100190 Beijing, P. R. China
{xingx, wyma}@Microsoft.com

Abstract

Scalable image retrieval systems usually involve hierarchical quantization of local image descriptors, which produces a visual vocabulary for inverted indexing of images. Although hierarchical quantization has the merit of retrieval efficiency, the resulting visual vocabulary representation usually faces two crucial problems: (1) hierarchical quantization errors and biases in the generation of "visual words"; (2) the model cannot adapt to database variance. In this paper, we describe an unsupervised optimization strategy in generating the hierarchy structure of visual vocabulary, which produces a more effective and adaptive retrieval model for large-scale search. We adopt a novel Density-based Metric Learning (DML) algorithm, which corrects word quantization bias without supervision in hierarchy optimization, based on which we present a hierarchical rejection chain for efficient online search based on the vocabulary hierarchy. We also discovered that by hierarchy optimization, efficient and effective transfer of a retrieval model across different databases is feasible. We deployed a large-scale image retrieval system using a vocabulary tree model to validate our advances. Experiments on UKBench and street-side urban scene databases demonstrated the effectiveness of our hierarchy optimization approach in comparison with state-of-the-art methods.

1. Introduction

Retrieving objects and scenes in natural images poses significant technical challenges because of the complex variations in real world environments, such as backgrounds, viewpoints, illuminations, scales, and orientations. Recent advance in local feature descriptors [1][2] facilitates the Bag-of-Visual-Words (BoW) image representation in image retrieval [3], video search [6], scene recognition [4][5] and object categorization [2][14][19]. In such case, a visual vocabulary is built to transfer local features into "visual words" for database inverted indexing, and to represent an image as a BoW vector to convert the image search task into the classical document retrieval scenario. Based on visual vocabulary representation, we can refer to document analysis methods, such as TF-IDF [7], pLSA [8] and LDA [9], to improve retrieval performance.

Building visual vocabulary to translate image local patches [1][2] to "visual words" is the most crucial step in patch-based retrieval. Many large-scale retrieval applications usually involve gigantic databases. To ensure efficient online search in these cases, hierarchical quantization is usually adopted to generate visual words. The dominant methods [3][5][6][10] of generating "visual words" for efficient online search are hierarchical quantization approaches, such as Vocabulary Tree (VT)[3], Approximate K-Means (AKM)[10], K-D Tree[1], and their variants [11][12][13][20][22]. Typically, these approaches quantize image descriptors using a hierarchical subspace division (such as hierarchical k-means clustering) to produce visual words. As the basic component in the visual vocabulary, each quantized visual word contains the descriptors that are closer to its feature center than to others in the feature space.

Since the visual vocabulary generates the visual words by hierarchically dividing the feature space, it closely associates image retrieval performance with its hierarchical structure. Such a hierarchical structure contains the genesis of many crucial problems in patch-based visual retrieval, such as quantization errors [13], term weighting efficiency [3], middle-level effectiveness [3], and model flexibility. However, in-depth optimization of this hierarchical construction process is left without consideration in former papers [3][13][12] [11][20]. Recent research has already revealed the negative effect of hierarchical quantization bias in visual words generation. For instance, Philbin et al. [22] adopted visual words fuzzy matching to alleviate the negative impacts of hierarchical quantization in AKM. Nister et al. [3] combined the middle-level nodes of VT into a unified BoW vector for ranking similarities. Yang [12] investigated the efficiency of vocabulary size, term weighting, and stop-word removal issues in PASCAL and TRECVID. However, to the best of our knowledge, how to optimize the hierarchical vocabulary structure during the quantization process have not yet been considered in the literature. Current methods usually refer to refining the visual words in the lowest-level to improve retrieval [1][2][4][5][6], without regard to a feasible way to directly

This work was partially supported by National Key Basic Research Program of China (No.2009CB320905) and Natural Science Foundation of China (No.60472043).

optimize the vocabulary hierarchy during construction. In texton codebook generation, Jurie et al. [13] adopted scalable acceptance-radius clustering to refine k-means clustering bias in texton generation. Jegou et al. [21] investigated a two-layer clustering scheme together with a CDM distance measurement to learn a similarity metric in building visual vocabulary. However, the effectiveness of these methods in deeper hierarchies and scalable databases is restricted, especially for the large-scale applications with millions of descriptors. Learning-based word selection is another solution [11][20] in a supervised scenario: Wang et al. [11] proposed a codeword selection strategy by boosting among leaf nodes. Leung et al. [20] investigated mutual information, odds ratio and linear SVM to optimize the original codebook. However, due to the training example restriction, a supervised learning strategy is unsuitable for the large-scale retrieval tasks. Further problems come from their generality, in which the learned visual vocabularies and similarity measurements cannot be directly reapplied to a new database or maintained in an incremental database.

It is well-known that the textual words have no well-defined natural hierarchy. In contrast, the visual words are strongly related to the hierarchy of vocabulary. In hierarchical quantization, not only the lowest-level words but also the middle-level nodes are generated. In other words, not only the visual words representations but also their higher-level abstractions (can be analogized as phrases in text) are obtained. Consequently, a natural question is: Can we optimize and explore the vocabulary hierarchy to produce better visual words and ranking mechanism?

A first attempt to exploit middle-level nodes in similarity ranking come from Grauman et al. [23], which adopted pyramid matching to measure the similarity of high dimensional data with a Mercer kernel for image categorization. However, work in [23] quantized space by fixed division, which restricted its flexibility to adapt to data variances. Nister et al. [3] combined vocabulary middle-levels to produce better rankings. However, improvements were not significant due to the original un-optimized vocabulary hierarchy.

In this work, we present an unsupervised hierarchy optimization approach for vocabulary construction. We aim to optimize and exploit vocabulary hierarchy to achieve fast speed, excellent precision, and good flexibility in a unified framework. We deploy our improvements on the vocabulary tree model [3], which is a representative retrieval model with hierarchical quantization in visual vocabulary generation. Our approach could also be applied to other retrieval models such as the k-means clustering phase in AKM, or as a modification phase in scalable acceptance-radius clustering [13].

In particular, we first introduce a Density-based Metric Learning (DML) to unsupervisedly refine the similarity metric in the hierarchical clustering. DML makes the middle levels really powerful and meaningful in retrieval. Second, based on optimized vocabulary, we treat the quantization hierarchy in a "vertical-order" candidate rejection chain [14] to improve similarity ranking. Third, we propose a "VT Shift" algorithm to enable our retrieval model to be transferable across different databases by adapting the optimized vocabulary hierarchy for database variances. In a large-scale application scenario, our VT Shift algorithm can maintain good performance for database variance without model re-generation cost.

The rest of this paper is organized as follows: Section 2 discusses our insights into how the quantization hierarchy affects the retrieval performance. Section 3 presents our DML algorithm for hierarchy optimization. Section 4 investigates a "VT Shift" algorithm to transfer the optimized retrieval model across different databases. Experimental comparisons with state-of-the-art methods [3][5][10] are presented in Section 5.

2. A Close Look at Hierarchical Quantization

Compared with single-level quantization, using hierarchical quantization to generate and access the visual vocabulary can greatly accelerate retrieval, which is crucial in large-scale applications [3][5]. To produce a BoW vector for an image using the VT model (w branches, m words), the time cost is $O(w \times log_w(m))$, whereas using a single-level word division the time cost is $O(m)$. However, there are drawbacks to gain such acceleration. Here we discuss two issues in hierarchy structure:

2.1. How Hierarchy Affects Retrieval

Similarity Metric Bias: To produce good visual words, the denser regions in feature distribution should correspond to more generic image patches (Fig.10 left), and the sparser regions should be more discriminative (Validated in Fig.1). In TF-IDF term weighting [3][7], a visual word obtains less weight when it appears in more images and vice versa. However, earlier papers [13][16] have shown that the k-means process will asymmetrically divide feature space, which moves clusters to denser regions because of its "mean-shift"-like update rule. Moreover, we have found

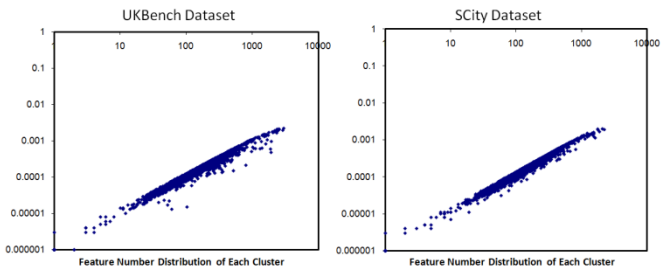

Figure 1: Feature-word frequency distribution (in log-log plot) of the finest level in hierarchical quantization. The fewer features a cluster has, the smaller the portion of images it contains, and hence it is more discriminative in the *BoW* vector.

out that the hierarchical quantization process in vocabulary generation iteratively magnifies such asymmetric division. More hierarchy levels lead to more asymmetric metrics in clustering, which causes the words' distributions biased to the denser regions and over-fit to the feature density. As a result, the discriminative patches (the sparser regions that rarely appear in images) would be coarsely quantized (given lower IDF and contribute less to ranking), while the general patches (the denser regions that frequently appear in many images) would be finely quantized (gain an inappropriately higher IDF in ranking). This is exactly the reason that IDF shows limited improvement in [6][12], which is caused by the similarity metric biases that are magnified by hierarchical clustering.

Furthermore, the nearest neighbor search within leaf nodes is inaccurate because of its locality nature in the vocabulary structure. By hierarchical quantization, the nearest neighbor search in leaf nodes becomes more inaccurate as the increase of nearest neighbor scale.

Tab.1 presents our validation of this assessment. We investigated the quantization errors in a 3-branch, 5-level vocabulary tree. In Tab.1, NN means the nearest neighbor search scope and GNP 1-5 means the numbers of branches we paralleled in GNP [5] search extension. We selected 3K images from our urban street scene database (introduced in Sec.5) to form a VT, with 0.5M features (average 2K features in each visual word). We compared the matching ratio between the global-scale NN and leaf-scale NN, in which the global-scale is the overall feature space while the leaf-scale is inside leaf nodes. We extended the leaf-scale to include more local neighbors using Greedy N-best Path (GNP) [5]. The quantization error was evaluated by the matching ratio (%) to see to what extent the VT quantization would cause mismatch of feature points. From Tab.1, the match ratios between the inside-leaf and the global-scale search results are extremely low when the GNP number is small.

TF-IDF Efficiency: It is well-known that the distribution of textual words in documents follows the Zipf's Law [17], in which the most frequent words comprise dominant portions of word occurrences. On the contrary, the visual word distribution did not follow Zipf's Law [17] (Fig.9) because of the inappropriate hierarchical generation. Furthermore, as the hierarchy level increases, the word distribution becomes more and more uniform (Fig.9). Consequently, we can explain the phenomenon observed in an earlier paper [12] that IDF is less effective for large word volumes. Moreover, we can also infer that, in the current suboptimal visual vocabulary, the "stop words" removal technique won't be very helpful: In many cases, there are no very frequent visual words in images like "a" or "the" in documents. Our inference has been validated by the "stop words" removal experiments in [12].

Table 1: Hierarchical Quantization Error Test

NN\GNP	1	3	5	10	15	20
50	41.46%	73.34%	85.00%	94.53%	97.11%	98.18%
200	57.46%	66.21%	79.00%	92.02%	95.00%	97.48%
1000	11.54%	38.27%	51.57%	67.48%	85.16%	94.91%
2000	6.38%	25.68%	40.59%	58.54%	79.21%	92.42%

2.2. Generality of Visual Vocabulary

Facing a new database for retrieval, a common strategy is to recluster the entire local feature dataset and regenerate all BoW vectors for new images. However, in large-scale databases, the computational cost of hierarchical reclustering is high (plan-level reclustering is even higher). It is a natural shortcut to reuse an existing visual vocabulary to the new database. However, it is very hard to construct a generalized visual codebook to be suitable for all scenarios, especially when the new database has a different data volume (e.g. 1K images vs. 1M images) and different image properties (e.g. natural scene images vs. city street images). If we directly reindex a new database with an old model, such differences will cause imbalance and mismatches in the middle level, resulting to the over-fitting of the original visual vocabulary. To address this problem, in Section 4 we present a VT adaption algorithm, which transfers a hierarchically optimized vocabulary model to fit to new data distributions.

3. Optimization of Visual Vocabulary Hierarchy

In this section, we present our **D**ensity-based **M**etric **L**earning (**DML**) algorithm to unsupervisedly optimize vocabulary tree generation. We achieve better retrieval precision (Section 5) using the DML-based tree, because: (1) Its "visual words" distribution follows Zipf's Law [17] more closely; and (2) It reduces the overall quantization error, the meaningful words (sparse clusters) are quantized more finely while the meaningless words (denser clusters) are quantized coarsely. Moreover, we present an "If...Else..." hierarchical rejection strategy to deal with hierarchical quantization error in the VT-based retrieval. It combines the idea of the boosting chain classifier [14] with the nature of the tree hierarchy in an unsupervised manner to improve retrieval.

3.1. DML to Optimize Vocabulary Hierarchy

The Main Idea: As discussed in Section 2, the vocabulary hierarchy iteratively magnifies the asymmetric feature space quantization of k-means clustering. It results to suboptimal visual words and suboptimal IDF in earlier papers [11][12]. DML addresses this issue by constructing a Density Field, which evaluates the distribution tightness of local features, to guide quantization procedure in local feature space. By extending denser clusters and shrinking sparser clusters, the similarity metric in k-means is modified to correct the biased hierarchical quantization. Subsequently, a refined hierarchical division is achieved by

DML-based hierarchical k-means, which offers more "meaningful" visual words:

The Algorithm: First, the Density Field in the SIFT feature space is estimated using the density of each SIFT point as a discrete approximation. The density of a SIFT point is defined as kernel density estimation in Eq.1:

$$\mathrm{D}(i) = \frac{1}{n}\sum_{j=1}^{n} e^{-||x_i - x_j||_{L2}} \quad (1)$$

where $D(i)$ is the point-density of the i^{th} SIFT point, n is the total number of SIFT points in this dataset, and x_j is the j^{th} SIFT point. We adopt $L2$ distance to evaluate the distance between two SIFT points.

To decrease the computational cost, we approximate the density of each SIFT point using only its local neighbors as:

$$\widetilde{D}(i,m) = \frac{1}{m}\sum_{k=1}^{m} e^{-||x_i - x_j||_{L2}} \quad (2)$$

where $\widetilde{\mathrm{D}}(x,m)$ is the point-density of the i^{th} SIFT feature in its m-neighborhood. We only need to calculate local neighbors of SIFT by estimating the point density by the neighborhood approximation: (1) cluster database into k clusters: $O(k\times h\times l)$, with h iterations on l points, and (2) nearest neighbor search in each cluster: $O(l/k)$. By choosing a large k (e.g. 2000), our DML would be very efficient. By using a heap structure, it can be further accelerated. Then, the similarity metric in hierarchical k-means is refined by density-based adaption:

$$DisSim(C,i) = AveDen(C) \times Dis(C_{center}, i) \quad (3)$$

Similarity$(c,i$ is the similarity in DML-based k-means between the c^{th} cluster and the i^{th} point; $AveDen(c)$ is the average density of SIFT points in the c^{th} cluster, and $Dis(C_{center},i)$ is the distance between the center SIFT point in c and the i^{th} SIFT point.

The Explanation: From the viewpoint of information theory, the generation of the visual words will cause information loss in the hierarchical quantization process. Our proposed DML method decreases the overall quantization errors by shortening the quantization steps for the sparser regions, therefore represents the "meaningful" points more precisely (shown in Fig.1). Our method can be also analogized to asymmetry quantization in signal processing. To see how the DML-based method reduces the overall quantization error, we view the information loss in quantization using its weighted quantization errors, in which the weight means the informative degree of the quantized word, evaluated by its IDF value. We denote the quantization error of the word i as $QA(i)$ ($QA(i)$>0):

$$QA(i) \stackrel{\text{def}}{=} \sum_{j=1}^{m_i}\sum_{k=1}^{128}(F_{ij}^{k\,2} - \tilde{F}_i^{k\,2}) \quad (4)$$

$F_{ji}^{\ k}$ is the k^{th} feature dimension in the j^{th} SIFT point of the i^{th} cluster; $\tilde{F}_i^k$ is the k^{th} feature dimension of the i^{th} cluster center; m_i is the number of SIFT in cluster i; w_i is the IDF weight of the i^{th} cluster; n is the number of visual words.

To evaluate the weighted quantization errors between the DML method and the original method, we compare their weighted signal-noise ratios as follows:

$$\frac{SNR_{DML}}{SNR_{Org}} = \frac{P_{Signal}^{DML}}{P_{Noise}^{DML}} \Big/ \frac{P_{Signal}^{Org}}{P_{Noise}^{Org}} = \quad (5)$$

$$\frac{\int x^2 p_x(x)dx}{\int\left[\Delta_{n_{DML}}(x)\right]^2 p_x(x)dx} \Big/ \frac{\int x^2 p_x(x)dx}{\int\left[\Delta_{n_{Org}}(x)\right]^2 p_x(x)dx} = \frac{\int\left[\Delta_{n_{Org}}(x)\right]^2 p_x(x)dx}{\int\left[\Delta_{n_{DML}}(x)\right]^2 p_x(x)dx}$$

w_i is the IDF of the i^{th} cluster as its word weight; $p_x(x)$ and $\Delta_n(x)$ are the sampling probability and squared input-output difference of x respectively. In the discrete case, Eq.5 can be replaced by Eq.6, in which we denote the DML case by superscript ' (such as n', j' and $w_{i'}$):

$$\frac{SNR_{DML}}{SNR_{Org}} = \frac{\sum_{i=1}^{n}(w_i\times\sum_{j=1}^{m_i}\sum_{k=1}^{128}(F_{ij}^k - \tilde{F}_i^k)^2)}{\sum_{i'=1}^{n'}(w_{i'}\times\sum_{j'=1}^{m_{i'}}\sum_{k=1}^{128}(F_{i'j'}^k - \tilde{F}_{i'}^k)^2)} \quad (6)$$

Constrained by: $\sum_{i=1}^{n} w_i = \sum_{i'=1}^{n'} w_{i'}$ (7)

The SIFT space is quantized into equal-numbered words, hence n=n'. Putting Eq.4 into Eq.6, we derive:

$$\frac{SNR_{DML}}{SNR_{Org}} = \frac{\sum_{i=1}^{n}(w_i\times\sum_{j=1}^{m_i}\sum_{k=1}^{128}(F_{ij}^{k\,2} - \tilde{F}_i^{k\,2}))}{\sum_{i'=1}^{n'}(w_{i'}\times\sum_{j'=1}^{m_{i'}}\sum_{k=1}^{128}(F_{i'j'}^{k\,2} - \tilde{F}_{i'}^{k\,2}))} = \frac{\sum_{i=1}^{n}(w_i\times QA(i))}{\sum_{i'=1}^{n'}(w_{i'}\times QA(i'))} \quad (8)$$

$QA(i)$ is the quantization error in cluster i with m_i points:

$$QA(i) = \sum_{k=1}^{128}\left(\sum_{j=1}^{m_i} F_{ij}^{k\,2} - m_i\tilde{F}_i^{k\,2}\right) = \sum_{j=1}^{m_i}\sum_{k=1}^{128}(F_{ij}^k - \tilde{F}_i^k)^2 \quad (9)$$

Since a denser region appears in a larger portion of images and hence would be assigned a very low IDF (shown in Fig.1), it can be neglected in Eq.9, because its IDF is close to zero from the logarithmic nature of the IDF calculations. On the contrary, in the sparser regions, the quantization is much finer than in the original method. Such a region has a much larger IDF and contributes more to the weighted quantization error.

Based on Eq.9, $QA(i)$ depends on: (1) Point count m_i; and (2) Distance between each point j and its word center. By DML construction, in sparser regions both (1) and (2) are smaller than in the original k-means, which lead to a smaller quantization error $QA(i)$. In other words, our method quantizes the "meaningful" regions with finer steps, and quantizes the "meaningless" regions with coarser steps. Based on the DML, we multiply a larger w with a smaller QA in Eq.8, while the larger QA indicates almost zero IDF.

Evaluating the overall quantization error, it is straightforward that $\sum_{i=1}^{n}(w_i \times QA(i)) \geq \sum_{i'=1}^{n'}(w_{i'} \times QA(i'))$ and $SNR_{DML} \geq SNR_{Org}$. In other words, regardless of the quantization errors in meaningless words such as "a" and "the" (such words can be disregarded in similarity ranking as "stop words"), there are smaller weighted quantization errors in the DML-based vocabulary model.

3.2. Employing Hierarchy in Retrieval

With an optimized vocabulary hierarchy in hand, we further investigate how to best use it in improving retrieval performance. Using internal levels in vocabulary hierarchy will increase the computational cost associated with

querying the inverted file (using either k-means, VT [3] or AKM [10]). As pointed out by Nister et al. [3], using internal nodes greatly increases the computational cost when querying the inverted file. This is due to the fact that the computational cost associated with this operation strongly depends on the rate of non-zeros elements in BOW vectors. To address this issue, we present a unsupervised hierarchical rejection strategy to further accelerate hierarchy-integrated retrieval, which achieves better performances comparing with the hierarchical combination of [3] and GNP[5] (Section 5).

First, by expanding the TF-IDF term weighting procedure to the hierarchical levels in the DML-based tree, middle-level nodes are introduced into the similarity ranking. In [3], at each hierarchical level, the IDF of the middle node is calculated and recorded for middle-level matching. If SIFT points are matched at a deeper level, they would be also matched at the higher hierarchy. Compared with GNP [5], it would improve efficient computational efficiency, while maintaining comparable performance (Fig.11 and Fig.12) after DML. Second, we have discovered that the integration of visual words in the vocabulary hierarchy is a natural analogue to boosting chain classification [14][18], which brings us a totally new insight about how to exploit the vocabulary hierarchy. We resort to unsupervised ranking and rejection level-by-level.

In a hierarchical tree structure, considering each level as a candidate selector, generally speaking, the upper levels have higher recall, while the lower levels have higher precision. This is straightforward because the higher level represents an abstraction of category information and is more uniform, with coarser similarity comparison. On the contrary, in the lowest level, the features are very limited and specific, leading to the highest precision.

As presented in Fig.2, each level is treated as a candidate image pool, which stores the qualified candidate images that have passed the upper level voting. Within this candidate image pool, we calculate their similarity to the query image based on the Bag-of-Visual-Words vector generated in the vocabulary of this level. We rank these similarities and reject the least similar candidate images from the candidate pool, then pass the surviving images to

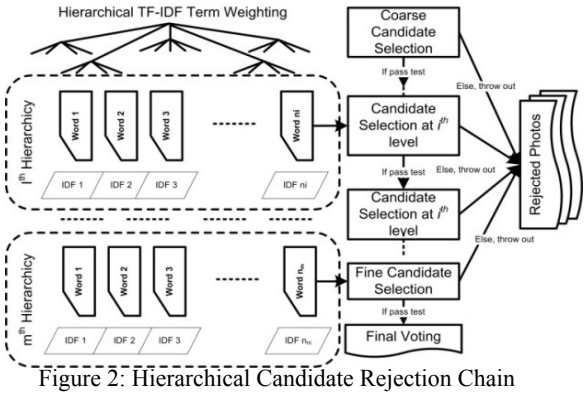

Figure 2: Hierarchical Candidate Rejection Chain

Algorithm 1: Hierarchical Candidate Rejection Chain

Input: Query Image with m SIFT features, DML-constructed vocabulary tree with L levels, candidate image set I: i_1-i_n selected from a coarse level, ascending rejection threshold K: k_1-k_l

For vocabulary hierarchy level i **from** 1 **to** L **{**

If…Else… Test: **{**

Go through BoW vocabulary of this level using query features; **get** its middle-level BoW query vector; **compare** it **with** *BoW* vectors of candidate images at the same level *I'*.

Discard top k_i dissimilar images in I, **update** *I' as I***}**

Stop Test:**{**

If the number of candidate images in *I'* is less than 50

Or this is the lowest level in the vocabulary hierarchy, **rank** the remaining images in *I'* as the final ranking results**}}**

Output: The final ranked list of remaining candidate images.

the next-level candidate pool. This vertically-ordered candidate rejection chain is initialized based on the coarsest level candidate image selection (using a coarse BoW vector representation to generate the 200 most similar candidate images), and ended based on the finest level candidate image selection (the finest BoW for output final ranking).

4. Transfer the Optimized Vocabulary Hierarchy

Facing a new database, the traditional solution is to rebuild the visual vocabulary and search model, which is time-consuming in large-scale applications. An alternative is to use the search model to inversed index new images, without updating the search model and visual vocabulary at all. It will cause accumulated errors in the long term. Yeh et al. [24] proposed a VT adaption for database incremental indexing, in which new data points are directly sent into the VT model to grow and adjust tree nodes. However, [24] did not consider how to transfer the vocabulary model to a totally different dataset. In our research, we discovered that the vocabulary hierarchy can help us to transfer a model between different datasets. Furthermore, the optimized vocabulary hierarchy produced by DML-based learning can achieve much better results than [24]. We present a novel tree adaptation algorithm: **VT Shift**, which enables adaptation of a vocabulary hierarchy among databases.

First, SIFT features of new database are sent through the original vocabulary hierarchy, and the term weightings of words are updated. The feature frequency of each leaf node reveals its rationale for existence and the necessity of further expansion or removal. Leaf nodes that are either over-weighted or over-lightened would be adaptively reassigned based on the following three operations:

1. **Leaf Delete**: If the feature frequency of a leaf is lower than a minimum threshold, its features are reassigned back to its parent. Subsequently, this parent uses its sub-tree (with this parent as root node) to assign the above features to their nearest leaves (other than the deleted leaf).

2. **Parent Withdraw**: If a leaf is the only child of its Parent, and its frequency is lower enough, we withdraw this

Algorithm 2: Vocabulary hierarchy adaptation based on VT Shift

Input: SIFT set of new dataset, empty Operation Queue (*OQ*).
For each feature in the new dataset{
  Recalculate term weightings of words in the new dataset
  Increase the hierarchical feature frequency *Fre* of each node, that is in the indexing path of current feature}
Go through each leaf node n_i of the vocabulary hierarchy{
  If $Fre_i \leq £_{min}$ or $Fre_i \geq £_{max}$, **push** n_i into the *OQ*.}
While the *OQ* is not empty{
  Get the first element n_j
  If $Fre_i \leq £_{min}$ {
    If there are siblings $n_j^{sibling}$ of n_j {*Leaf Delete*, **push** all $n_j^{sibling}$ into *OQ*.}
    Else {*Parent Withdraw*, **push** n_j's parent into *OQ*.}}
  If $Fre_i \geq £_{max}$ {
    Leaf Split, **push** new leaves into *Operation Queue*.}}
  Delete n_j.}
Output: Refined vocabulary tree after adaption.

leaf and downgrade its parent to a new leaf.

3. **Leaf Split**: If the feature frequency of a leaf is higher than the maximum threshold, we re-grow this node as a sub-tree with the same branch factor in construction.

5. Experimental Results

System Framework: Our system consists of both an offline part and an online part. In the offline part, SIFT features [1] are extracted from the image dataset as local descriptors to build the vocabulary tree. A document list is built for each word to record which scene contains the word, thus forming an inverted index file for words. In the online part, SIFT are extracted from the query image and mapped to a *BoW* vector, based on which the relevance score for every document is calculated for ranking.

Experimental Dataset: In our experiments, two datasets were investigated: *Scity* and *UKBench*. *Scity* consists of 24,500 street-side photos, captured automatically along Seattle streets by a car, as shown in Figure 4. We resized these photos to 400×300 pixels and extracted 300 features from each photo on average. Every six successive photos were defined as a scene. *UKBench* dataset contains 10,000 images of CD covers and indoor objects.

In both datasets, each category was divided into both a query set (to test performance) and a training set (to create VT): In the *Scity* dataset, the last image of each scene was

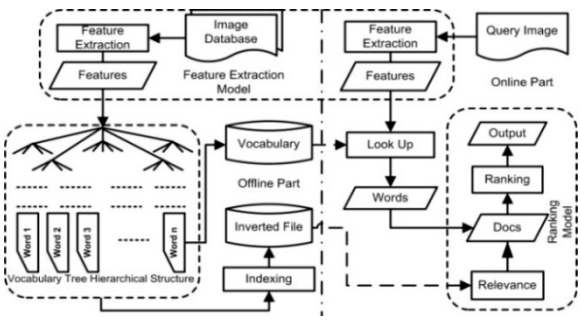

Figure 3: System Architecture of VT-based Recognition.

Figure 4: Examples from Scity Dataset

utilized for the query test, the first 5 was used to construct the ground truth set. In the *UKBench* dataset, in each category, the first 1 image was added to the query set, the rest were used to construct the ground truth set.

We used the Success Rate at *N* (*SR@N*) to evaluate performance. This evaluation measure is commonly used in evaluating Question Answering (QA) systems. *SR@N* represents the probability of finding a correct answer within the top *N* results. Given *n* queries, *SR@N* is:

$$SR@N = \frac{\sum_{q=1}^{n} \theta(N - pos(a_q))}{n} \quad (10)$$

a_q is the answer of query q, $p(a_q)$ is its position, $\theta()$ is a Heaviside function: $\theta(x)$ =1, if $x \geq 0$, otherwise $\theta(x)$=0. When n=4, SR@N is the identical criterion in [3].

We build a 2-branch, 12-level VT for both *Scity* and *UKBench*. In both trees, if a node had less than 2,000 features, we stopped its k-mean division, whether it had achieved the deepest level or not. In tree adaptation, the maximum threshold $£_{max}$ was set as 20,000; the minimum threshold $£_{min}$ was set as 500.

Hierarchy Optimization Results: Fig.5 and 6 present the SR@N in *UKBench* before and after DML, for both of which four comparisons were conducted: 1 Traditional leaf comparison, in which only the leaf nodes were used for *BoW*-based ranking; 2 Leaf Comparison + IDF weighting; 3 Hierarchical chain, in which we adopt the hierarchical rejection chain for multi-evidence search; 4. Hierarchical chain + IDF, which combines *BoW* vector at hierarchical level and their IDF as weights in similarity ranking.

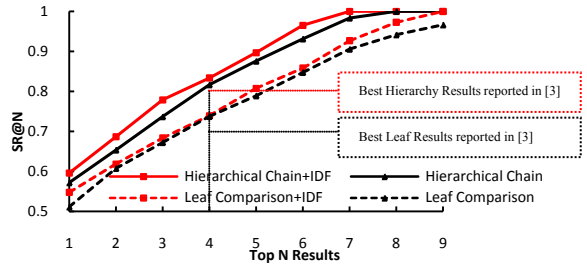

Figure 5: Performance Comparison on *UKBench* Original Tree

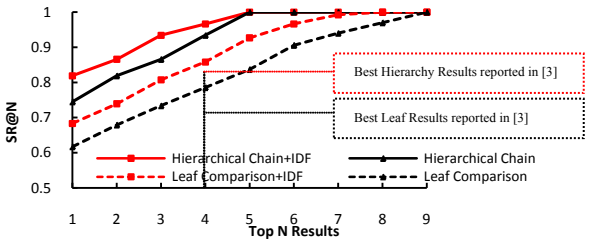

Figure 6: Performance Comparison on *UKBench* DML Tree

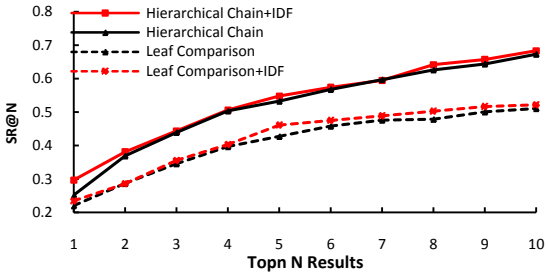

Figure 7: Performance Comparison on *Scity* Original Tree

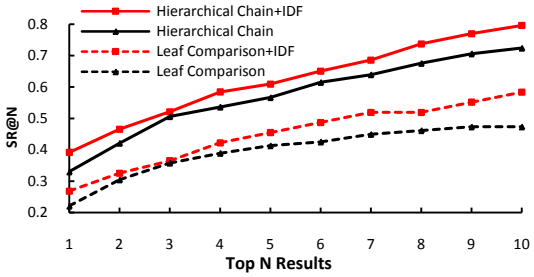

Figure 8: Performance Comparison on *Scity* DML Tree

It is obvious that, before DML learning, the real powers of the hierarchical rejection chain and hierarchical IDF cannot be expressed. But the combined performance enhancements of both methods after DML learning were significant-almost 20% over the leaf-level baseline. Compared Fig.5 with Fig.6, the DML-based VT performs much better than the original VT, with 20% higher over its best results. The same result holds in *Scity* (Fig.7 and 8).

Indeed, this phenomenon can be expressed by Fig.9, after DML-based construction: the word distribution in each hierarchical level follows *Zipf*'s Law better, which means that the vocabulary is more discriminative and

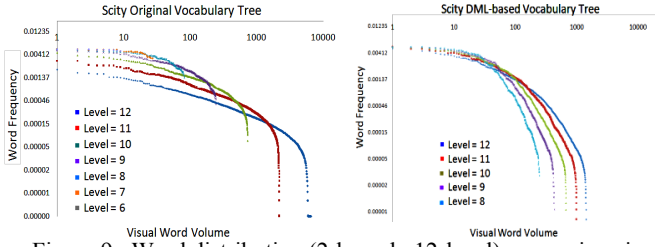

Figure 9: Word distribution (2-branch, 12-level) comparison in *Scity*. Words are ranked by frequency. (Log-Log Plot)

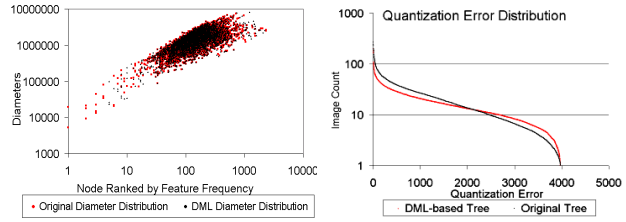

Figure 10: (Left) Cluster diameter (maximum L2 distance within it) distribution in the 12^{th} level ranked by feature frequency. After DML, the diameter distribution is more uniform, hence the feature space division is more uniform; (Right) weighted quantization error distribution ranked by image counts of cluster. The weighted quantization errors with more images are lower, and vice versa.

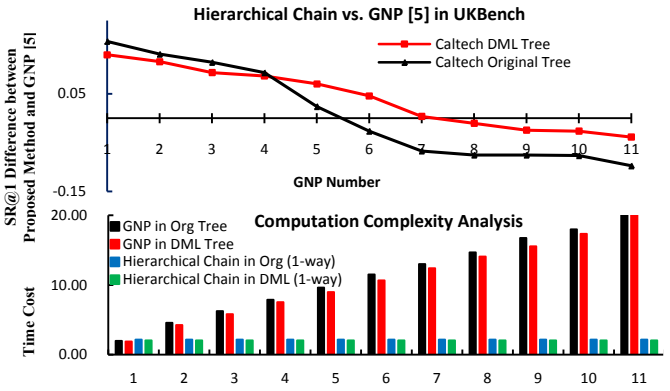

Figure 11-12: Performance Comparison between Hierarchical Recognition Chain (1-way) and GNP. (GNP number: 1-11)

suitable for using IDF (The textual word distribution has been proven to suitable for using TF/IDF in ranking).

Actually, without DML-based optimization, the hierarchical k-means process would be biased to the denser regions, incorrectly dividing them deeper and tighter rather than sparse regions. This bias is hierarchically accumulated and wrongly assigns denser regions high IDF in recognition. This is why the use of IDF is not better in the original VT [6] [12] but is better in the DML-based VT. Fig.10 further explains this enhancement in *UKBench*. An intuitive motivation for *DML*-based tree construction is to refine the original distance metric in k-mean clustering, which is achieved by generating more clusters in dense regions and fewer clusters in sparse regions. Since the VT hierarchy magnifies unbalanced space division, revising such bias by the DML correction would be beneficial.

We further investigate the efficiency of the hierarchical recognition chain. Fig.11 and 12 compare our method with GNP [5]. We also evaluate how the chain level affects the recognition performance in *UKBench*. The method in Nister et al. [3] that combines middle-levels into a unified *BoW* vector were compared (Fig.13). Our method achieves better results than the hierarchy combination [3]. It is worth mentioning that our DML strategy could be also adopted in refining AKM [10] search precision, which would also improve performance by optimize the cluster process within each hierarchical level construction.

VT Shift Results: We evaluated our proposed VT Shift

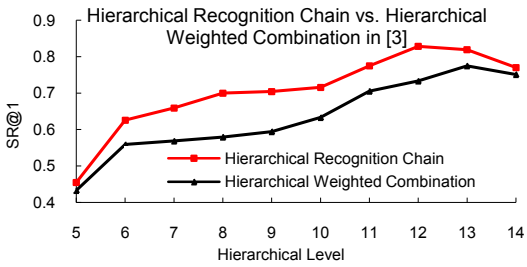

Figure 13: Performance of hierarchical chain at different hierarchy levels. Comparing with [3], we achieve better performance. When n>14, over-quantization is observed.

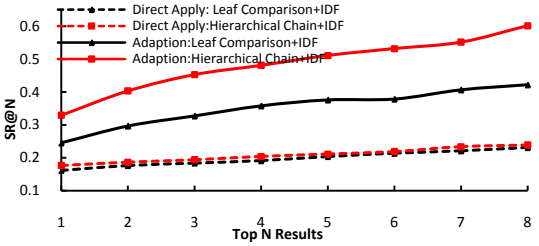

Figure 14: VT Shift performance from *Scity* to *UKBench*

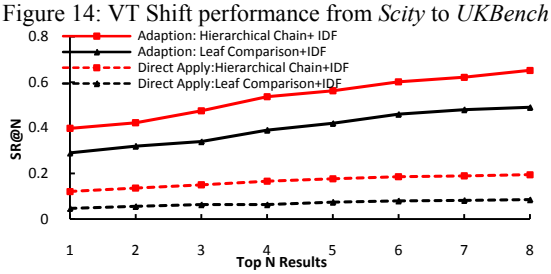

Figure 15: VT Shift performance from *UKBench* to *Scity*

algorithm by the following experiments: Fig.14 and 15 present the VT Shift performance between *UKBench* and *Scity* with different volumes (9K vs. 20K) and applications (scene vs. object). It is obvious that direct application of a vocabulary across datasets of very different constitution and purpose would cause significant performance degradation. However, the *VT Shift* algorithm can well address the tree adaptation problem. Another discovery (Tab.2) is that *VT Shift* in a DML tree performs much better than in the original tree. This demonstrates an advantage of the DML construction in model reapplication.

As in Fig.14 and 15, while straightforward VT reapplication across different datasets gives the expected performance degradation, the performance of VT Shift is much better, close to the vocabulary regeneration. The computational cost of VT Shift depends on the diversity of the two datasets. In general, it is much faster than re-build a new vocabulary, since the IDF updating is much faster than reclustering. In an extreme case, if features of a new dataset were crowded into a single leaf, VT Shift will recluster the entire database. In other cases, the number of features to be adapted is limited, and they are distributed between different leaves. Hence the overall complexity is limited.

6. Conclusion

Our main contribution is the hierarchy optimization of visual vocabulary to improve and transfer the retrieval model. We present a density-based metric learning (DML) algorithm for unsupervised optimization of vocabulary hierarchy construction, based on which we introduce a hierarchical rejection chain to achieve efficient online search. Compared with state-of-the-arts [3][5][10], our performance enhancement is 6-10% in *UKBench* database and 10-20% in *Scity* database. In addition, transferring optimized vocabulary model between different databases provides new inspirations in using former retrieval models.

Table 2: Performance Analysis of *VT Shift*

Reclustering (*UKBench*) vs. *VT Shift* (*Scity-UKBench*)						
Tree/SR@N	1	2	3	4	5	Time Cost
Org	0.196	0.287	0.379	0.433	0.497	8394.9s
DML	0.419	0.466	0.534	0.611	0.669	8279.5s
Shift Org	0.194	0.257	0.330	0.376	0.438	1937.4s
Shift DML	0.348	0.422	0.475	0.537	0.563	1988.2s
Reclustering (*Scity*) vs. *VT Shift* (*UKBench-Scity*)						
Tree/SR@N	1	2	3	4	5	Time Cost
Org	0.152	0.269	0.339	0.404	0.433	6034.2s
DML	0.292	0.367	0.421	0.484	0.509	6129.4s
Shift Org	0.095	0.173	0.227	0.289	0.311	1434.7s
Shift DML	0.206	0.329	0.403	0.453	0.481	1125.3s

7. References

[1] D. Lowe: Distinctive image features form scale-invariant keypoints. *IJCV*. 20(2), 91--110, 2004

[2] J. Matas, O. Chum, M. Urban and T. Pajla: Robust wide baseline stereo from maximally stable extremal regions. *BMVC*, 2002

[3] D. Nister, H. Stewenius: Scalable recognition with a vocabulary tree. *CVPR*. 2006

[4] F.-F. Li and P. Pietro: A Bayesian hierarchical model for learning matural scene categories. *ICCV*. 2007

[5] G. Schindler, M. Brown: City-scale location recognition. *CVPR*. 2007

[6] J. Sivic and A. Zisserman: Video Google: a text retrieval approach to object matching in videos. *ICCV*. 2003

[7] G. Salton and C. Buckley: Term-weighting approaches in automatic text retrieval. *Information Processing and Management*. 24(5), 513—523. 1988

[8] T. Hofmann: Unsupervised learning by probabilistic latent semantic analysis. *ML*. 41, 177—196. 2001

[9] D. Blei, A. YNg, and M. Jordan: Latent Dirichlet allocation. *Journal of MLR*. 3, 993—1022. 2003

[10] J. Philbin, O. Chum, M. Isard, J. Sivic and A. Zisserman: Object retrieval with large voc. and fast spatial matching. *CVPR*. 2007

[11] L. Wang: Toward a discriminative codebook: codeword selection across multi-resolution. *CVPR*. 2007

[12] J. Yang: Evaluating bag-of-visual-words representations in scene classification. *ACM Multimedia*. 2007

[13] F. Jurie and B. Triggs: Creating efficient codebooks for visual recognition. *IJCV*. 604—610. 2005

[14] P. Viola, M. Jones: Rapid object detection using a boosted cascade of simple features. *CVPR*. 2001

[15] L. Kennedy, M. Naaman, and S. Ahern: How flickr helps us make sense of the world: context and content in community contributed media collections. *ACM Multimedia*. 2007

[16] D. MacQueen: Information theory, inference, and learning algorithms. Cambridge P. 2003

[17] R. Yates, B. Neto: Modern information retrieval. *ACM P*. 1999

[18] M. Dundar, J. Bi. et. al.: Joint optimization of cascaded classifiers for computer aided detection. *CVPR*. 2007

[19] K. Mikolajczyk, B. Leibe, and B. Schiele: Local features for object class recognition. ICCV. 1792—1799. 2005

[20] T. Leung and J. Malik: Representing and recognizing the visual appearance of materials using 3-d textons. *IJCV*. 2001

[21] H. Jegou, H. Harzallah, and C. Schmid: A contextual dissimilarity measure for accurate and efficient image search. *CVPR*. 2007

[22] J Philbin, O. Chum, M Isard, J Sivic, and A Zisserman: Lost in quantization: improving particular object retrieval in large scale image databases. CVPR. 2008

[23] K Grauman and T Darrell: Approximate Correspondences in High Dimensions. *NIPS*. 2007

[24] T. Yeh, J. Lee, and T. Darell: Adaptive Vocabulary Forest for Dynamic Indexing and Category Learning. *CVPR*. 2007