Partial-Duplicate Clustering and Visual Pattern Discovery on Web Scale Image Database

Wei Li, Changhu Wang, *Senior Member, IEEE*, Lei Zhang, *Senior Member, IEEE*, Yong Rui, *Fellow, IEEE*, and Bo Zhang

***Abstract*—In this paper, we study the problem of discovering visual patterns and partial-duplicate images, which is fundamental to visual concept representation and image parsing, but very challenging when the database is extremely large, such as billions of images indexed by a commercial search engine. Although extensive research with sophisticated algorithms has been conducted for either partial-duplicate clustering or visual pattern discovery, most of them can not be easily extended to this scale, since both are clustering problems in nature and require pairwise comparisons. To tackle this computational challenge, we introduce a novel and highly parallelizable framework to discover partial-duplicate images and visual patterns in a unified way in distributed computing systems. We emphasize the nested property of local features, and propose the generalized nested feature (GNF) as a mid-level representation for regions and local patterns. Initial coarse clusters are then discovered by GNFs, upon which n-gram GNF is defined to represent co-occurrent visual patterns. After that, efficient merging and refining algorithms are used to get the partial-duplicate clusters, and logical combinations of probabilistic GNF models are leveraged to represent the visual patterns of partially duplicate images. Extensive experiments show the parallelizable property and effectiveness of the algorithms on both partial-duplicate clustering and visual pattern discovery. With 2000 machines, it costs about eight and 400 minutes to process one million and 40 million images respectively, which is quite efficient compared to previous methods.**

***Index Terms*—Local features, parallel algorithms, partial-duplicate images, visual patterns.**

Manuscript received August 27, 2014; revised January 05, 2015 and April 13, 2015; accepted April 26, 2015. Date of publication May 01, 2015; date of current version June 13, 2015. The work of W. Li and B. Zhang was supported in part by the National Basic Research Program of China (973 Program) under Grant 2013CB329403 and Grant 2012CB316301, in part by the National Natural Science Foundation of China under Grant 91120011 and Grant 61273023, and in part by the Tsinghua University Initiative Scientific Research Program 20121088071. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Lexing Xie. *(Corresponding author: Changhu Wang.)*

W. Li and B. Zhang are with the State Key Laboratory of Intelligent Technology and Systems, Tsinghua National Laboratory for Information Science and Technology, Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China (e-mail: liwei2658@gmail.com; dcszb@mail.tsinghua.edu.cn).

C. Wang and Y. Rui are with Microsoft Research, Beijing 100080, China (e-mail: chw@microsoft.com; yongrui@microsoft.com).

L. Zhang is with Microsoft Research, Redmond, WA 98052 USA (e-mail: leizhang@microsoft.com).

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.2015.2428996

I. Introduction

VISUAL patterns are the basic re-occurring visual information in the visual world, the discovery of which plays an important role in computer vision research and applications. Visual patterns can be defined in different ways according to the semantic level and appearance variance. For example, trees can be regarded as one pattern and the sky as another. These patterns have clear semantic meanings, but with significant intra-class variance. On the other hand, we can focus on a lower semantic level with less appearance variance, like different logos or landmark patterns. Although the former class-level definition is very useful for applications like image parsing and image categorization, users normally prefer to search more specific patterns in real-life applications. For example, in the augmented reality scenario, it is more useful to tell a user that he is looking at a loquat tree rather than only a tree, or the Kinkakuji Temple instead of a temple. Thus, in this work we mainly focus on instance-level visual pattern with limited appearance variance, which plays an important role in specific object recognition.

Partial-duplicate images always share this kind of visual patterns. With the explosive growth of web images and convenience of image acquisition and sharing, more than 28% web images have partial-duplicate images [1]. They mainly come from manipulations on images by cropping and editing. Partial-duplicate clustering in a web-scale database plays an important role in many real-world applications, e.g., image annotation by mining surrounding texts of partial-duplicate images [2].

Many works have been devoted to finding partial-duplicate clusters in a collection of images [1], [3]–[6]. Most of them are based on local features, either being tested only on small datasets [5] or evaluated on small datasets and tested on million-level datasets without detailed evaluations [3], [6]. A few papers leverage global features, but they do so for image retrieval rather than clustering [4], or for partitioning huge image sets into smaller disjoint ones [1]. Although methods like min-hash show promising results [3], [5] with local features, when the database scales up they suffer from scalability issues [1] and have high false positive rates [6]. [1] is one of the few works that address partial-duplicate clustering on a very large image database, i.e., 2 billion images. It adopts global features to partition image space and cluster images, followed by cluster merging via matching local features. It has good scalability but low recall, because the global feature-based partition only keeps globally similar images rather than partial-duplicates with small local region overlaps. We argue that these local regions are of great importance for parsing images with finer granularity and discovering visual patterns. This motivates us to develop an algo-

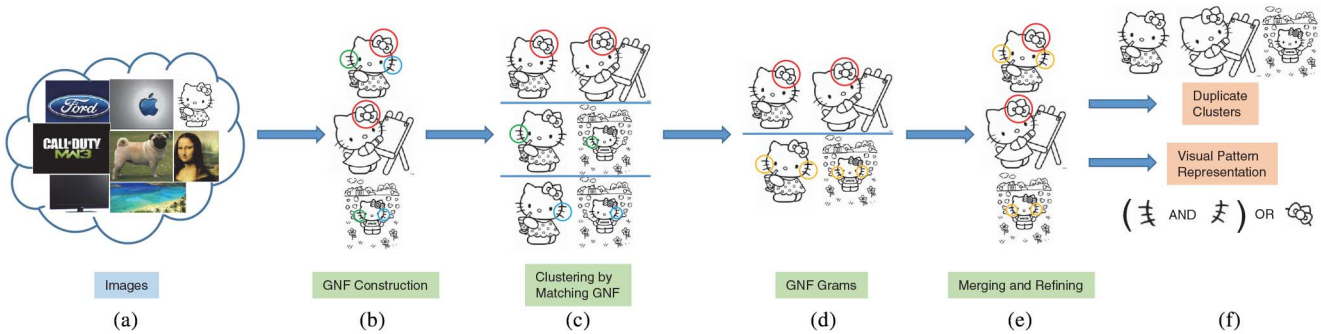


Fig. 1. Overview of the proposed framework. Each circle represents a GNF. It illustrates the detection of one cluster and its visual patterns. From input images (a) we first extract GNF groups as a mid-level representation for regions and local patterns (b). Initial clusters can be discovered efficiently by GNF matching (c). n-gram GNFs are further defined for a more discriminative representation for visual patterns (d). Finally, efficient merging and refining algorithms (e) are used to get partial-duplicate clusters as well as visual patterns modeled by logical (AND/OR) combinations of GNFs (f). For more details please refer to Section III.

rithm that can discover partial-duplicates even with very small overlaps.

Some other works focus on discovering or representing visual patterns without explicit partial-duplicate clustering [7]–[11]. Most of them are devoted to finding a higher level representation of objects and patterns than local features like SIFT [12], e.g., visual phrase [7] or similar representations [8]–[11] that combine several local features together. But the computational cost of such kind of work is sometimes high and thus not easy to scale up [7]. Moreover, sometimes image category information is required prior to pattern discovery [10], [11].

Although partial-duplicate clustering and visual pattern discovery were usually studied separately in the literature, the two problems are actually coupled with each other. Images in the same partial-duplicate cluster normally share the same visual patterns, and visual patterns are the reason for cluster generation. This makes the problems more challenging, especially in a super large database, e.g., billions of images indexed by a commercial search engine, where some sophisticated algorithms need to be avoided for efficiency and parallelization.

In this paper we propose a novel and highly parallelizable framework to solve the two problems of partial-duplicate clustering and visual pattern discovery in a unified way, as shown in Fig. 1. To the best of our knowledge, this is the first work to address these two problems at the same time. The main idea is to first propose candidate visual patterns from each image and cluster images by matching and filtering proposed patterns, followed by the construction of more discriminative patterns and the discovery of better partial-duplicate clusters. In this framework, we utilize the nested relationship between local features, from which we propose the generalized nested features (GNFs) to represent spatially related local features as a mid-level representation for regions and local patterns [Fig. 1(b)]. Based on this representation, initial clusters can be discovered efficiently by matching GNFs [Fig. 1(c)], and n-gram GNFs are further defined for a more discriminative representation for visual patterns [Fig. 1(d)]. Finally, efficient merging and refining algorithms [Fig. 1(e)] are used to get partial-duplicate clusters, and logical [AND/OR] combinations of GNFs are leveraged to model the visual patterns of partially duplicate images [Fig. 1(f)]. Based on the intermediate n-gram clusters, we can also construct probabilistic GNF models, which are more robust and expressive than single GNF representation, and can facilitate applications like image annotation and image retrieval. The framework and algorithms are implemented on a distributed system with thousands of machines to solve real-world problems. We conduct experiments on a variety of public datasets with scales from hundreds to 40 million images. Experimental results show the effectiveness and high parallelization of the proposed framework on partial-duplicate clustering and visual pattern discovery. With 2000 machines, it costs about 8 minutes and 400 minutes to discover partial-duplicate image clusters and typical visual patterns from 1 million and 40 million images respectively. The efficiency and highly parallelizable property make this framework practical to scale up to the billions of images indexed by search engines.

One potential application of this work is automatic image annotation [2]. The visual patterns discovered from a web-scale image database might cover most representative patterns. Particularly for web images indexed by search engines, there are rich surrounding texts and click information that links semantic queries with clicked images. We can link this semantic information with discovered visual patterns. An unseen image can be matched to some indexed visual patterns, whose tags can be propagated to this unseen image. Because of the local pattern match, it is also possible to tag a local region instead of the whole image. Another application is to discover the manipulation history of web photos [13], based on the locally matched parts of partial-duplicate clusters. We can also build a large knowledge base from discovered visual patterns, facilitating other object-level applications, such as object recognition.

The main contributions of this paper include: a) a novel and highly scalable approach to solve partial-duplicate clustering and visual pattern discovery in a unified framework which has achieved promising results; b) the proposed GNF as an improved method to represent and cluster visual patterns and its modeling for applications such as image annotation; and c) the implementation of this framework on a distributed system, which differs from most previous works that used only a single machine, to fully understand the properties of web-scale image database.

The rest of the paper is organized as follows. In Section II we discuss some related works and differentiate them from ours. The framework and algorithms are presented in Section III. In Section IV, experimental results are presented, followed by conclusions and future work in Section V.

II. Related Work

We briefly introduce some representative approaches related to this work on partial-duplicate detection and visual pattern representation and discovery.

Partial-Duplicate Detection: It is worth noting that partial-duplicate clustering is different from partial-duplicate retrieval, where the latter is a retrieval problem with complexity of $O(N)$, but the former is a clustering problem with complexity of $O(N^2)$ in nature. The retrieval problem has been intensively studied [14] since the first introduction of the bag-of-words model to images and videos in 2003 [15]. In particular, many algorithms have been developed to address the problem of spatial information loss in the bag-of-words model [8], [9], [16]–[19]. Spatial information is leveraged in different ways. [8], [9], [19] constructed feature groups based on some spatial relationships to increase the discriminative power of local features. At the search time, [17] matched geometry-preserving visual phrases with spatial information stored in the index, and [18] integrated spatial correspondence in the similarity measure. Reference [16] re-ranked retrieval results with spatial verification. However, most of these techniques can not be easily adapted to the clustering problem. It is also non-trivial to directly apply the retrieval framework to clustering, which essentially requires the use of every image in the database to search for its partial-duplicate version. When the database size increases to tens of millions or billions, clustering by search is impractical simply because of its high computational cost.

Multiple feature hashing [20] is a recent work on near-duplicate video retrieval. This algorithm learns hash functions that map multiple global and local features to binary codes. Then the similarity between two videos could be computed with the Hamming distance between two representative codes, which had high accuracy and efficiency. But the learning scheme requires a number of matrix manipulations. Some of these matrices have size N_t-by-N_t, where N_t is the number of training images. This restricts the scale of the training set. Furthermore, it is non-trivial to design a robust and fast algorithm for clustering simply based on the binary codes representation.

Among the few papers on partial-duplicate clustering, one representative work was proposed by Chum and Matas [3]. It first used s-tuple min-hash values based on local features to discover cluster seeds. Then retrieval techniques were leveraged to find duplicates by using cluster seeds as queries. Geometric constraints were further applied to local features to improve the performance [5], and faster computation of min-hash signatures could speed up the detection process [21]. The idea was further extended by [6], where min-hash values were extracted from image partitions on grids.

Although some of these min-hash-based approaches can effectively solve partial-duplicate clustering on a million level database, it might suffer from the following issues when scaling up even further. First, as pointed out in [6], on a very large scale dataset false positives caused by min-hash is a serious problem. It may lead to many wrong clusters. Second, the complexity of the seed growing step is quite high. Chum and Matas [3] showed that the computational complexity of seed growing is $O(NL)$, where N is the total number of images in the database and L is the number of cluster seeds. As observed in [1], when N is large, L is close to N and $O(NL) \approx O(N^2)$. In contrast, our framework avoids such issues. We leverage feature groups and appropriate matching thresholds to ensure precision. As can be seen in Section III, the most time-consuming step in our framework has complexity $O(NG)$, where G is the average number of GNFs that share the same bounding feature ID, which is normally orders of magnitude smaller than N and L. Furthermore, this step can be completed fully in parallel, resulting in an actual running speed that is much faster than previous methods.

Another representative solution to duplicate clustering is [1], in which global features were first used to partition image space and cluster images, and then local features were adopted to merge clusters. Although it can be scaled up to a billion level database, it can only merge within similar subspaces measured by global descriptors. Because many partial-duplicate images on the web have large appearance variances, partitioning by global descriptors would filter out many diverse images in the cluster, leading to low recall.

Visual Pattern Representation and Discovery: After the introduction of the visual phrase [7], many variances were studied [8]–[11], [22] to obtain a higher level representation of local visual information. For example, [10] constructed delta visual phrases and clustered them into visual synset. Reference [22] grouped local features with K-means and represented each group with a compact descriptor. [9] is an effective method. It grouped local features into bundling groups based on regions detected by a second local feature detector. We base our approach on nested features [8] for several reasons. First, it is simpler and faster than most of the other methods. It only uses one type of local feature and does not require complex computations or image category information, which is quite suitable for discovering patterns on a very large image database. Second, Nested-SIFT outperformed bundling features [9] and some other features in retrieval experiments [8], which showed its discriminative capability. Third, nested features can be generalized to handle images with very few features, e.g., logos and objects with simple shapes, which are common on the web.

Discovering recurring patterns from a single image was studied in [23]. Its idea of forming spatially related feature groups to represent a pattern inspired our framework. Although its algorithm can be adopted to detect patterns recurring in different images, its high computational complexity makes it impossible to be applied to web scale applications.

Instance-level object discovery is closely related to our work. [24] discovered instance-level objects from scenes of daily living. It used one or more image segments to represent an object, which is different from our local-feature-based representation. Also it focused on mining objects from daily scenes, while we use web images as input. Reference [25] further improved the framework with a large product image database to tackle the problem of sparse observations and viewpoint changes. It showed the power of data-driven methods. As in our work, we also observe that the amount of data plays an important role in visual pattern discovery.

A recent work [26] proposed thread of features for scalable visual instance mining. It was shown to be very robust against all kinds of intra-class variations. However, its precision on large

datasets is very low, making it hard to be applied to real-world applications.

There are many other works on visual pattern discovery and we refer the readers to [27] for a more complete survey.

III. Framework and Algorithms

In this section, we first give a brief overview of the proposed framework, and then introduce detailed algorithms.

A. Overview

We design an effective and scalable framework to discover representative visual patterns and partial-duplicate clusters, as shown in Fig. 1. The key idea is to first from each image detect some candidate visual patterns that have high probabilities to find matches in other images. This essentially avoids the combinatorial explosion in visual feature group construction. The candidate visual patterns are discriminative local feature groups in each image, from which images with similar local patterns can be roughly clustered together in a very efficient way. Then, representative visual patterns can be discovered based on the rough clusters, followed by refining partial-duplicate clusters and combining visual patterns. To guarantee the efficiency and parallelizability of the framework, we try to make these steps as separate as possible, each of which can be formulated in the map-reduce manner and thus fully parallelizable.

First, we extract generalized nested features (GNFs) for each image (Section III-B), followed by the clustering algorithm based on GNF matching (Section III-C). Then, a more discriminative local pattern representation, n-gram GNF, is introduced to improve the precision of image clustering (Section III-D). Finally, clusters are further merged to increase the recall of partial-duplicate clustering (Section III-E), the visual patterns of which can be represented by the logical combinations of GNFs (Section III-F). We also propose a probabilistic method to model visual patterns based on intermediate n-gram clusters (Section III-G).

B. Generalized Nested Feature Extraction

In this section, we introduce the proposed generalized nested feature (GNF), which is a group of spatially related local features and thus more discriminative. GNF not only serves as the basic element of our visual pattern representation, but also plays an important role in partial-duplicate clustering.

We briefly introduce the nested local features proposed by [8], followed by the proposed GNFs.

1) Nested Local Features: In this work, we use the local feature described in [28] rather than SIFT [12] due to its discriminative power and efficiency. We use a pre-trained 1 million codebook to quantize feature descriptors to visual words (all experiments use the same codebook). Each local feature is represented by: a) the (x, y) location in the image, b) the orientation, c) the scale, d) the quantized visual word ID, and e) the strength. The strength of a local feature measures the response level of the feature detector. More details on local feature extraction can be found in [12] and [28].

Combinations of local features with spatial relationship have been proven to be more discriminative. We base our work on Nested-SIFT [8] because of its simple computation process, as well as better performance than other ones such as bundling features [9]. It is observed that, many local features are spatially nested in other features in each image. Grouping the nested features together can greatly increase the discriminative power of local features and generates a higher-level representation for local regions and patterns. It can also leverage geometric relationships between features in the same group. As defined in [8], a nested feature group consists of a bounding feature and several member features that are spatially included in the bounding feature, as illustrated in Fig. 2. The minimum number of member features in a group is set to two to remove trivial groups.

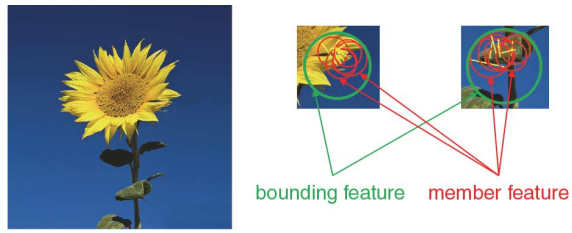


Fig. 2. Examples of nested feature groups in an image. Each circle represents a local feature.

2) Generalized Nested Features (GNFs): We introduce the generalized nested features (GNFs) based on [8] by adding the selection strategy for nested features, and generalizing it from nested features to neighbor features.

Nested Group Selection: One problem of the nested features in [8] is that there is no feature ranking and selection strategy, and thus all the nested groups need to be extracted and used. This might lead to noisy groups of low repeatability, and increase the computational complexity.

To solve this problem, we define the strength of each group and only keep the strongest groups. Group strength can be defined based on the strength values of its bounding feature and member features. In this work, considering the efficiency of group selection, we adopt the strength value of the bounding feature as the strength of the group. We also tested more complex ways to calculate the group strength, such as combination of strength of both bounding and member features, but did not observe obvious improvements. Experiments in Section IV show that group selection can improve the performance. In all experiments, the number of top groups is set to 100, which is a trade-off between eliminating noisy groups and keeping useful information.

Neighbor Feature Group: Another problem of the original nested features is the difficulty in finding spatially nested features when dealing with small images with simple shapes and textures. This is an important issue for our problem, because a large portion of web image thumbnails that we target are these kinds of data, for example logos, simple line drawings, diagram charts, etc.

Thus we generalize the nested relationship to a neighborhood relationship. For a local feature that does not belong to any nested group, we find its 10 nearest neighbors and construct a group with that feature as the bounding feature and its neighbors as member features. Note that for local features which already belong to one or more nested groups, we do not construct any neighbor groups to avoid redundant information and noise.

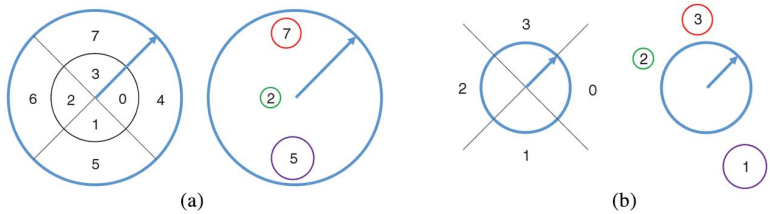


Fig. 3. Illustration of two kinds of GNF. (a) shows a nested feature group. The left figure illustrates how to assign region IDs, in which the arrow represents the orientation of the bounding feature. The right figure shows an example group with region IDs marked in member features. Similarly, (b) illustrates neighbor feature group.

Section IV shows the effectiveness of adding neighbor feature groups. Nested feature groups and neighbor feature groups are constructed in different ways and thus should be considered separately when matching GNFs.

GNF Definition and Representation: The generalized nested features (GNFs) include both nested and neighbor feature groups. The set of GNFs in an image can be formulated as $F = F_{nd} \cup F_{nr}$, where $F_{nd} = \{f_1, f_2, \ldots, f_u\}$ is the set of u nested feature groups and $F_{nr} = \{f_{u+1}, f_{u+2}, \ldots, f_{u+v}\}$ is the set of v neighbor feature groups. Each GNF $f \in F$ is represented by a 5-tuple $< b, M, R, s, t >$, where b is the visual word ID of the bounding feature, $M = \{m_k\}$ is the visual word IDs of member features, $R = \{r_k\}$ is the geometric locations of the member features, s is the strength of GNF, and t is the type of GNF (nested or neighbor). The geometric locations r_k are represented by region IDs relative to the scale and orientation of the bounding feature. For nested feature groups we divide the area of the bounding feature into 8 regions as in Fig. 3(a), numbered from 0 to 7. For neighbor feature groups, as the member features usually appear outside the bounding feature, we only consider the orientation but not the distance, resulting in 4 orientation regions as a weaker geometric constraint, as shown in Fig. 3(b).

C. Clustering via GNF Matching

With each image represented by GNFs, we apply a divide-and-conquer strategy for image clustering for efficiency. We first divide all GNFs to different buckets according to their bounding IDs, i.e., the visual word IDs of their bounding features. Each bucket thus contains GNFs with the same bounding ID and the number of buckets equals to the codebook size. Then, for each bucket, we conduct GNF clustering independently, making it efficient and fully parallelizable. Note that, each GNF corresponds to one image, and thus each image with multiple GNFs might belong to multiple clusters. In later steps we will allow GNFs with different bounding IDs to be merged together on some conditions. Also note that one image might have some identical GNFs due to recurring patterns. We found this is rare in the experiments and does not affect our framework; thus we do not consider it separately.

The codebook size affects the number of buckets as well as the matching criteria during clustering. If the codebook is too small, each bucket will have many images, leading to higher computational cost. On the contrary, if a very large codebook is used, the matching criteria will become too strict to discover various clusters. Therefore in this work, we choose 1 million as the codebook size. Similarly-sized codebooks were widely used in related works.

Next, we will first introduce the GNF similarity measurement and clustering algorithm for each bucket, followed by the analysis of computational complexity.

1) GNF Similarity: To conduct clustering, we first define the similarity between two GNFs. As there are two kinds of GNFs, we consider the following three conditions.

i) *Two nested feature groups*. The similarity is defined as in [8]. Matched member features contribute to the similarity score. As aforementioned, a member feature in a GNF is represented by its visual word ID m and its region ID r. We consider all member feature pairs $< m_i, r_i >$ and $< m_j, r_j >$. If $m_i = m_j$, which means that they have the same visual word ID, there will be an ID match. If both $m_i = m_j$ and $r_i = r_j$ are satisfied, which means that they not only share similar appearances but also are located at similar positions relative to their bounding features, we call it a geometric match. Let S_i be the weight for one ID match, and S_g for one geometric match, the similarity is given by

$$S = 1 + N_i S_i + N_g S_g \tag{1}$$

where N_i and N_g are the numbers of ID matches and geometric matches. Note that if two features are geometrically matched, they are also counted in ID match. In the experiments we empirically set S_i to 1 and S_g to 6.

ii) *Two neighbor feature groups*. As neighbor feature groups have the same representation as nested feature groups, the score computation is also similar. The difference is that geometric match will occur more frequently than nested feature groups, because of the weak geometric constraint. Thus, we reduce the geometric match weight to avoid false positives. The similarity score is

$$S = 1 + N_i S_i + \frac{N_g S_g}{4}. \tag{2}$$

iii) *One nested feature group and one neighbor feature group*. Nested feature groups and neighbor feature groups are constructed in different ways. Thus, it is meaningless to match them. We also observed many false matches in experiments if considering this kind of matching. Therefore we define

$$S = 0. \tag{3}$$

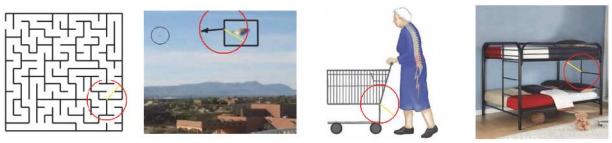


Fig. 4. Unigram GNF with poor discrimination capability. Each circle represents a GNF. In this example, each region contains a T-shape pattern. We can see that although these GNFs belong to one unigram GNF, they actually correspond to different semantics.

2) Clustering: Based on the GNF similarity measurement, we leverage the ϵ-clustering algorithm [1] for GNF clustering. This algorithm connects two points if they are similar enough (controlled by a distance threshold) and outputs the connected components as clusters. An appropriate threshold is important for this algorithm and will be further discussed in Section IV. K-means algorithm is another option. However, it is inappropriate to use K-means, as the number of clusters K is normally close to the number of images given the nature of the partial-duplicate clustering problem where single images (without duplicates) are considered as individual clusters. This makes it difficult to estimate K beforehand. Also K-means is not as efficient as ϵ-clustering because it requires multiple iterations. We thus choose ϵ-clustering in this work.

After ϵ-clustering, we get many coarse clusters. Each cluster corresponds to one type of GNFs, and every GNF in the cluster is linked to one image. The clustered GNFs are repeatable and representative, while less meaningful GNFs such as those on the background are filtered. However, these clusters are generated via simple matching rules within the same bucket. The simple matching rules are not sufficient to achieve a good trade-off between precision and recall. For example, many partial-duplicate images may exist in different buckets. Therefore further merging and refining steps are needed.

3) Complexity Analysis: Clustering is the most time-consuming step in this framework. The algorithm complexity is $O(VG^2)$, where V is the size of codebook used for feature quantization, and G is the average number of GNFs that share the same bounding feature ID. During the construction of GNFs, because of the top group selection, the number of GNFs in one image is less than a certain number and thus can be treated as a constant. It is therefore easy to know that $O(VG) = O(N)$, where N is the number of input images. Therefore, the complexity can be rewritten as $O(NG)$. In practice, when a large codebook is used, G is orders of magnitude smaller than N. Thus theoretically it is much more efficient than previous methods like [3]. Furthermore, since different buckets are independent, it is easy to process multiple buckets in parallel, making it even faster.

D. n-Gram GNFs

Although GNF matching leverages geometric information to ensure spatial consistency, it still fails in some cases. Recall that each cluster corresponds to a set of visually similar GNFs. They can be considered as a quantized GNF called unigram GNF. Fig. 4 shows an example where unigram GNF has poor discrimination capability. Many web images contain texts, grids and simple shapes, and quantization errors are likely to occur in these cases. Thus a single GNF is sometimes not sufficient to represent a meaningful pattern for clustering. Therefore, we propose n-gram GNFs, i.e., n co-occurrent GNFs, to address this issue.

1) n-Gram GNF Construction: The basic idea is that, if a unigram GNF is not discriminative enough, another one will be combined with this one to compose a pair of GNFs, i.e., bigram GNF. In the same way we can get a 3-gram (unigram+bigram) or 4-gram (bigram + bigram) GNF if this bigram GNF is not discriminative, and so on. In this work, we only adopt unigram and bigram GNFs for efficiency.

Let U be the set of unigram GNFs that our framework discovers. For each unigram GNF $u \in U$, we check whether it is ambiguous by measuring the similarities of the images in its cluster. Specifically, top 40 local features (ordered by strength) are extracted from each image, and then we pairwise compare images in the same cluster by top features to see if they match. More than 6 common top features in two images indicate a matched image pair. If the ratio of matched image pairs is above a certain threshold (50%), we judge that u is representative, otherwise it is ambiguous. In this way U is split into two sets U_r and U_a, containing representative and ambiguous unigram GNFs respectively. Then we consider two different GNFs $u_i \in U_a$ and $u_j \in U_a$. Of all possible u_i and u_j, we only select the frequently co-occuring ones (their clusters share common images and no spatial relationships between u_i and u_j are required), and $< u_i, u_j >$ is called a bigram GNF. All such $< u_i, u_j >$ pairs form the bigram set B. The final cluster of a bigram GNF is the intersection of its two unigram clusters, making sure that each image in the cluster have both patterns. $U_r \cup B$ becomes a set of more discriminative patterns.

The n-gram GNF construction process can be easily formulated in the map-reduce manner, and thus is quite scalable. Specifically, in the map stage, we check all possible $< u_i, u_j >$ pairs and filter them by comparing their clusters to see if they frequently co-occur. The selected pairs are outputted to the reduce stage, where bigram representations are constructed and corresponding clusters are merged.

2) Representation: As a result, there are both unigram and bigram GNF clusters, in which each image has a corresponding pattern represented by a unigram or bigram GNF. Bigram GNFs or higher-order n-gram GNFs can be treated as the logical AND of two or more unigram GNFs. Each image can have several n-gram GNFs, and thus distributes in several clusters.

E. Cluster Merging and Refining

1) Merging: As aforementioned, an image might have different representative patterns (i.e., n-gram GNFs), and thus might belong to different clusters, as shown in Fig. 5. Therefore, if two clusters share a high proportion of images, we merge them as a new cluster. The merging threshold controls the balance between precision and recall. A small threshold leads to many possible merging operations and increases the noise. On the other hand, a large threshold barely helps to increase recall. In the experiments this threshold is set to 50% of the size of the smaller cluster empirically. The merged visual pattern is represented by logical OR of two n-gram GNFs. That is, any

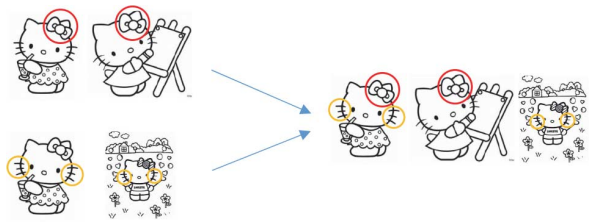


Fig. 5. Illustration of two clusters which can be merged to form a larger cluster. Not all patterns can be detected in an image due to manipulations and quantization errors.

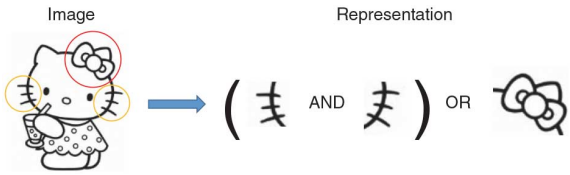


Fig. 6. Example of logical combinations of visual patterns in an image. The image has one unigram GNF and one bigram GNF, and we can represent it by concatenating GNFs with logical operators AND and OR.

one of them is sufficient to identify the semantics of the merged cluster.

For efficient implementation, an inverted index structure is used to avoid pairwise comparisons.

2) Refining: Although the recall of partial-duplicate clustering is significantly improved after merging, the precision of some clusters may drop as merging introduces noises. Therefore, cluster refining is necessary as a post-processing step to guarantee high precision for the final results. In this step, we independently process each cluster; this step is also suitable for parallel computation.

We assign a unique ID to each n-gram GNF (corresponds to a cluster before merging). We call it a super code in this work, as it can be considered as a more discriminative visual word. Each image is then represented by multiple super codes. For each cluster after merging, we remove outlier images by ϵ-clustering, having at least C common super codes as the threshold. In experiments C is checked from 1 to 6 to get the best parameter. This will eliminate a large portion of false positives, improving precision while maintaining the level of recall.

F. Logical Operations on GNFs

The unigram GNFs obtained after GNF clustering are the basic components of the visual pattern representation. The n-gram GNF generation introduces the logical operator AND, while cluster merging brings the OR operator. As a result, each image in partial-duplicate clusters is represented by the logical combinations of unigram GNFs, which indicate the visual pattern model of that image, as shown in Fig. 6.

In this way, each image can be represented by a few super codes, which are the quantized type IDs of its frequently occurring GNFs. This representation is more compact and discriminative than the original local features. Note that the number of super codes in an image is not only far less than the number of local features, but also less than the total number of GNFs, as some GNFs are filtered during clustering.

G. Visual Pattern Modeling

Here we introduce a method to model visual patterns in a probabilistic way, based on which it will be easy to calculate the similarity between an n-gram GNF and a model, facilitating the recognition of unseen images and other potential applications.

1) Super Code Probabilistic Model: As aforementioned, super code means a set of similar n-gram GNFs. We introduce a probabilistic model to represent it. Here we only discuss the modeling of a set of unigram GNFs, which can be directly extended to n-gram GNF models, i.e., the AND combination of n unigram models. In the rest of this section, we use GNF to represent unigram GNF for short.

In our framework, all GNFs corresponding to a super code share the same bounding feature ID, but they may have different member features. It is apparent that if a member feature frequently occurs, it will carry more information and thus is more important in this model. So we define the weight W_a of a member feature a as

$$W_a = \frac{|\{f | f \in F_{SC}, a \in f\}|}{|F_{SC}|} \tag{4}$$

where F_{SC} is the set of GNFs corresponding to the specific super code and $a \in f$ means that a is a member feature of GNF f. If two member features have the same visual word ID but different region IDs, they will be considered as two different member features. In this way, we have the distribution of member features and their locations. Fig. 7 shows an example. The probabilistic model of a super code contains: a) bounding feature ID, b) a list of $< member\ feature\ ID, region\ ID, weight >$ tuples, and c) GNF group type (nested/neighbor). Note that GNFs in Section III-F can be replaced with models to get a better representation.

2) Matching GNF and Model: For recognizing unseen images and many other applications, sometimes it is necessary to compute the similarity between a GNF (in an unseen image) and a model. When matching a unigram GNF and a model, first they must share the same bounding feature ID and the same GNF group type, otherwise the similarity is defined as

$$S_m = 0. \tag{5}$$

If the group type is a nested feature group, the similarity is computed by

$$S_m = \sum_{p=1}^{P} W_p S_i + \sum_{q=1}^{Q} W_q S_g \tag{6}$$

where P and Q are the number of ID matches and geometric matches respectively, W_p and W_q are weights of matched member features in the model, and S_i and S_g are the scoring weights of ID match and geometric match respectively (1 and 6 in our experiments). If the group type is a neighbor feature group, the similarity, considering the weak geometric constraint, is calculated by

$$S_m = \sum_{p=1}^{P} W_p S_i + \sum_{q=1}^{Q} \frac{W_q S_g}{4}. \tag{7}$$

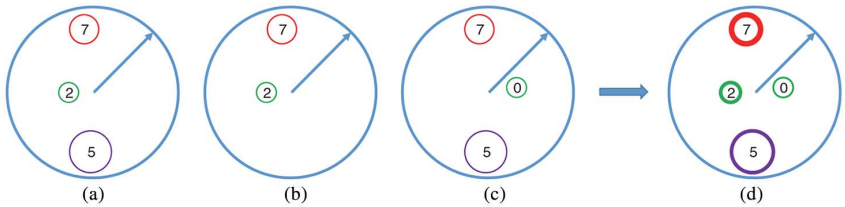


Fig. 7. Visualization of super code modeling. In this example, there are three unigram GNFs in the cluster. Different colors represent different visual word IDs and line width of member features in the model illustrates the weights. (a) Unigram GNF 1. (b) Unigram GNF 2. (c) Unigram GNF 3. (d) Model.

For n-gram GNFs, the similarity score is the sum of all unigram scores if all n bounding IDs and group types match, otherwise it is 0.

IV. Experiments

We evaluate the proposed framework and algorithms on a variety of datasets for partial-duplicate clustering and visual pattern discovery. We will first introduce the experiments on some small datasets, and then extend to Clickture datasets with 40 million web images.

A. Evaluations on Small Datasets

We first introduce the datasets, evaluation measures, and baseline algorithms. Then, the proposed algorithms are evaluated and compared.

1) Datasets: We adopted two datasets with ground truth clusters to evaluate the clustering performance of our framework. The first one is the PartialDup dataset used in [9]. It contains 782 images with 20 clusters. Images from the same cluster are partial-duplicate images obtained from the web. The second is the UKBench dataset [29]. It has 10200 images in total and every four images are in one cluster, resulting in 2550 clusters. The images in each cluster have the same object taken from different views. Note that for web images which we focus on, the partial-duplicates are mainly caused by manipulations rather than view changes, so we will prefer best settings of PartialDup dataset for larger datasets.

2) Evaluation Measures: There are a variety of evaluation measures in the literature [3], [5], [6]. In order to extend the evaluation measures to datasets without ground truth in later experiments, we designed a new method to measure the clustering performance. Different evaluation measures will not change the conclusions.

For each ground truth cluster C_{GT}, we first find all resulting clusters with matched images, denoted as candidate clusters. For each candidate cluster C_{CD}, we calculate its precision (P), recall (R), and F-measure (F) as below. The candidate cluster with the best F is selected, and the corresponding P, R and F are considered as results of the ground truth cluster C_{GT}.

$$P = \frac{|C_{GT} \cap C_{CD}|}{|C_{CD}|} \tag{8}$$

$$R = \frac{|C_{GT} \cap C_{CD}|}{|C_{GT}|} \tag{9}$$

$$F = \frac{2PR}{P+R} \tag{10}$$

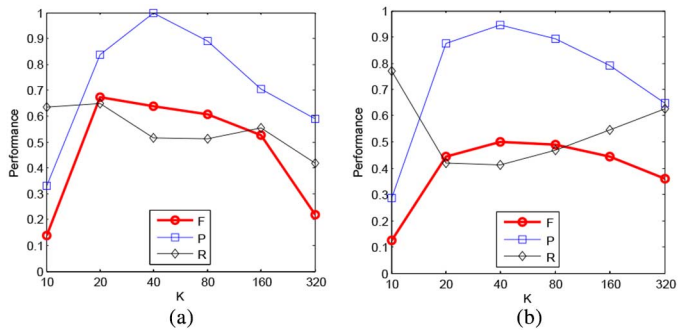


Fig. 8. Performance achieved by the baseline LFC algorithm with the best matching threshold T under different numbers of top features K. (a) and (b) show results on the PartialDup and UKBench datasets, respectively. Note that F is not necessarily between P and R as the three measures are averaged independently.

We average the three measures over all ground truth clusters to get the final results.

3) Baseline Algorithm: We treat the BoW-based clustering by search algorithm as our baseline algorithm, which was actually considered as the upper bound of local-feature-match-based partial-duplicate clustering in web-scale databases in [1], [2]. Note that the min-hash-based solution [3], [5] is also an efficient approximation of the BoW-based search approach.

In the baseline algorithm, top K local features in each image are extracted and quantized using the same codebook as in our framework. For every image in the dataset, we take it as a query and match it with all other images. If the number of matched feature pairs is above a threshold T, the image will be retrieved. All retrieved images plus the query image form a cluster. This simple local-feature-based algorithm (LF) can be further improved by pairwise comparison (LFC). Instead of querying the dataset, we directly conduct ϵ-clustering on the dataset with the same T as the distance threshold. LFC performs slightly better than LF in our experiments, but it is not as scalable as LF. LFC is used as a baseline on small datasets. For the Clickture datasets, considering the scalability issue, we use LF as the baseline. Experiments show that changing baseline from LF to LFC or vice versa will not affect the results and conclusions.

Parameters K (number of top features) and T (matching threshold) are important for the clustering result. With a fixed K, a larger T increases the precision while decreases the recall. Thus, an appropriate T is required. The influence of K is more interesting. Fig. 8 shows the performance curves on two datasets with K changed. For each K we adopt the best T for evaluation. The conclusion is that a moderate K performs the best. The reason is that a small K leads to too few features,

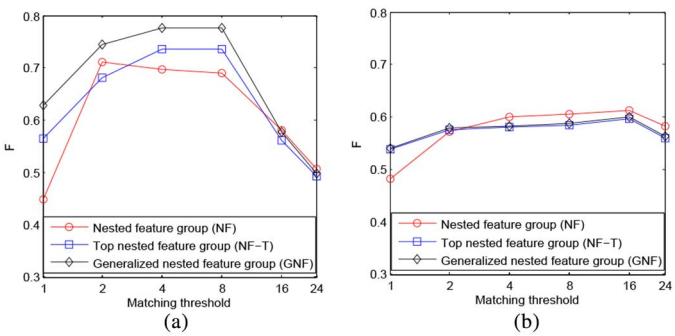


Fig. 9. Clustering performance (F-measure) of our framework with different group construction methods and matching thresholds on (a) PartialDup and (b) UKBench datasets.

while a large K leads to too many weak features with noises. For the PartialDup dataset the best configuration is $K = 20$ and $T = 2$. For the UKBench dataset it is $K = 40$ and $T = 4$. Results with these parameters will be compared with ours.

4) Evaluation on Different Settings: In this section, we evaluate the feature group construction methods and GNF matching thresholds. The final clustering results after cluster merging and post-processing are used for comparison, in which the feature group construction part has three options: a) all nested feature groups (NF), b) only top nested feature groups (NF-T), and c) the proposed generalized nested feature (GNF).

Fig. 9 shows the comparison results of F-measure with different thresholds and group construction methods. We can see that on the PartialDup dataset, GNF performs best among the three methods, while NF is the worst. NF-T improves precision by removing noise groups. GNF further improves recall by the use of neighbor feature groups, especially for image clusters with simple shapes and patterns. We also find that an appropriately chosen threshold helps balance precision and recall.

However, on the UKBench dataset group selection decreases the performance, and neighbor feature groups only slightly improve the results. That is because UKBench contains larger images than PartialDup, and more effective local features can be extracted from each image. Thus, it becomes less discriminative when using only top features. Another reason is that, the images in UKBench are photos taken from different views, which are likely to have complex textures, so most features belong to nested feature groups rather than neighbor groups.

Our goal is to conduct large-scale image clustering on web images, in which thumbnail images will be leveraged for efficiency, and images with simple shapes and textures such as clipart images and logo images will exist. Thus, the target dataset is more similar to PartialDup than UKBench. Since GNF can generate fewer groups and is more robust for images with fewer features, we adopt GNF as the feature group construction method in the next experiments.

5) Comparison of Algorithms: Finally, we compare our framework with the baseline. We use SC (short for super code) to refer to the final clustering results of the proposed framework. Both LFC and SC use the best parameter settings, including K, T and GNF matching threshold. Table I shows the results. It is obvious that our framework outperforms the baseline, with an improvement of about 0.1 absolute F-measure on both datasets. This shows the effectiveness of our framework.

TABLE I
COMPARISONS OF DIFFERENT ALGORITHMS ON PARTIALDUP AND UKBENCH DATASETS

Dataset	*Algorithm*	*F*	*P*	*R*
PartialDup	LFC	0.672	0.838	0.650
PartialDup	SC	0.776	0.900	0.756
UKBench	LFC	0.499	0.947	0.412
UKBench	SC	0.600	0.917	0.513

B. Evaluations on Clickture Datasets

We further evaluate the proposed framework on larger scale and practical web image datasets. First, we introduce the experiment settings, including datasets, evaluation measures, and baseline algorithm. Then, experiments are conducted to evaluate the proposed algorithms and framework on partial-duplicate clustering. After that, we illustrate the effectiveness of visual pattern discovery and GNF modeling, followed by parallelizability and statistics of the framework.

1) Datasets: The Clickture-Lite and Clickture-Full datasets [30] are used in the following experiments. Clickture-Lite contains 1 million thumbnail images from a commercial image search engine, while Clickture-Full is a superset of Clickture-Lite, containing 40 million images. They are good samples of web images. Each image in these two datasets has corresponding search engine queries and their click counts by real users. Such information was collected from search engine logs and provides a good description of image contents. We use Clickture-Lite to further evaluate clustering algorithms, and Clickture-Full to test the scalability and the visual patterns.

2) Evaluation Measures: Clickture-Lite does not have ground truth clusters, and it is hard to manually label the whole dataset. Thus, we designed a fair approach to compare two algorithms without full ground truth. The basic idea is to first make the two methods achieve similar precision by adjusting parameters, and then use one as ground truth to evaluate the recall of the other one.

In our case, we compare the clustering results of LF (baseline local feature matching) and SC (final output of our framework) algorithms. Next we will give the details of forward and inverse evaluations.

i) *Forward evaluation*. We first randomly sample some clusters from the results of LF (500 clusters in this paper). Each cluster is manually labeled whether it was a correct partial-duplicate cluster. A strict labeling rule is used. A cluster is labeled as positive only if it has 100% precision. The LF precision P_{LF} is given by the number of positive clusters divided by the number of sampled clusters. Then we keep the positive clusters as ground truth to evaluate the recall and extension of SC.

The recall R_{SC} measures the proportion of images that SC can correctly discover in each positive LF cluster, while the extension E_{SC} measures the number of images that SC can discover compared to LF. For each positive LF cluster C_{LF}, we first calculate the recall of all SC clusters, and select the SC cluster with the best recall as the matched SC cluster. The recall and extension of that SC

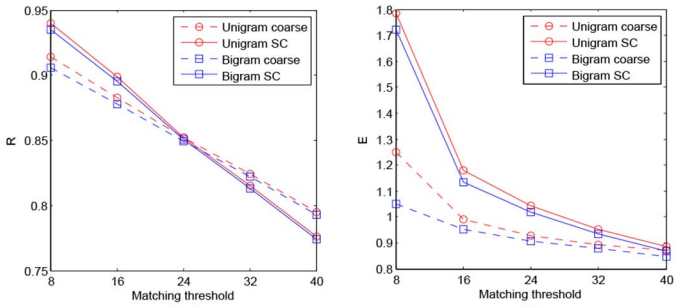


Fig. 10. (a) Recall and (b) extension of our framework (Bigram SC) with different GNF matching thresholds on Clickture-Lite dataset. Performance without bigrams (Unigram SC) and performance of coarse clusters without cluster merging and refining (Bigram/Unigram coarse) are also presented for comparison.

cluster C_{SC} are adopted for this LF cluster, as defined below.

$$R_{SC} = \frac{|C_{LF} \cap C_{SC}|}{|C_{LF}|} \tag{11}$$

$$E_{SC} = \frac{|C_{SC}|}{|C_{LF}|} \tag{12}$$

Averaging over all LF clusters results in the final R_{SC} and E_{SC} measures.

ii) *Inverse evaluation*. The steps are very similar to forward evaluation. We could get P_{SC}, R_{LF} and E_{LF} by switching the roles of LF and SC in forward evaluation.

In summary, from forward evaluation, we obtain the precision of LF P_{LF}, as well as the recall and extension of SC R_{SC} and E_{SC}, while from inverse evaluation, we get P_{SC}, R_{LF} and E_{LF}. A large R_{SC} means that SC discovers most images that LF can find, while a large E_{SC} indicates that SC discovers more diverse images that LF cannot find. Because the data are not fully labeled, we do not know the accuracy of the extension value. However, since we can make the precision of the two algorithms at the same level, the comparison between E_{LF} and E_{SC} does make sense.

3) Baseline Algorithm: In the baseline algorithm LF, we set the number of top features $K = 40$ and match threshold $T = 8$. The value of K was set according to the experiments on small datasets. Because our framework has high precision ($P_{SC} \geq 0.93$), we need to find a configuration with high precision for LF. Thus T was set to 8, which is the minimum number that ensures clustering precision more than 90%. With this configuration we have $P_{LF} = 0.912$. The average number of images of the 500 sampled LF clusters is 23.23, and the average number of images of the 456 correct ground truth LF clusters is 13.31.

4) Bigrams and Thresholds: We want to find a proper GNF matching threshold Th to get a similar precision as LF, and then compare recall and extension with LF. Fig. 10 shows the recall and extension curves with Th from 8 to 40. $Th = 8$ means that two GNFs (with the same bounding ID) have a geometric match. It is the minimal value we can adopt to guarantee the geometric relationship, and a lower threshold will cause the precision to drop fast. As Th increases, R_{SC} and E_{SC} both decrease, and we find that the precision P_{SC} (Bigram SC) achieves 95% when $Th = 8$, and continues increasing with a larger (stricter) Th. Thus, $Th = 8$ is a good choice for GNF matching.

Fig. 10 also shows the performance without bigram GNFs (Unigram SC) and the performance of coarse clusters without cluster merging and refining (Bigram/Unigram coarse) for comparison. Bigram increases the precision by combining unigrams (for example by 2% when $Th = 8$), with the slight cost of recall and extension. The performance of coarse clusters is much worse than SC when the matching threshold is relatively small, showing the necessity of cluster merging step. For large thresholds it becomes better than SC, because they make the post-processing step after cluster merging too strict, and thus affect the recall of SC.

TABLE II
COMPARISON OF LF AND SC ON CLICKTURE-LITE DATASET

Method	*Algorithm*	*P*	*R*	*E*
Unigram	LF	0.912	0.840	0.871
Unigram	SC	0.930	0.940	1.783
Bigram	LF	0.912	0.889	0.937
Bigram	SC	0.950	0.935	1.722

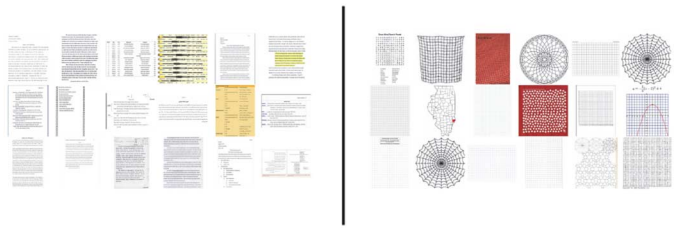
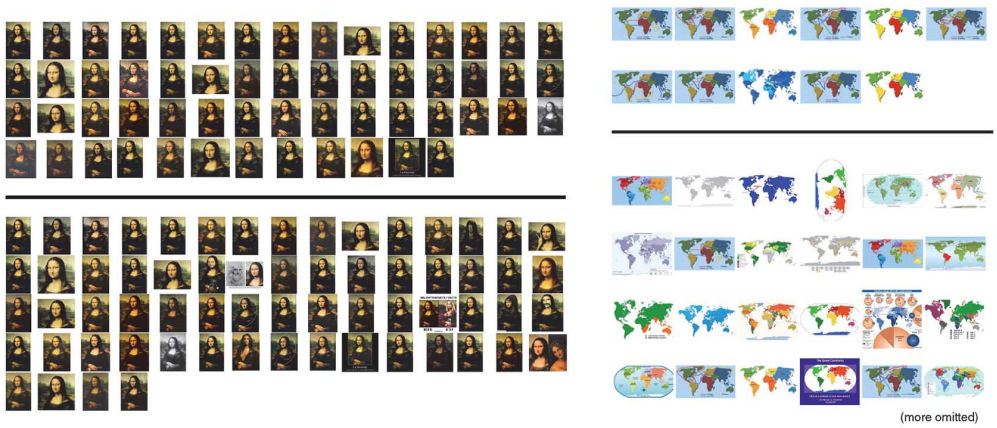


Fig. 11. Bad cases generated by the baseline algorithm. That is because local features are not discriminative for text and grid patterns. However, our framework can avoid this kind of results by using more discriminative representation, i.e., n-gram GNF.

5) Comparison of Algorithms: Table II shows the evaluation results of LF and SC with their best parameters. Results without and with bigrams are all provided. We can see that our framework greatly outperforms the baseline method on all measures. With an appropriate GNF matching threshold, our framework can achieve better precision while discovering more diverse image clusters. As aforementioned, the usage of bigram GNFs improves the precision at the cost of recall loss. As the increase of precision is more obvious (and for many applications, more crucial) than the decrease of recall, bigram becomes a necessity in our framework.

Fig. 11 shows two cases in which LF fails, where images contain texts and grids. These cases are quite common in web images, which are hard to differentiate by local features. However, our framework can avoid most of them by using more discriminative representation, i.e., n-gram GNF. For a wrong cluster like this, SC usually either generates no corresponding cluster or produces some subsets of it that contain full-duplicates.

Fig. 12 shows two LF clusters and their corresponding SC clusters, the Mona Lisa and the world map. They are two examples of popularly manipulated images. A variety of transformations and modifications exist in these images, some of which are not easily identified even by humans. We can see that SC can detect more diverse images, apparently outperforming LF. Meanwhile, it does not sacrifice precision.



(a) (b)

Fig. 12. Two LF clusters (top) and corresponding SC clusters (bottom). We can see that SC clusters have better recall and diversity. (a) Mona Lisa. (b) World map.

Fig. 13. Sampled images from three clusters detected on Clickture-Full dataset. The selected clusters are all from top 20 clusters ordered by cluster size, each of which has more than 1000 images with almost perfect precision.

We also run our framework with the Clickture-Full dataset, with the best parameter settings on Clickture-Lite. Fig. 13 shows some sampled images from detected clusters. The selected clusters are all from top 20 clusters ordered by cluster size, each of which has more than 1000 images with almost perfect precision. Some of the images are highly modified, including cropping, overlapping and occlusion. It shows that the increase of dataset size does not lower clustering performance. On the contrary, as there are more diverse images in larger databases, the advantages of our framework might be more obvious.

6) Visual Patterns and Applications: We now have a close look at the discovered visual patterns. Fig. 14 shows some examples of image visual pattern representations. The areas of the bounding features in GNFs are shown as circles, and corresponding local patches are cropped for illustration. We observe that most of these visual patterns are quite discriminative.

Next we link semantics to visual patterns. We assume that each image has several keywords (either surrounding texts or click data), which describe the image content. The keywords of an image can be propagated to an n-gram GNF if the image contains this GNF. Thus, each n-gram GNF corresponds to a collection of keywords. For a pattern model $pattern$ with keyword collection C_k, we can define the probability $P(w|pattern)$ as

$$P(w|pattern) = \frac{|\{k|k \in C_k, k = w\}|}{|C_k|} \tag{13}$$

where w is a specific keyword in the dictionary. This probability distribution describes the relationship between keywords and patterns. Similarly we can get the probability distribution of keywords given a cluster $P(w|cluster)$. The probabilistic representation of pattern semantics and cluster semantics will be useful to a variety of applications. We leveraged click logs of each image in Clickture datasets to generate the image keywords. For each image we compute the distribution of keywords in its click logs and select top three keywords.

Fig. 15 lists some frequently occurring visual patterns with top three keywords ordered by $P(w|cluster)$. We can see that the visual patterns and keywords have good semantic relationship. Furthermore, the local patches are quite discriminative.

One application is to use discovered visual patterns to recognize visual patterns of unseen images. To guarantee fast matching, we build an index for all pattern models by hashing bounding IDs and group types, so that only a very small portion of models need to be checked when matching GNFs with our models.

We first compute all GNFs of an unseen image using the same way in our framework. For each GNF, we find all matched models in the index and compute similarity scores. Let M be the number of matched models and S_j be the similarity with the jth model. For a keyword w we define its probability to this GNF as

$$P(w|GNF) = \frac{\sum_{j=1}^{M} S_j P(w|pattern_j)}{F} \tag{14}$$

where F is the normalization factor.

Fig. 16 shows some example results. The pattern models were learnt from Clickture-Full, and the testing images are from a separate image set without intersection with Clickture-Full. We can see that the detected local patterns and assigned keywords matched well.

7) Parallelizability and Statistics: To test the parallelizability of the proposed framework, we leveraged a distributed system with thousands of machines to implement this framework. We tested on the Clickture-Lite dataset with 50 and 2000 nodes in

Fig. 14. Some example images with discovered visual patterns and their logical representations.

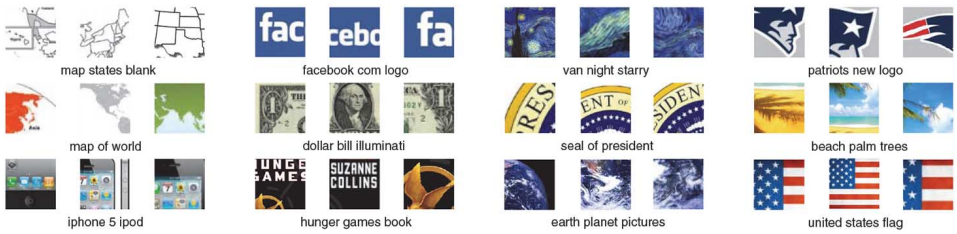


Fig. 15. Illustrations of frequently occurring visual patterns and corresponding top three cluster keywords. Visual patterns are visualized as local patches.

Fig. 16. Example results of detected local patterns for unseen images. For each result the circle indicates position and size of the local region, and top three keywords with their probabilities are listed next to the image.

turn. The ϵ-clustering by GNF matching has the highest complexity and cost the most time. It took about 3 hours on 50 nodes, but only 5 minutes on 2000 nodes. This shows the full parallelizable property of the framework, since when the number of nodes increased to 40 times, the running time reduced to about 1/40. The whole framework cost about 8 minutes using 2000 machines, which is quite efficient.

With 2000 nodes, Clickture-Full was also completed in a reasonable time. GNF matching took about 4 hours, and the whole framework cost less than 7 hours. By comparing with Clickture-Lite, we find that running time is almost linear to the dataset size. One reason is that we removed some stopwords before GNF matching, most of which are meaningless. We believe that with careful parameter selection, algorithm speedup, and more computing resources, it is not hard to scale this framework to billion-level database.

Table III summarizes some statistics of the two datasets. We also find that the results on these two datasets have similar pre-

TABLE III
SOME STATISTICS ON CLICKTURE-LITE AND CLICKTURE-FULL DATASETS

Dataset	Lite	Full
#images	1,000,000	40,000,000
#clusters with 3+ images	17,876	1,331,218
#clusters with 10+ images	953	156,903
%images with 2+ duplicates	8.3%	20.5%
#GNF types	721,580	32,600,318

cision (about 95%). That means with our framework, scaling up the database will help discover more clusters and visual patterns without precision loss.

V. CONCLUSION AND FUTURE WORK

We have presented in this paper a novel and highly scalable framework to cluster partial-duplicate images and discover visual patterns in web scale image databases. Each step in the framework was deliberately designed to ensure good performance, fast speed, and high parallelization. Experimental results showed that our framework performed much better than the baseline algorithm (which sometimes was considered as an upper bound in web-scale image clustering), and the discovered visual patterns were discriminative and semantically meaningful. Experiments also verified its good scalability.

Despite of the superior performance of our framework, it has some limitations. As our framework requires a relatively strict matching rule, it is not very robust to large variations of viewpoint and non-rigid objects. One possible future work is to explore more robust visual pattern representation to tackle this variance problem. Furthermore, we will try to scale up the database to the order of billions, and then build a visual pattern knowledge base, based on which many computer vision tasks can be completed. Some vertical applications related to visual patterns may also be considered in the future, such as logo recognition [31] and [32] and product recognition.

REFERENCES

[1] X.-J. Wang, L. Zhang, and C. Liu, "Duplicate discovery on 2 billion internet images," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog. Workshops*, Jun. 2013, pp. 429–436.

[2] X.-J. Wang, L. Zhang, M. Liu, Y. Li, and W.-Y. Ma, "ARISTA – Image search to annotation on billions of web photos," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, Jun. 2010, pp. 2987–2994.

[3] O. Chum and J. Matas, "Large scale discovery of spatially related images," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 2, pp. 371–377, Feb. 2010.

[4] A. Torralba, R. Fergus, and W. Freeman, "80 million tiny images: A large dataset for non-parametric object and scene recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 11, pp. 1958–1970, Nov. 2008.

[5] O. Chum, M. Perdoch, and J. Matas, "Geometric min-Hashing: Finding a (thick) needle in a haystack," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, Jun. 2009, pp. 17–24.

[6] D. Lee, Q. Ke, and M. Isard, "Partition min-Hash for partial duplicate image discovery," in *Proc. Eur. Conf. Comput. Vis.*, 2010, pp. 648–662.

[7] J. Yuan, Y. Wu, and M. Yang, "Discovery of collocation patterns: From visual words to visual phrases," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, Jun. 2007, pp. 1–8.

[8] P. Xu, L. Zhang, K. Yang, and H. Yao, "Nested-SIFT for efficient image matching and retrieval," *IEEE Multimedia Mag.*, vol. 20, no. 3, pp. 34–46, Jul.–Sep. 2013.

[9] Z. Wu, Q. Ke, M. Isard, and J. Sun, "Bundling features for large scale partial-duplicate web image search," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, Jun. 2009, pp. 25–32.

[10] Y.-T. Zheng, M. Zhao, S.-Y. Neo, T.-S. Chua, and Q. Tian, "Visual synset: Towards a higher-level visual representation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, Jun. 2008, pp. 1–8.

[11] S. Zhang, Q. Tian, G. Hua, Q. Huang, and S. Li, "Descriptive visual words and visual phrases for image applications," in *Proc. ACM Int. Conf. Multimedia*, 2009, pp. 75–84.

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

[13] L. Kennedy and S.-F. Chang, "Internet image archaeology: Automatically tracing the manipulation history of photographs on the web," in *Proc. ACM Int. Conf. Multimedia*, 2008, pp. 349–358.

[14] C.-W. Ngo, C. Xu, W. Kraaij, and A. El Saddik, "Web-scale near-duplicate search: Techniques and applications," *IEEE Multimedia Mag.*, vol. 20, no. 3, pp. 10–12, Jul.–Sep. 2013.

[15] J. Sivic and A. Zisserman, "Video Google: A text retrieval approach to object matching in videos," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2003, vol. 2, pp. 1470–1477.

[16] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman, "Object retrieval with large vocabularies and fast spatial matching," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, Jun. 2007, pp. 1–8.

[17] Y. Zhang, Z. Jia, and T. Chen, "Image retrieval with geometry-preserving visual phrases," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, Jun. 2011, pp. 809–816.

[18] X. Shen, Z. Lin, J. Brandt, S. Avidan, and Y. Wu, "Object retrieval and localization with spatially-constrained similarity measure and k-NN re-ranking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, Jun. 2012, pp. 3013–3020.

[19] X. Liu, Y. Lou, W. Yu, and B. Lang, "Search by mobile image based on visual and spatial consistency," in *Proc. IEEE Int. Conf. Multimedia Expo*, Jul. 2011, pp. 1–6.

[20] J. Song, Y. Yang, Z. Huang, H. Shen, and R. Hong, "Multiple feature hashing for real-time large scale near-duplicate video retrieval," in *Proc. ACM Int. Conf. Multimedia*, 2011, pp. 423–432.

[21] O. Chum and J. Matas, "Fast computation of min-Hash signatures for image collections," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, Jun. 2012, pp. 3077–3084.

[22] Z. Liu, H. Li, W. Zhou, R. Hong, and Q. Tian, "Uniting keypoints: Local visual information fusion for large-scale image search," *IEEE Trans. Multimedia*, vol. 17, no. 4, pp. 538–548, Apr. 2015.

[23] J. Liu and Y. Liu, "GRASP recurring patterns from a single view," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, Jun. 2013, pp. 2003–2010.

[24] H. Kang, M. Hebert, and T. Kanade, "Discovering object instances from scenes of daily living," in *Proc. IEEE Int. Conf. Comput. Vis.*, Nov. 2011, pp. 762–769.

[25] H. Kang, M. Hebert, A. Efros, and T. Kanade, "Connecting missing links: Object discovery from sparse observations using 5 million product images," in *Proc. Eur. Conf. Comput. Vis.*, 2012, pp. 794–807.

[26] W. Zhang, H. Li, C.-W. Ngo, and S.-F. Chang, "Scalable visual instance mining with threads of features," in *Proc. ACM Int. Conf. Multimedia*, 2014, pp. 297–306.

[27] H. Wang, G. Zhao, and J. Yuan, "Visual pattern discovery in image and video data: A brief survey," *Wiley Interdisciplinary Rev.: Data Mining Knowl. Discovery*, vol. 4, no. 1, pp. 24–37, 2014.

[28] S. Winder and M. Brown, "Learning local image descriptors," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, Jun. 2007, pp. 1–8.

[29] D. Nister and H. Stewenius, "Scalable recognition with a vocabulary tree," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, Jun. 2006, vol. 2, pp. 2161–2168.

[30] X.-S. Hua, L. Yang, J. Wang, J. Wang, M. Ye, K. Wang, Y. Rui, and J. Li, "Clickage: Towards bridging semantic and intent gaps via mining click logs of search engines," in *Proc. ACM Int. Conf. Multimedia*, 2013, pp. 243–252.

[31] S. Romberg, L. Pueyo, R. Lienhart, and R. van Zwol, "Scalable logo recognition in real-world images," in *Proc. Int. Conf. Multimedia Retrieval*, 2011, pp. 25–32.

[32] S. Romberg and R. Lienhart, "Bundle min-Hashing for logo recognition," in *Proc. Int. Conf. Multimedia Retrieval*, 2013, pp. 113–120.

Wei Li received the B.E. degree in computer science and technology from Tsinghua University, Beijing, China, in 2008, and is currently working toward the Ph.D. degree in computer science and technology at Tsinghua University.

Since 2013, he has been a Research Intern with Microsoft Research Asia, Beijing, China. His research interests include computer vision, multimedia analysis, and the mining of visual patterns and visual instances from large-scale image databases.

Changhu Wang (S'07–M'10–SM'14) received the B.E. and Ph.D. degrees from the University of Science and Technology of China, Hefei, China, in 2004 and 2009, respectively.

Since June 2009, he has been with Microsoft Research Asia, Beijing, China, where he is currently a Researcher with the Multimedia Search and Mining Group. His research interests include multimedia retrieval and computer vision.

Lei Zhang (M'04–SM'11) received the B.E., M.S., and Ph.D. degrees in computer science and technology from Tsinghua University, Beijing, China, in 1993, 1995, and 2001, respectively.

He was previously a Senior Researcher with Microsoft Research Asia, Beijing, China, for 12 years, and was then a Principal Development Manager with Bing Image Search, Redmond, WA, USA, for two years. He is currently a Senior Researcher with Microsoft Research, Redmond, WA, USA. He holds 40 U.S. patents. His research interests include image search and computer vision.

Dr. Zhang has served as Program Area Chair or Committee Member for many conferences.

Yong Rui (SM'04–F'10) received the B.E. degree from Southeast University, Dhaka, Bangladesh, M.S. degree from Tsinghua University, Beijing, China, and Ph.D. degree from the University of Illinois at Urbana-Champaign, Champaign, IL, USA, in 1991, 1994, and 1999, respectively. He also holds a Microsoft Leadership Training Certificate from the Wharton Business School, University of Pennsylvania, Philadelphia, PA, USA.

He is currently a Senior Director and Principal Researcher at Microsoft Research Asia, Beijing, China. He holds 56 issued U.S. and international patents. He has authored or coauthored 16 books and book chapters, and over 100 referred journal and conference papers.

Dr. Rui is a Fellow of the IAPR and SPIE, and a Distinguished Scientist of the ACM. He is an Executive Member of ACM SIGMM, and the founding Chair of its China Chapter. He is the Editor-in-Chief of the IEEE MULTIMEDIA MAGAZINE, an Associate Editor of the *ACM TOMM*, a founding Editor of the *International Journal of Multimedia Information Retrieval*, and a founding Associate Editor of the IEEE ACCESS. He was an Associate Editor of the IEEE TRANSACTIONS ON MULTIMEDIA (2004–2008), the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY (2006–2010), the *ACM/Springer Multimedia Systems Journal* (2004–2006), and the *International Journal of Multimedia Tools and Applications* (2004–2006). He is an Organizing Committee Member and Program Committee Member of numerous conferences including ACM MM, ACM ICMR, IEEE ICME, SPIE ITCom, and ICPR. He was the General Co-Chair of ACM MM in 2009 and 2014, ICMR in 2006 and 2012, and ICIMCS in 2010, and the Program Co-Chair of ACM MM in 2006, PCM in 2006, and ICME in 2009. He is/was on the Steering Committees of ACM MM, ICMR, ICME and PCM.

Bo Zhang received the B.E. degree in automatic control from Tsinghua University, Beijing, China, in 1958.

He was a Visiting Researcher at the University of Illinois at Urbana-Champaign, Champaign, IL, USA, from 1980 to 1982. He is currently a Professor with the Department of Computer Science and Technology, Tsinghua University. His research interests include machine learning, pattern recognition, knowledge engineering, intelligent robotics and intelligent control.

Dr. Zhang is a member of the Chinese Academy of Sciences. He was the recipient of the European Artificial Intelligence Prize in 1984. He won the First Class Science and Technology Progress Prize in 1987, 1993, and 1998, and the Third Class Science and Technology Progress Prize in 1995 and 1999.