Duplicate Discovery on 2 Billion Internet Images

Xin-Jing Wang, Lei Zhang
Microsoft Research Asia
5 Danling Street, Beijing, China
{xjwang, leizhang}@microsoft.com

Ce Liu
Microsoft Research New England
1 One Memorial Drive, Boston, USA
celiu@microsoft.com

Abstract

Duplicate image discovery, or discovering duplicate image clusters, is a challenging problem for billions of Internet images due to the lack of good distance metric which both covers the large variation within a duplicate image cluster and eliminates false alarms. After carefully investigating existing local and global features that have been widely used for large-scale image search and indexing, we propose a two-step approach that combines both local and global features: global descriptors are used to discover seed clusters with high precision, whereas local descriptors are used to grow the seeds to cover good recall. Using efficient hashing techniques for both features and the MapReduce framework, our system is able to discover about 553.8 million duplicate images from 2 billion Internet images within 13 hours on a 2,000 core cluster.

1. Introduction

Current image search engines heavily rely on image-to-text relevance whereas the surrounding texts extracted from the Web are often inadequate or inaccurate. Therefore, image-to-text techniques, *i.e.* annotating images with semantic and informative text descriptions, are very useful for improving text-to-image search. One key idea of image-to-text is to propagate textual annotations from similar images in a large database to a query image [23, 13, 15]. However, such approaches can hardly scale up to billions of Internet images as web-scale similar image retrieval still remains a challenging problem [10, 7, 6, 8].

As many images tend to have duplicates on the Web, and each instance of a duplicate may be annotated differently, it is straightforward to aggregate annotations from the duplicates of an input image to obtain accurate, comprehensive textual annotations for the input. Such an idea has been successfully explored in [24]. Although obtaining accurate text tagging for images with duplicates may seem insufficient in the computer vision community, the amount of duplicates is a good indication for the importance of an image. Therefore, it is of great value to obtain accurate tagging for

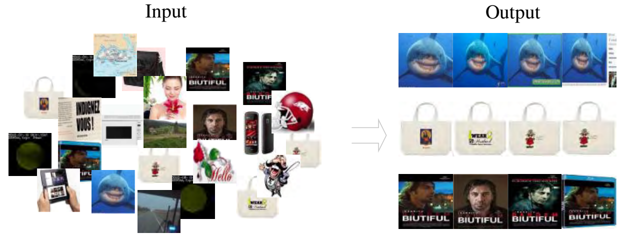

Figure 1. The problem of duplicate discovery is to take the entire dataset as input and output the duplicate image clusters.

frequently appeared, duplicate images on the Web.

It is very inefficient to retrieve duplicates for each image on the web since in this case, to annotate all images with duplicates would require to take each image in the database and query the database with it, which is quadratic in the size of the database and hence not feasible for large databases [18, 2, 9]. A natural next step is then to discover the duplicate image clusters from all the web images, to aggregate text tags for each duplicate image cluster, and to assign the tags to all the instances of a cluster. In this paper, we study this *duplicate discovery* problem for billions of images on the Web, as illustrated in Figure 1: the input is billions of images, and the output is duplicate image clusters. Based on the discovered clusters, image annotation can thus be both reliable and efficient through aggregation and assignment within each cluster.

Although duplicate discovery may seem trivial at a glance, it is indeed a very challenging problem. We are not only interested in clustering *exact duplicate*, but also *global duplicate* and *near duplicate* (see Figure 2). Exact duplicates are images having exactly the same appearance. Global duplicates tolerate small alterations on the content, whereas near duplicates allows rotating, cropping, transforming, adding, deleting and altering image contents. It is challenging to define a good distance metric such that within a threshold all samples are duplicates (high precision), and all the duplicates fall into the hyper-sphere defined by the threshold (high recall). Furthermore, it is nontrivial to discover duplicate image clusters in web-scale images.

Existing large-scale duplicate discovery approaches adopt two different schemes [2, 3, 9, 11]. One extracts

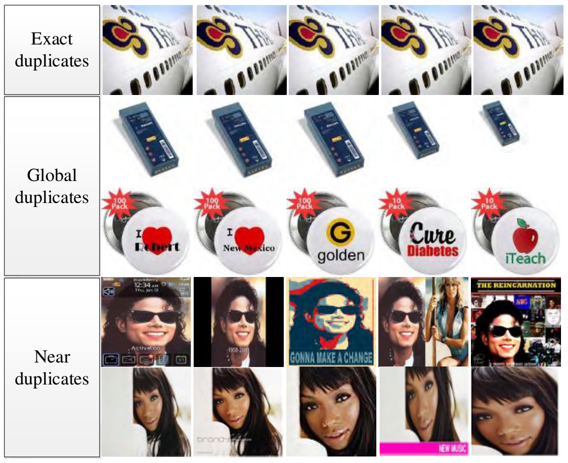

Figure 2. Duplicate images are close enough in image space, measured by selected distance metrics and with certain parameter settings. Global image features are good at identifying *exact duplicates* and *global duplicates*, which are exactly the same or tolerate small alterations such as subtle scale variances and local pixel changes. *Near duplicates*, on the other hand, generally require local image descriptors to address crops, combinations, and alterations on pixels larger than the limit of a global feature-based distance metric.

local descriptors and represents an image in bag-of-visual words (BOW) [17] model. For example, Chum et al.[2] proposed to discover spatially related near-duplicate images by first generating seed images with min-Hash functions [1] and then performing BOW retrieval using the seeds as queries. The other represents an image in global feature-based compact codes [20, 19, 11]. For example, Liu et al. [11] conducted clustering on 1.5B images via nearest neighbor search in the MapReduce framework [4]. They first converted the re-scaled version of 64×64 pixels of an image to the Haar wavelet domain, and then reduced the dimensionality of the feature vectors via random projections.

In this paper, we carefully study the advantages and disadvantages of global and local image descriptors on large-scale duplicate discovery. We found that local features are able to discover near duplicates but often result in low precision and low efficiency. On the other hand, global features generate high precision and good efficiency, but low recall. To ensure both high precision and recall, we propose to combine global and local features in a coarse-to-fine approach: global features are used to partition the image space into buckets and to generate seed clusters in each bucket with high precision, whereas local descriptors are used to grow the seed clusters to discover near duplicates for high recall. This approach is based on the assumption that near duplicate images measured by local features are likely to reside in close buckets in the global feature space, so that we can grow seed clusters only locally in nearby buckets for efficiency. Experiments show that our approach has satisfying precision and recall on a 2B image set (crawled from the Internet). As the system is implemented in the MapReduce framework [4], duplicate image discovery on these 2B images is achieved within 13 hours on about 2,000 CPU cores in peak time.

2. Related Work

Large scale duplicate image discovery was often investigated in the context of finding spatially related images [2], where s-tuple min-Hash [1] values based on bag-of-words (BOW) representations are used to discover seed images, which are further used to retrieve near duplicates. The min-hash function was then improved by applying geometric constraints on image features using multiple independent min-hash functions[3]. This idea was extended by [9], where s-tuples of min-hash values are extracted within image partitions on grids.

Although these local feature-based approaches were shown to be feasible on millions of images, it is still challenging to directly apply them to billion-scale. First, false positives resulted from min-Hash is a serious problem on a large-scale dataset [9], which we also reinforce in our study (see Section 3). Second, the complexity of seed growing, *i.e.* image retrieval, is still high. Chum et al.[2] estimated the computational complexity of the seed growing step to be $O(NL)$, where L is the number of seeds and N the dataset size. We observed that $L \approx 0.2B$ when $N = 2B$ by our approach, resulting in $O(NL) \approx O(N^2)$.

On the other hand, global feature-based compact codes have been widely used for image retrieval or clustering on large-scale datasets [20, 19, 11]. For example, semantic hashing [16] is used to convert the GIST descriptor[14] to a few hundred bits [20] so that real-time search performed on millions of images achieves comparable recognition results to full GIST descriptors. Even raw pixels of tiny 32×32 images were shown to be useful in identifying semantically similar images from a set of 80M images [19]. Liu et al. [11] used Haar wavelets of image thumbnails to represent an image, and conducted image clustering on 1.5B images via tree-based nearest neighbor search.

However, the major drawback of global features is the recall. Using global features only can hardly identify near duplicates (especially with crops), as we will discuss later. Moreover, the approach of [11] tends to discover (disconnected) balls of nearest neighbor images by searching in multiple spill trees, resulting in near duplicates being split.

3. Global Descriptor vs. Local Descriptor

In this section, we study the performances of global and local feature based duplicate discovery on a small-scale ground truth dataset manually built. We did not perform exhaustive evaluations on all available compact codes or local descriptors [12, 14, 19, 7], but the descriptors used for

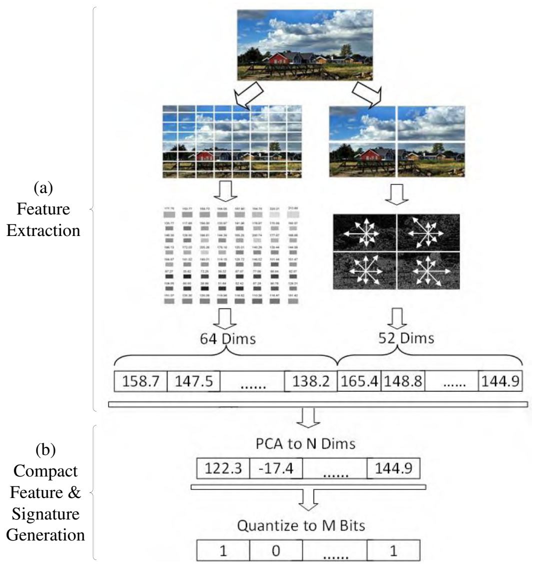

Figure 3. The feature extraction process.

the evaluation are orthogonal to their relative performances.

3.1. The Descriptors

The global descriptors we used in the study is vector quantized PCA-transformed histogram features of gridded images. The process is illustrated in Figure 3. We found this feature design is light-weight but effective in discovering global duplicates, and is very efficient to compute which is an important merit in processing billion-scale images.

Image blocking was shown a simple and effective way to produce descriptive image features [21, 27, 9]. We evenly partition an image into 8×8 blocks and calculate the mean gray value from each block. The image is again divided into regular 2×2 blocks, and a histogram of edge distribution in 12 directions is extracted from each block. We also keep one dimension of non-edge pixel ratio, which in total make up of a 116-dim raw feature. Figure 3(a) illustrates the process.

The 116-dim raw feature is then projected into a PCA space pre-learnt from a sufficiently large number of images (3.2 million) which are represented by the same type of raw features, and the least significant dimensions after projection are discarded. PCA projection helps reduce small noises and potential value drifting in raw features.

As a summary, an image is represented with two global descriptors. One is an n-bit hash signature used for space partitioning. Figure 3(b) illustrates the process. A signature is given by binarizing the PCA-ed raw features using the mean of each dimension as a threshold, *i.e.* if the feature value of a dimension is above a threshold, its corresponding bit is set to one, otherwise to zero. The other is a 24-dim PCA-ed feature vector. It is used for a confident similarity verification of two images.

We adopt the learning-based local descriptors proposed by Winder et al.[26], which was designed for image matching and 3D reconstruction. Various published local descriptors including SIFT [12] can be cast into the framework [26].

Table 1. Using global descriptors obtained higher precision and lower recall than using local descriptors for duplicate clustering. Our approach performs in between as it uses both global and local descriptors.

	pure global	our approach	pure local
precision	99.69%	99.30%	96.81%
recall	83.05%	87.02%	89.14%

3.2. The Ground Truth Dataset

The ground truth image set was manually collected. We randomly selected 386 images and performed near duplicate search on the 2B image set. The search is based on the same local descriptors and an image is represented in the BOW model. False positives in the search results were manually removed, which gave 8,837 images in total. Then we mixed the 8,837 images into 1M distractor images which are not duplicates of the image clusters.

3.3. The Evaluation

Table 1 provides the precisions and recalls of three different approaches. The method "pure global" first generates seed clusters by pair-wise similarity comparison (with a strict threshold) within each hash bucket, whereas buckets are formed by images sharing the same signatures. Then seed clusters are grown by checking its visual similarity across buckets whose Hamming Distances ($Dist_h$) of signatures satisfy $Dist_h \leq 2$. The method "our approach" is what presented in this paper. Both these two methods use 24-bit signatures to generate buckets. We adopt Chum et al's approach [2] as the "pure local" method, where 20 min-Hash functions and 3-tuple of min-Hash collision were used which are optimal parameters.

Table 1 shows that using only global descriptors obtained higher precision while lower recall than using only local descriptors. Our approach performs in between - its recall is higher than using only global descriptors whereas its precision is higher than using only local descriptors. These observations suggest that to collaborate global and local image descriptors is promising in balancing precision and recall.

Figure 4 shows sampled images from three duplicate image clusters of which "pure global" achieved high precision but "pure local" found false positives. False positive is a big issue for local feature based duplicate discovery, especially on large datasets (see more discussions in Section 5.2), which was also observed by Lee et al. [9].

Figure 5 illustrates the recall issue of the "pure global" method. Images in each row are near duplicates in the ground truth set, but are from separate clusters generated by the "pure global" method. Such images are generally different crop versions of the same objects.

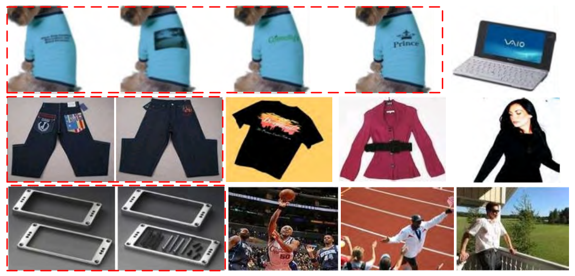

Figure 4. Examples of false positives of near duplicates identified using local features. Images in red blocks are duplicates identified by both global and local-feature based approaches, while the rest are false positives. Each row represents a cluster.

4. Duplicate Discovery from Billions of Images

In this section, we describe our approach, which leverages both global descriptors and local descriptors in a coarse-to-fine way. We use global features to divide the image set into subspaces in which the images are visually similar, and then perform clustering within each subspace to generate seed image clusters. Then we adopt local features to merge the seed clusters by checking near duplicates across similar subspaces. To make the entire process efficient, the similar subspaces are identified by global features.

4.1. Image space partitioning

A natural choice to split a large-scale image space into subspaces on which clustering is feasible is hashing. Though the ultimate goal is to learn compact binary codes of data-points whose Hamming distances correlate with semantic similarities (so that clustering can be performed within hash buckets and without growing clusters across hashing boundaries), semantic hashing is still an open research topic [16, 22, 25].

We use the hashing method described in Section 3.1 to generate image signatures, and images which have identical signatures are assigned into the same buckets. The advantage of our hashing technique lies in that the cost of training the hash function is much lower than existing semantic hashing approaches [16, 22, 25] and is feasible and efficient on large-scale datasets. Apparently the signatures learnt are not semantic, but the problem is not critical since the cluster merging step tackles the boundary issue of hashing, i.e. it will check across neighbor buckets.

4.2. Image Clustering with Global Descriptors

We perform image clustering within a bucket by checking the pair-wise similarity of two images, based on the 24-dim PCA-ed feature vector associated with an image. Note that 24 is a tradeoff between memory cost and performance and the length of a signature is unnecessarily the same as the dimension of feature vectors used for image clustering.

We propose an ϵ-clustering algorithm to discover image clusters within a bucket. Its idea is to initialize each image as one cluster, and then merge clusters via pair-wise similarity matching between images. We adopt a greedy strategy to do the merge, i.e. if there is one image x_i that is a neighbor of two distinct clusters A and B, then $\{A, B, x_i\}$ forms a new cluster. Algorithm 1 gives the pseudo code.

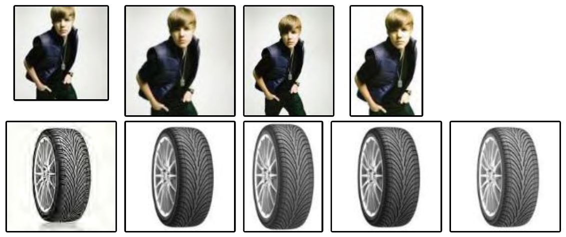

Figure 5. Global features fail to group cropped images so using global features only for clustering has recall issue. For each row, each image comes from a distinct duplicate image cluster.

In Algorithm 1, each image is associated with a flag δ which indicates its cluster membership, and the pair-wise matching procedure operates according to the flag status of two images, which makes the algorithm simple to imple-

Algorithm 1 The Algorithm of ϵ-clustering

```
Terms:
$N$: hash bucket size
$\{x_i | 1 \leqslant i \leqslant N\}$: images in a hash bucket.
$\mathcal{C}_i$: the $i$-th duplicate cluster, $1 \leqslant i \leqslant N$
$\delta_i \in \{1, ..., N\}$: cluster membership of $x_i$, $x_i \in \mathcal{C}_{\delta_i}$
$\epsilon$: distance threshold

Initialization:
$\mathcal{C}_i = \{x_i\}, 1 \leqslant i \leqslant N$
$\delta_i = -1, 1 \leqslant i \leqslant N$

Clustering:
for all $i = 1 : N$ do
  for all $j = i + 1 : N$ do
    if $Dist(x_i, x_j) \leqslant \epsilon$ then
      if $\delta_i = -1$ && $\delta_j = -1$ then
        $\delta_j = i$
        $\mathcal{C}_i = \mathcal{C}_i \cup \{x_j\}$ //$\epsilon$-ball creation
      end if
      if $\delta_i > 0$ && $\delta_j = -1$ then
        $\delta_j = \delta_i$
        $\mathcal{C}_{\delta_i} = \mathcal{C}_{\delta_i} \cup \{x_j\}$ //$\epsilon$-ball expansion
      end if
      if $\delta_i > 0$ && $\delta_j > 0$ then
        $\delta_j = \delta_i$
        $\mathcal{C}_{\delta_i} = \mathcal{C}_{\delta_i} \cup \mathcal{C}_{\delta_j}$ //$\epsilon$-ball merge
        $\mathcal{C}_{\delta_j} = \varnothing$
      end if
    end if
  end for
end for

return $\{\mathcal{C}_i | |\mathcal{C}_i| \geqslant 2, 1 \leqslant i \leqslant N\}$
```

Figure 6. The algorithm of ϵ-clustering.

ment. Specifically, consider two images x_i, x_j which are neighbors (i.e. $Dist(x_i, x_j) \leqslant \epsilon$ where $Dist(\cdot)$ is a distance measure and ϵ is a threshold):

- If both x_i and x_j have negative flags (i.e. $\delta_i, \delta_j = -1$), which means neither of them have ever been visited, then the two images get merged (i.e. $\mathcal{C}_i \cup \mathcal{C}_j$). This is called ϵ*-ball creation*.
- If x_i has been visited (i.e. $\delta_i > 0$, which means x_i belongs to a cluster whose size is larger than one) but x_j has not, then x_j (i.e. $\mathcal{C}_j$) is merged into the cluster of x_i (i.e. $\mathcal{C}_{\delta_i}$). We call it ϵ*-ball expansion*. Isolated duplicate images are identified in this step.
- If both x_i and x_j have ever been visited (i.e. $\delta_i, \delta_j > 0$), then their corresponding clusters are merged (i.e. $\mathcal{C}_{\delta_i} \cup \mathcal{C}_{\delta_j}$). This is called ϵ*-ball merge*, which groups two duplicate image clusters and improves recall.

The advantage of ϵ-clustering is threefold: 1) duplicate images may distribute on a skewed space (e.g. a horseshoe) rather than a normal ball, and the greedy strategy adopted by ϵ-clustering works on skewed spaces by connecting a list of small ϵ-balls to reproduce the space. This is superior to Liu et al's approach [11], which generates disconnected normal balls; 2) ϵ-clustering is efficient and applicable to large-scale datasets. Though the computational complexity of ϵ-clustering is $O(n^2)$, n is a controllable parameter by the hashing approach so that the task can be fulfilled within a limited time; and 3) the order of images being visited is insensitive.

4.3. Cluster Merging with Local Descriptors

The two steps above can generate seed clusters with high accuracy, but near duplicates which have large variances will inevitably scatter across multiple buckets due to the lack of semantics in hashing. Therefore, we perform cluster merging based on local descriptors [26] to improve the recall.

We assume that near duplicate images are more likely to reside in close buckets in the global feature space. Therefore, we only merge seed clusters locally in nearby buckets. This ensures the feasibility of cluster merging on the 2B image set, since to grow clusters based on all the seed clusters without sacrifice recall seems impractical [1].

A natural selection of neighbor buckets are whose images are visually close (since semantics are not available). Such a strategy is easy to implement in our approach, since the image signatures are good indications of the visual closeness of two buckets - duplicate images are more likely to appear in the buckets that have slightly different signatures than those that have large Hamming distances ($Dist_h$). Recall that the signatures are generated by PCA projection in our approach.

[1]We only succeeded the merging by checking across *all* the seed clusters rather than only the neighbor buckets by removing the visual words that are shared by more than 4K images, or else the job is too expensive to be fulfilled. In this case, only 167,714 seed clusters got merged, far less than growing within neighbor buckets shown in Table 4.

Table 2. The number of buckets generated vs. signature lengths

	20-bit sig.	24-bit sig.	32-bit sig.
#buckets	1.0M	14.7M	125.6M

For one seed cluster, we define its similar buckets as those whose corresponding signatures have m bits different from that of the seed cluster. Let n be the length of the signatures, the number of similar buckets is $H = \sum_{i=1}^{m} \binom{n}{m}$. In our implementation, we choose $m = 2$ as a tradeoff between efficiency and performance. Therefore, when $n = 24$, in total $\binom{24}{0} + \binom{24}{1} + \binom{24}{2} = 301$ buckets will be checked per seed cluster in cluster merging.

We randomly select one image from each seed cluster as the representative of the cluster's visual appearance. This is reasonable because the seed clusters generally contain global duplicates or exact duplicates which are close in their visual appearances. Local descriptors are then extracted from each representative image, based on which an image graph is built. That is, whenever two images share one visual word, the edge weight between them will add one. Any two images whose edge weight exceeds a threshold is assumed as a near duplicate image pair [18]. We then merge the corresponding seed clusters of near duplicate images, which gives the final duplicate image clusters.

5. Evaluations and Observations

In this section, we present our observations on duplicate image discovery on 2B images. They may expose the nature of image population on the Web.

5.1. Statistics on space partitioning and clustering

Table 2 shows the effect of signature length on the number of hash buckets produced. It can be seen that the numbers (especially with long signatures) are far less than 2^n where n is the signature length. This suggests that real web images are not uniformly distributed but form certain "clouds". Therefore, to do image clustering is meaningful on real web data.

The numbers of seed clusters corresponding to different signature lengths are given in Table 3. Using 24-bit signatures as an example, comparing Table 2 and Table 3, it can be seen that:

- There are 160.0M seed clusters generated from 14.7M buckets, i.e. on average 11 image clusters are formed in one bucket. Though this number may be reduced in the cluster merging step, it still suggests that simply using a non-semantic hashing method on a billion-scale

Table 3. The number of seed clusters and their coverage on images

Cluster Size	20-bit Signature		24-bit Signature		32-bit Signature		Exact match	
	#cluster	image coverage	#cluster	image coverage	#cluster	image coverage	#cluster	image coverage
$\geqslant 2$	161.8M	29.6% | 569.2M	160.0M	28.7% | 553.8M	153.9M	27.0% |521.0M	131.6M	21.7% | 417.7M
$\geqslant 10$	3.8M	7.8% | 149.4M	3.7M	7.2% | 139.4M	3.2M	6.5% |124.6M	1.8M	4.5% | 85.8M
$\geqslant 100$	83.7K	3.8% | 73.0M	78.1K	3.5% | 66.7M	64.0K	3.3% |62.9M	3.7K	2.7% | 52.3M

image set cannot guarantee the accuracy of duplicate images clusters.

- As the signature length increases, so does the number of image buckets, but that of seed clusters decreases. This suggests that larger signature length will cause severe boundary issue, i.e. more duplicate images are scattered in multiple buckets.

Table 3 depicts the number of seed clusters we discovered from the 2B images with various signature length settings, how many images they contain, and the corresponding coverage rate. With 24-bit signatures, we can see that

- 28.7% images have at least one duplicate, which is over 550M images on a 2B image set (and even more on the real Web). This is a very large population. Consider that most of these images are user-interested, such as celebrities, products, events, and funny pictures, to understand the semantics of all these images by image annotation [24] tends to have a positive effect on image search.
- The "exact match" column shows the evaluation results on discovering *exact duplicate* images, i.e. images that have no differences in any aspects. We assume two images to be exact duplicates if their first 2M pixels are identical (2M is a loose bound which covers most of the images on the Web). It can be seen that about $(553.8M - 417.7M)/553.8M = 24.6\%$ images of the seed image clusters are not exact duplicates, and for popular images which have more than 10 duplicates, $(139.4M - 85.8M)/139.4M = 38.5\%$ images are not identical. This suggests the effectiveness of our approach in identifying global duplicate images.
- The number of seed clusters reduces sharply from 160.0M to 3.7M when the cluster size increases from two to ten, and only $78.1K/160.0M = 0.5\%$ seed clusters contain more than 100 duplicate images. This suggests that very popular images occupy a small population of the web images.

5.2. Statistics on cluster merging

Evaluations in this section are based on seed clusters generated with 24-bit signatures. To determine the similar buckets, we evaluated the effect of different n (i.e. the signature length used for identify similar buckets) on clustering performance. Specifically, We tested $n = 16, 20, 24$, which corresponds to 137, 211, and 301 similar buckets respectively.

Table 4. The number of clusters after growing

	$n = 16$	$n = 20$	$n = 24$
#final clusters	144.31M	145.31M	146.37M
#seeds merging	21.88M	21.32M	20.48M
#after merge	6.19M	6.64M	6.85M

Table 4 gives the number of final clusters (#final clusters) with different n, its corresponding number of seed clusters that really got merged (#seeds merging), and the number of merged clusters based on the seeds attending merging (#after merge). Comparing #final clusters and #clusters in Table 3, it suggests that most of seed clusters were not merged. This is partly due to the $m = 2$ setting of close buckets (see Section 4.3) which is a tradeoff between computational feasibility and recall. On the other hand, for those seed clusters that did get merged by local descriptors, the merge rate is very high, e.g. on average $21.32M/6.64M = 3.21$ seed clusters merged into one for $n = 24$.

The distribution of the number of clusters against cluster sizes before and after cluster merging with $n = 24$ is shown in Figure 7. On average clusters become larger after merging and both the curves meet the power law.

5.3. Precision and recall

We randomly selected 2.5K image clusters (after merging) to evaluate the precision and recall. Since small clusters dominate the population (See Figure 7), to avoid biased sampling, we sample 625 clusters from those whose size M ranges in $M \geq 2, 10, 100, 1K$ respectively. We then randomly sample one image from each selected cluster as a

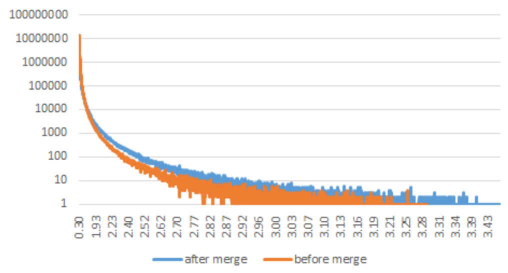

Figure 7. The histogram on the number of clusters vs. cluster sizes meets the power law. Note the logarithmic scales of axes.

Table 5. The average precision and recall performances

	$n = 16$	$n = 20$	$n = 24$	BOW
Avg.Prec.	94.38%	94.41%	94.37%	79.14%
Avg.Recall	71.47%	71.46%	71.42%	92.27%

query to perform bag-of-visual-word (BOW) search on the 2B image set. Since exhaustive search is adopted, it is reasonable to assume that this method gives the upper bound of recall over existing duplicate discovery algorithms [2, 11].

Since no ground truths are available on such a large-scale image set, we apply RANSAC [5] onto the merged set of our clusters and the BOW search results, and assume that RANSAC discovers the ground truth duplicate images[2]. The precision and recall are given in Table 5. It shows that our method exceeds BOW in precision while there is still a gap in recall.

Figure 8 shows four examples of noisy clusters given by the BOW search method. It suggests why BOW achieved so low precision (79.14%) in the evaluation. Local descriptors are not good at such images which unfortunately occupy a large population on the Web and are major sources of false positives for duplicate discovery on real web images.

Figure 9 shows five examples of near duplicate images detected; they cannot be discovered in the first two steps of our approach, which are purely based on global descriptors.

5.4. Observations on duplicate images

Figure 10 illustrates images sampled from clusters of different sizes. Interestingly, images having more than 1K duplicates tend to be less interesting or meaningful for web users than those having fewer duplicates, especially those having more than 10K duplicates - most of them are machine-generated icons or ads. To discover and remove such images from a search engine's index can greatly improve index quality. Moreover, the most web user-interested images are very likely those having 10-1K duplicates, which include celebrities, movie posters, paintings, etc.

[2]In fact, RANSAC is quite strict in judging duplicates. We observed many false negatives. Therefore, the real precision and recall performance should be better than reported here.

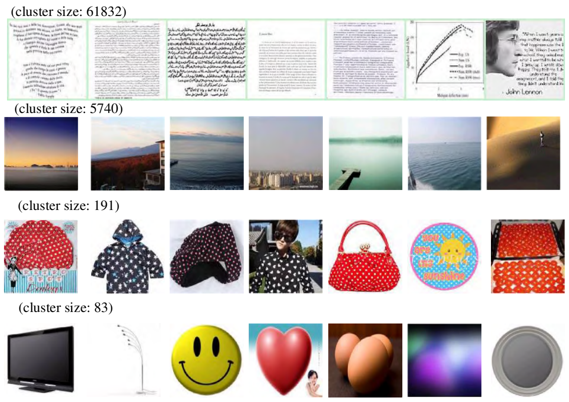

Figure 8. Four examples of noisy clusters obtained by BOW search. Local image descriptors are vulnerable to such images while such images have a large population on the web.

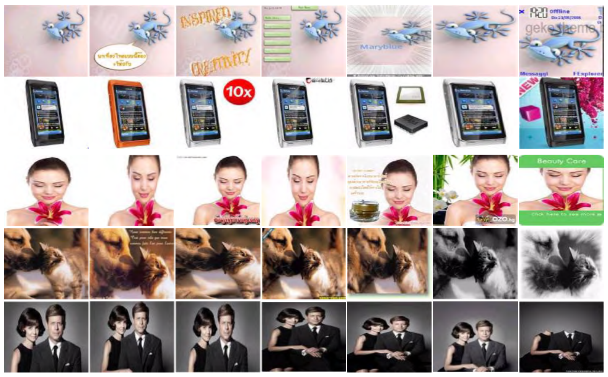

Figure 9. Five examples of near duplicates by our approach.

6. Conclusions

To discover duplicates from billions of images and annotate these images has great value for image search and indexing. In this paper, we carefully studied the advantages and disadvantages of global and local image descriptors on large-scale duplicate discovery, and proposed to use global features to discover seed clusters with high accuracy, and then leverage local descriptors to merge the clusters which can identify near duplicates for high recall. With this approach, we discovered that about 28.7% images, i.e. 553.8M, have duplicates out of 2B images with average precision of 94.41% and average recall of 71.46%.

The proposed approach is efficient. By implementing it on a computer cluster with 2,000 CPU cores, the approach was completed in about 13 hours. Moreover, our approach is promising to scale to even larger datasets, e.g. trillions of images, by deducing weights of seed clusters measured by their sizes since images of higher user interests tend to have a larger number of global duplicates.

References

[1] A. Z. Broder. On the resemblance and containment of documents, 1997. In: SEQUENCES. 2

[2] O. Chum and J. Matas. Large scale discovery of spatilly related images. *IEEE T-PAMI*, 2010. 1, 2, 3, 7

[3] O. Chum, M. Perdoch, and J. Matas. Geometric min-hashing: Finding a (thick) needle in a haystack, 2009. In: CVPR. 1, 2

[4] J. Dean and S. Ghemawat. Mapreduce: Simplified data processing on large clusters, 2004. In: OSDI. 2

[5] M. Fischler and R. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *In: Communications of the ACM*, 24(6):381–395, 1981. 7

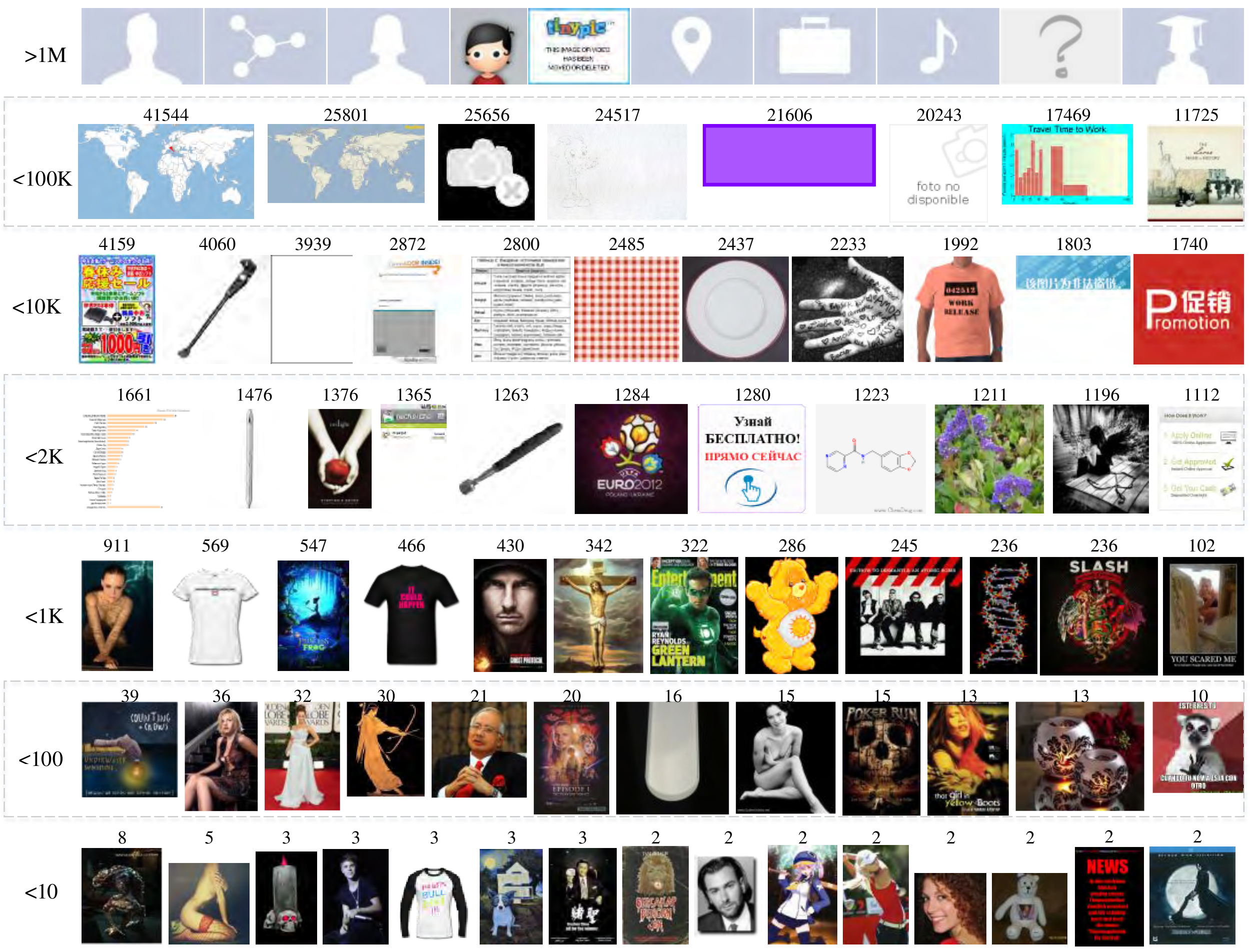

Figure 10. Examples of duplicate images from clusters of different sizes. Images having less then 1K duplicates tend to be more interesting than those having more than 1K duplicates. Most of the images having more than 2K duplicates are advertisements, products, or funny pictures. Images having more than 10K duplicates are much less likely web users' interests.

[6] P. Jain, B. Kulis, I. Dhillon, and K. Grauman. Online metric learning and fast similarity search, 2008. In:NIPS. 1

[7] H. Jegou, M. Douze, and C. Schmid. Hamming embedding and weak geometric consistency for large scale image search, 2008. In: ECCV. 1, 2

[8] Q. Le, R. Monga, and *et al*. Building high-level features using large scale unsupervised learning, 2012. In: ICML. 1

[9] D. Lee, Q. Ke, and M. Isard. Partition min-hash for partial duplicate image discovery, 2010. In: ECCV. 1, 2, 3

[10] Z. Li, X. Xie, L. Zhang, and W.-Y. Ma. Searching one billion web images by content: Challenges and opportunities, 2007. In:MCAM. 1

[11] T. Liu, C. Rosenberg, and H. A. Rowley. Clustering billions of images with large scale nearest neighbor search, 2007. In: IEEE Workshop on Applications of Computer Vision. 1, 2, 5, 7

[12] D. G. Lowe. Object recognition from local scale-invariant features, 1999. In:ICCV. 2, 3

[13] A. Makadia, V. Pavlovic, and S. Kumar. Baselines for image annotation. *IJCV*, 2010. 1

[14] A. Oliva and A. Torralba. Modeling the shape of the scene: a holistic representation of the spatial envelope. *In: IJCV*, 42:145–175, 2001. 2

[15] M. Rubinstein, C. Liu, and W. Freeman. Annotation propagation in large image databases via dense image correspondence. In *ECCV*, 2012. 1

[16] R. R. Salakhutdinov and G. E. Hinton. Learning a nonlinear embedding by preserving class neighbourhood structure, 2007. In: AISTATS. 2, 4

[17] J. Sivic and A. Zisserman. Video google: A text retrieval approach to object matching in videos, 2003. In Proc. ICCV. 2

[18] J. Sivic and A. Zisserman. Video data mining using configurations of viewpoint invariant regions, 2004. In: CVPR. 1, 5

[19] A. Torralba, R. Fergus, and W. T. Freeman. 80 million tiny images: a large dataset for non-parametric object and scene recognition. *In: IEEE T-PAMI*, 30(11):1958–1970, 2008. 2

[20] A. Torralba, R. Fergus, and Y. Weiss. Small codes and large image databases for recognition, 2008. In: CVPR. 2

[21] B. Wang, Z. Li, M. Li, and W.-Y. Ma. Large-scale duplicate detection for web image search, 2006. In: ICME. 3

[22] J. Wang, S. Kumar, and S.-F. Chang. Semi-supervised hashing for large scale search. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 2012. 4

[23] X.-J. Wang, L. Zhang, F. Jing, and W.-Y. Ma. Annosearch: Image auto-annotation by search, 2006. In: CVPR. 1

[24] X.-J. Wang, L. Zhang, M. Liu, Y. Li, and W.-Y. Ma. Arista - image search to annotation on billions of web photos, 2010. In: CVPR. 1, 6

[25] Y. Weiss, A. Torralba, and R. Fergus. Spectral hashing, 2012. In: ECCV. 4

[26] S. Winder and M. Brown. Learning local image descriptors, 2007. In: CVPR. 3, 5

[27] L. Zhang, L. Chen, F. Jing, D. Deng, and W.-Y. Ma. Enjoyphoto: A vertical image search engine for enjoying high-quality photos, 2006. In: ACM MM. 3