Finding Groups of Duplicate Images in Very Large Datasets

Winn Voravuthikunchai
winn.voravuthikunchai@unicaen.fr
Bruno Crémilleux
bruno.cremilleux@unicaen.fr
Frédéric Jurie
frederic.jurie@unicaen.fr

GREYC — CNRS UMR 6072,
University of Caen Basse-Normandie,
Caen, France

Abstract

This paper addresses the problem of detecting groups of duplicates in large-scale unstructured image datasets such as the Internet. Leveraging the recent progress in data mining, we propose an efficient approach based on the search of *closed patterns*. Moreover, we present a novel way to encode the bag-of-words image representation into data mining transactions. We validate our approach on a new dataset of one million Internet images obtained with random searches on Google image search. Using the proposed method, we find more than 80 thousands groups of duplicates among the one million images in less than three minutes while using only 150 Megabytes of memory. Unlike other existing approaches, our method can scale gracefully to larger datasets as it has linear time and space (memory) complexities. Furthermore, the approach does not need (to build or use) any precomputed indexing structure.

1 Introduction

Querying 'Paris' on an image search engine such as Google image search returns more than two billion links to image files spread over the Internet. A quick glance at the first page of results reveals quite a few similar images of Eiffel tower. This simple observation suggests two conclusions (a) the number of images on the Internet is unimaginable and (b) in terms of true content, there is potentially a high amount of redundancy. Such redundancies are natural on the Internet as different entities (e.g. news websites, website designers) might obtain their original content from the same source (e.g. Reuters, commercial image galleries, respectively), slightly modify and reuse them. Even the same entity (e.g. people) might upload the same images, or slightly modified versions, on different sites (Flickr, Facebook etc.) simultaneously. In the present paper, our interest is thus in two related questions: how high is this redundancy and, how to identify it given the very high search space of all the images on the Internet. We are interested here in 'duplicate' images which come from the same source but have been slightly modified with different types of changes such as those resulting from compression, scaling, small crops, insertion/substitution of small regions, changing brightness/contrast. The problem of discovering duplicates has important uses: it can be used in many applications such as (a) image databases to improve space efficiency by keeping only one instance per duplicate set, (b) personal photo collection management

http://dx.doi.org/10.5244/C.26.105

by eliminating duplicates, (c) image search by eliminating duplicate results and finally (d) applications related to image copyright enforcement.

Please note that finding groups of duplicates is very different from Content-Based Image Retrieval (CBIR), which has received a lot of attention in the recent computer vision literature. CBIR methods assume that a query image is input to the system, which in turn returns the similar/duplicates from the indexed dataset. Despite recent advances in image representation and indexing techniques as which have allowed CBIR methods to be able to scan through millions of images and return the results in milliseconds, these still cannot be used to find groups of duplicates as each image of the dataset would have to be considered in turn as a query, and all the results would have to be merged. Similarly, traditional clustering algorithms could be argued to be applicable in this case. While in theory they could be used to detect duplicate groups, in practice they are not scalable to large datasets without resorting to coarse approximation. Furthermore, they would be inefficient as it is undesirable to cluster the whole set of images but only to discover groups of duplicates. As far as we know, this problem of discovering groups of near-duplicate images in very large datasets has been only marginally addressed in the computer vision literature.

When the task requires processing very large datasets, it is natural to think about techniques provided by research on data mining. Indeed, algorithms in the field of data mining are designed for extracting information such as itemsets from extremely large datasets. In particular, we show in this paper how the discovery of groups of duplicate images is related to the discovery of closed itemsets and can strongly benefit from advanced data mining techniques in this area.

We make three contributions: (a) we establish the link between mining closed itemsets and the discovery of groups of duplicate images, (b) propose a novel image representation built in terms of data mining transaction, allowing us to make use of efficient data mining algorithms, and (c) provide a new dataset consisting of one million images to experiment on the problem of detecting groups of duplicates on large-scale image datasets.

The paper is organized as follows: Section 2 discusses previous related works while Section 3 describes our approach by explaining how to encode images as data mining transactions and explain how to obtain the groups of duplicates. Section 4 provides the experimental validation of our ideas: we first validate our image representation in an image search scenario, then evaluate the quality of the group detection as well as the complexity of our approach in terms of efficiency and memory usage in large scale experiments. Finally, Section 5 concludes the paper.

2 Related Work

As explained in the introduction, finding groups of similar images can be seen as a clustering problem. As an illustration, [2] proposed a hierarchical spectral clustering method using visual, textual and link analysis for organizing the results of image search into different semantic clusters. In the same way, [13] automatically generated representative and diverse views of the world's landmarks using a combination of context and content-based tools to generate representative sets of images for location-driven features and landmarks. Clustering has also been used to re-rank image search results [15], assuming relevant images belonging to large clusters. Closer to our work, [4] used a clustering algorithm to find near duplicate images. However, as these previous works use standard clustering algorithms which do not scale well with the number of images, they cannot be used for clustering large datasets consisting of several millions of images.

On the other hand, large-scale clustering has been also studied in the recent literature, mostly by considering clusters as dense regions and using some heuristics, usually relying on user-defined density thresholds or sub-sampling strategies to identify dense and non-dense regions [9]. Unfortunately, they cannot be used in our case as groups of duplicates are not dense e.g. two duplicates can form a group. Furthermore, we are not interested in clustering the whole set, which would waste a lot of time, but just finding the duplicates.

Most of the existing methods for duplicate detections are based on image search, which has received a lot of attention in the recent computer vision literature[5, 6, 12, 20]. However, as explained in the introduction, using image search techniques to discover groups of duplicates would require to use each image in turn as a query and to eventually merge the search results, which would be very computationally expensive.

Two of the most related works to ours are [8] and [27]. In [27], image features are first projected to a lower dimensional space using PCA. The new features are transformed to a 32-bit binary string used as hash codes for the images. They detect groups of duplicates by grouping similar hash codes together. Due to the property of PCA, similarity is better expressed by a few significant bits. For this reason, hash codes are grouped if the L most significant bits are identical and the Hamming distance of remaining bits is below a certain threshold. Experiments showed that their approach is fast for a dataset of up to 100,000 images, but the time complexity grows exponentially with the number of images. On the other hand, [8] addressed the problem by using an inverted list and a hash table. Each entry of the inverted list consists of a list of images containing the same LSH index. All posible pairs of images are found in each list and are used to increment the value of the image pair index in the hash table. The pairs are seen as duplicates if their hash pair count is larger than a threshold. In order to merge the duplicate pairs to form groups of duplicates, a graph connecting the duplicate pairs is built. Duplicate groups are obtained by separating the groups which are not linked together. Clearly the complexity grows quadratically during the process in hashing image pairs.

Because of its ability to deal with very large amounts of data, data mining has been used for addressing several computer vision tasks. In [28], a mining process identifies salient terms from textual descriptions of search results. In [29], co-occurrence patterns are used in a classification framework allowing to select groups of features that can best discriminate between two classes. We can also mention [21], in which frequent itemsets are used to automatically find spatial configurations of local features occurring frequently on instances of a given object class, and rarely on the background. Even if there are these few works investigating relationships between data mining and computer vision, there is not yet any work dealing with data mining methods for the detection of groups of duplicate images.

In conclusion, as far as we know, there is no method that can discover groups of duplicate images while at the same time scaling linearly in the number of images, and therefore able to handle very large datasets. This is precisely the core contribution of this paper.

3 Method

The key idea of our approach is to model images as sets of binary visual attributes and to define groups of duplicates as groups of images sharing large enough amounts of attributes.

Let's consider a very simple example illustrated by Figure 1. In this example we have 5 different images denoted $I_1, \ldots, I_5$ and 9 different visual attributes that an image can have or not, denoted $a_1, \ldots, a_9$. If images are considered as near duplicates when they share 4 or more attributes, then this example contains only one group of near duplicates, i.e. the group

Database	
I_1	$\{a_1, a_2, a_3, a_5, a_8\}$
I_2	$\{a_1, a_2, a_3, a_5, a_6\}$
I_3	$\{a_1, a_7, a_8, a_9\}$
I_4	$\{a_1, a_2, a_3, a_5, a_7, a_8\}$
I_5	$\{a_1, a_8\}$

(a) Transaction database

FrequentItemset	*freq*	*L*	*support*
$\{a_1\}$	5	1	I_1, I_2, I_3, I_4, I_5
$\{a_2\}$	3	1	I_1, I_2, I_4
$\{a_8\}$	4	1	$I_1, I_3, I_4\ I_5$
$\{a_1, a_2, a_3, a_5\}$	3	4	I_1, I_2, I_4
$\{a_1, a_8\}$	4	2	$I_1, I_3, I_4\ , I_5$

(b) A subset of the frequent itemsets with $minfr = 3$

Figure 1: Example of frequent itemsets and the notations. I: images, a: visual attributes, $freq$: number of occurrences of an itemset in the database, L: length of an itemset, $support$: list of transactions containing an itemset.

made of the images I_1, I_2, I_4 in which they share the 4 attributes a_1, a_2, a_3 and a_5.

In line with data mining terminology, we refer to the visual attributes as *items*, and groups of items as *itemsets*. Images are referred to as *transactions* and each image is defined by one itemset. Data Mining aims at leveraging the available data and extracting knowledge from it. A typical output of data mining algorithms are itemsets having "interesting" properties [1]. Two interesting properties of itemsets are their *frequencies* (denoted *freq*) and their *lengths* (denoted L). The frequency of an itemset X is the number of its occurrences in the database. For instance, in Figure 1, $freq(\{a_1,a_8\}) = 4$ since $\{a_1,a_8\}$ occurs in the transactions I_1, I_3, I_4, and I_5. The set of transactions containing X is called the *support* of X. An itemset X is said *frequent*, if its frequency exceeds a given minimal threshold $minfr$: $freq(X) \geq minfr$. By setting $minfr$, data analysts can fix the number of returned itemsets, lower frequency thresholds resulting in larger number of frequent itemsets. The *length* of an itemset is the number of items it contains. Among the different ways to discover "interesting" itemsets [16], we focus here on *closed itemsets*. Closed itemsets can be seen as maximal rectangles of ones in a binary matrix. If X is a closed itemset, it is impossible to add any item to X without decreasing its frequency. In Figure 1, both $\{a_2\}$ and $\{a_1,a_2,a_3,a_5\}$ are itemsets of frequency 3 but only $\{a_1,a_2,a_3,a_5\}$ is closed because adding any item will decrease its frequency. Unlike $\{a_2\}$ in which its frequency still remains at 3 when items such as a_1, a_3, or a_5 are added.

Closed itemsets represent the maximum amount of similarity among sets of transactions. Our goal is to discover all closed itemsets that are long enough (i.e. that include sufficient common visual properties) and that have a frequency greater than 2 (i.e. this long list of visual attributes is occurring in more than two images). The set of transactions (i.e. the group of images) containing these itemsets will be considered as groups of duplicates.

Stating the problem this way, it becomes necessary to explain (a) how to represent images as itemsets of visual attributes and (b) how to mine out long closed itemsets whose frequency is greater than 2. These two steps are described in the two following sections.

3.1 Coding image and transactions of visual properties

As explained before, the purpose of this first stage is to represent images by sets of visual attributes (i.e. items). Our representation is built on the recent and powerful bag-of-visual-words (BoW) [11, 23, 26] but, in contrast, our representation has to be a binary representation. Indeed, the presence or absence of an item in a transaction (an image in our case) is binary information. To overcome this difficulty, we represent images by lists of their most informative visual words, using the *tf-idf* weighting (term frequency-inverse document frequency). Tf-idf has been introduced to give higher weights to more important words in text documents [22], and has been also successful for normalizing BoW in vision tasks as

well [6, 17, 23]. Following this line of work, we represent images by the list of their top K tf-idf weighted visual words.

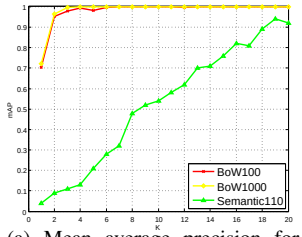
Compactness is important, as we will have to keep the database into the main memory. In terms of memory usage, since each item in a transaction can be any of the possible visual words, storing a transaction requires only $K \times log_2(D)$ bits, where D is the size of the visual vocabulary. Regarding the choice of K, we would like K to be as small as possible. As shown later (Section 3.2), the mining complexity grows exponentially in the number of visual words per itemset. In practice, K has been determined experimentally (see Section 4.2); we show that the performance for retrieving near exact duplicates starts to saturate when $K \geq 6$. However, we choose $K = 10$ in order to be able to detect duplicates in which the original image has been altered by harder attacks. We believe $K = 10$ is a good trade-off between quality and efficiency.

Representing images by only 10 visual words results in high loss of information, but, as shown in our experiments, the remaining information is still sufficient for finding near duplicates. In addition, this representation is tolerant to additional noise, as the top weighted tf-idf visual words are among the most frequent ones, they are very stable. Finally, this representation is robust to transformations such as JPEG compression, scaling, rotation, slight crops and illumination changes, as show in the experiments section.

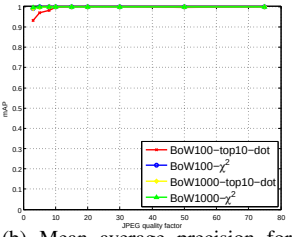
3.2 Mining groups of similar images

After representing images as transactions of items, we aim at extracting *all* frequent closed itemsets whose length is greater than a given threshold (denoted *minlength*) and whose frequency is greater than *minfr* = 2. This is a challenging problem because of the size of the search space. For m items, the search space is made of 2^m possible itemsets. Data mining algorithms have developed safe pruning strategies to cope with this difficulty. It is easy to observe that if an itemset is infrequent all its supersets are also infrequent. This property leads to one of the most crucial pruning strategies. Itemset mining can also be improved using *redundancy*. An itemset is redundant if it can be derived from the other itemsets found. Itemset condensed representations [3] restrict the mining to specific itemsets like the free or the closed patterns [19]. These itemsets partition the search space into equivalence classes, free itemsets being minimal elements (w.r.t. itemset inclusion), closed itemsets being maximal elements. Interestingly, mining either free itemsets or closed itemsets is enough to infer the frequency of any pattern, and allows the use of specific pruning strategies [3]. Therefore, mining frequent closed itemsets using anti-monotonicity and specificities of the closure operator is much more efficient than mining all frequent itemsets [19].

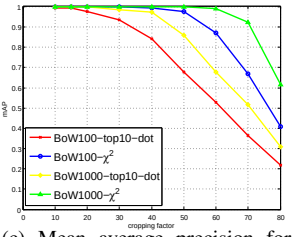
Our mining strategy is based on LCM [25], which is one of the most efficient algorithms for mining frequent closed itemsets. As we are looking for groups of duplicates, each itemset has to be supported by at least two images ($minfr = 2$). Even if this value seems low and may lead to a large number of itemsets, it significantly reduces the search space. Furthermore, as the search of closed itemsets is very efficient, it can be used on very large databases (see Section 4). The longer the itemset is, the more similar the images containing the itemset are. Therefore, introducing length constraints when searching for closed patterns would be possible and might be done more efficiently, by transposing the representation [18]. However, this technique is limited to datasets with a small number of transactions because the mining complexity grows exponentially according to the number of items [10]. With this technique, the number of items would be the number of images after transposing the representation, and since we are working with a huge number of images we cannot use this. On the other hand, LCM is linear in time for the number of frequent closed itemsets [25].



(a) Mean average precision for JPEG75 attacks, as a function of K (number of attributes per image)



(b) Mean average precision for JPEG attacks from JPEG3 (very low quality) to JPEG75 (typical web quality).



(c) Mean average precision for cropping attacks from 10% to 80% of the image surface.

Figure 2: Image retrieval experiments: performance of the proposed representation and of the baseline representation, for two vocabularies (100 and 1,000 visual words).

4 Experiments

This section presents the experimental validation of the proposed approach. We first describe the datasets used for the experiments, validate the proposed binary representation by comparing it with a baseline representation, and, finally, give quantitative and qualitative results for large-scale groups of duplicates detection.

4.1 Datasets

The **One million random web images database** is one of the contributions of this paper. Our motivation for creating this dataset is the need of having a representative sample of the set of images available on the Internet. Indeed, publicly available image databases, which are often made by using limited numbers of specific text queries, contain images that lie in a few local "regions" of the whole Internet image set. Besides, their numbers of duplicate groups have a bias according to the queries. Images from queries such as "logos", will have more duplicates than images from more generic queries such as "animals".

Unlike other datasets, we used ~100,000 random alphabet strings as query inputs to Google image search engine. The images were downloaded and split into 1,000 batches of 1,000 images each, in order to ease the access to the dataset. 1,000 text files are associated to each batch, containing the names and the source links of the images. The images so obtained are very diverse, including man-made objects, sceneries, logos, sketches, animals, etc. This dataset can be downloaded from authors' web pages.

The **Copydays dataset** was proposed in [7] for evaluating the robustness of image descriptors against artificial image transformations in an image search scenario. The dataset contains 157 original images and their "copies", which have suffered three types of artificial attacks (JPEG, cropping and "strong"), referred as "attacked" images. Each attacked image is obtained by transforming one original image with one attack. As said before, attacks are of three types: (i) image scaling by a factor of 16 in surface, followed by 9 JPEG compressions ranging from the very low quality JPEG3 to the typical web quality JPEG75, (ii) 9 cropping factors in a range of 10% to 80% of the image surface, and (iii) various strong attacks such as print and scan, paint, blur, very strong crop, etc. As we are interested in finding similar images or near duplicates, we neglected the strong attacks and kept only 18 attacks (9 JPEG+9 cropping factors) of the two first types.

Each of the 18 attack sets used consists of exactly 157 images, i.e. each original image has been transformed only once for each attack. Together with the original image set, there are 2,983 images in total.

4.2 Representing images as transactions of itemsets

We have introduced (Section 3) a new binary image representation using the top K tf-idf visual words for the image. We first validate this representation by comparing it with a "standard" image representation. We do it in an image search scenario, on the **Copydays dataset**.

The evaluation protocol is as follows: we rank the attacked images according to their distances from the original images (which are used as query images). We compute the average precision for each original image and report the mean average precision. 10,000 distractor images randomly sampled from the **One million random web images database** are added to the **Copydays dataset**, to make the task harder.

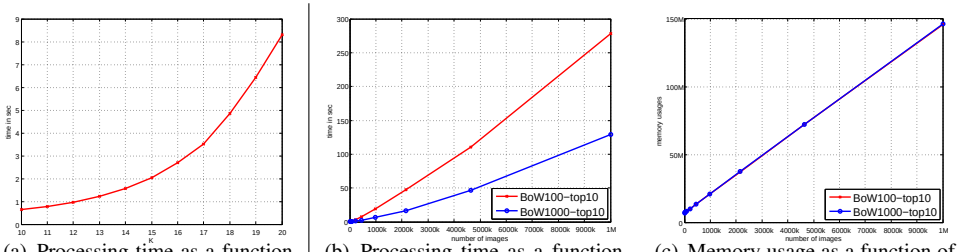
The "standard" representation is that of representing images by L1 normalized BoW histograms. The similarity between two images is given by the χ^2 distance between their representations. Regarding the representation by itemsets, the items representing an image are the top K visual words, after tf-idf weighting (see Section 3). Itemsets representations for images are given by binary vectors in which "1" (respectively "0") means that the corresponding visual word is (respectively is not) one of the top K visual words for that image. Similarity between binary vectors is computed with the dot product which is equivalent to that used in the data mining process i.e. the similarity between two transactions is the number of common items.

The representation by itemsets depends on the size of the vocabulary, and it is therefore important to know if smaller or larger vocabularies are better suited. We suspect that very large vocabularies would produce highly specialized representations. Hence, we consider two vocabularies with 100 and 1,000 visual words respectively. We also suspect that visual words might become very local and lack of semantic meanings. Hence, we built another binary representation based on semantic features. In this case, we compute 110 attributes according to [24] and represent image by the K semantic attributes having the highest scores.

We first present some experiments done on with the JPEG75 attack (this attack represents typical duplicates we can find on the Internet). Three conclusions can be drawn from these experiments, whose results are given in Figure 2a. First, representing images by their top visual words performs better than representing them by their top semantic concepts. One explanation is that two images can contain exactly the same semantic concepts while being visually very different e.g. two images representing a 'plane' can be different. Second, it can be seen that $K \geq 6$ gives optimal performance for the binarized BoW representation. For further experiments, we take $K = 10$ to handle even stronger attacks than JPEG75. Third, the size of the vocabulary is not crucial in this case, and both vocabularies are equivalent in terms of performance.

The second set of experiments aims at comparing the BoW based binary representation (with $K = 10$ and dot product similarity) to a standard BoW representation (with χ^2 distance). We do the comparison by using the JPEG attacks (from JPEG3 to JPEG75). The results are given in Figure 2b. Two conclusions can be drawn. First, using the larger vocabulary gives better results for both the binary representation and the BoW representation. Second, when using the binary representation, the performance of the larger vocabulary is better for strongest compressions. We also did some experiments using cropping attacks (from 10% to 80%), and the performance is given in Figure 2c. Larger vocabulary performs better again, and the binary representation is almost as good as the BoW one, for cropping factors below 30%.

In conclusion, these experiments demonstrate that this representation is sufficient for



(a) Processing time as a function of K (for 10,000 images)

(b) Processing time as a function of the size of the database

(c) Memory usage as a function of the size of the database.

Figure 3: Duplicate detection: quantitative results.

detecting near duplicate images, while being very compact (each image is encoded by ~13 bytes only). Furthermore, as this representation is made of lists of items, it can be used efficiently for finding frequent closed patterns.

4.3 Duplicate detection

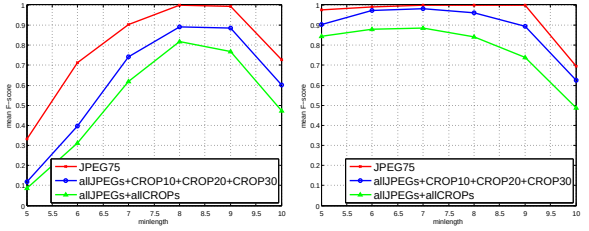
After showing in the previous section that our new representation can be used efficiently for comparing images, this section experimentally validates the mining algorithm proposed for discovering groups of duplicate images.

Quantitative results. Quantitative results can be obtained only by using datasets for which we have ground truth. Here again, we use the **Copydays dataset**, but in a different way: in these experiments, we put together the 157 original images and corresponding attacked images. We performed three different experiments: i) in the first experiment, we use only pairs made of one original image and its JPEG75 compression, resulting in 157 groups of 2 images ii) in the second experiment, we use all of the 9 JPEG attacks and the 3 lightest cropping attacks, resulting in 157 groups of 13 images (the original plus 12 duplicates) and iii) in the third experiment, the 9 JPEG and 9 cropping attacks are used, giving 157 groups of 19 images. To make it more difficult, we increase the size of the dataset by adding 1,000,000 artificial image descriptors (generated by producing random lists of transactions). In the ideal case, the algorithm should correctly discover the 157 groups of duplicates.

The performance is evaluated by using the mean F-score, as introduced by [14]. The mean F-score is equal to one if and only if the system outputs exactly 157 groups containing the original image and its transformations only.

An important parameter of the algorithm is the *minlength* (defined in Section 3.2). This is actually the number of attributes two images have to share for being considered to be duplicates. Figure 4 shows the F-Score as a function of *minlength*, for representations made from 100 and 1,000 visual words dictionaries. $minlength = 7$ and 1,000 visual words dictionary give optimal results. We can see that for the light attacks, the groups of images are perfectly detected. Even for the strongest attacks the results are still very good.

In addition, we have also evaluated how the computation time and the memory usage scale with the size of the dataset. We use the **One million random web images database**, from which we sample between 1,000 and 1,000,000 images. Figure 3b and Figure 3c show that both the computation time as well as the memory usage are linear in the number of images of the dataset, as expected from the theoretical analysis. In addition, Figure 3a shows that the computation time grows exponentially with the number of items K in each transaction. Finally, we can observe that processing 1 million images takes less than three minutes using a single core of 2.2 GHz processor. The state of the art in image search [12] with its fastest setting, requires 1.5 milliseconds per query for searching in 1,000,000 images. This



(a) 100 visual words, $K = 10$ (b) 1,000 visual words, $K = 10$

Figure 4: Mean F-score as a function of *minlength*

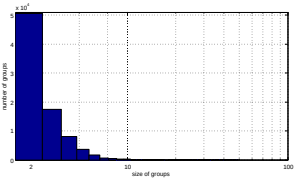


Figure 5: Size statistics found on the **One million random web images database**

Figure 6: Some of the groups of duplicate/similar images found on the **One million random web images database**

makes approximately 25 minutes to obtain pairs of duplicates, if we take all images in turn as queries. Note that such image search does not produce groups of images, and more computations would be required for detecting groups. In addition, our approach does not need to build any index.

Qualitative results. These experiments aim at estimating the amount of duplicate images present in our **One million random web images database**. By running our algorithm, we obtained more than 80 thousands groups of duplicates in less than 3 minutes. Figure 6 shows some of these groups. Beside computational efficiency, these results demonstrate the robustness against compression, scaling, slight crops, rotation, insertion/removal of small elements, brightness/contrast changes. We also provide some statistics in Figure 5. However, as we do not know how many actual duplicates are within the dataset, we cannot measure the quality of those groups.

5 Conclusions and future work

This paper presented a novel approach for detecting groups of duplicate images in very large databases by encoding images as data mining transactions and mining out long closed

itemsets. To our knowledge, we are the first to propose a method that scales linearly, both in time and memory, to the number of images. Our system requires less than three minutes and around 150 Megabytes for detecting approximately 80 thousands groups of duplicates in a database of 1 million images. This efficient search method has been evaluated on a new dataset, the **One million random web images database**. We believe that this algorithm can scale to the detection of redundant images over thousands of billion images spread over the Internet, a colossal task which has not been addressed before. Our future works include the use of even more efficient pruning strategies, based on minimal similarity constraints between duplicates in order to have an even faster system.

Acknowledgements This work was partially funded by the QUAERO project supported by OSEO, French State agency for innovation.

References

[1] R. Agrawal, H. Mannila, R. Srikant, H. Toivonen, and A. I. Verkamo. *Fast discovery of association rules*, chapter 12. Advances in Knowledge Discovery and Data Mining. AAAI/MIT Press, 1996.

[2] D. Cai, X. He, Z. Li, W.Y. Ma, and J.R. Wen. Hierarchical clustering of www image search results using visual, textual and link information. In *ACM Multimedia*, pages 952–959, 2004.

[3] T. Calders, C. Rigotti, and J-F. Boulicaut. A survey on condensed representations for frequent sets. In *Constraint-Based Mining and Inductive Databases*, volume 3848 of *Lecture Notes in Computer Science*, pages 64–80. Springer, 2005.

[4] E. Chang, C. Li, J. Wang, P. Mork, and G. Wiederhold. Searching near-replicas of images via clustering. In *SPIE Multimedia Storage and Archiving Systems IV*, 1999. URL http://ilpubs.stanford.edu:8090/391/.

[5] E.Y. Chang, J. Ze Wang, C. Li, and G. Wiederhold. Rime: A replicated image detector for the world-wide web. In *Proc. of SPIE symposium of voice, video and data communications*, pages 58–67, 1998.

[6] O. Chum, J. Philbin, and A. Zisserman. Near duplicate image detection: min-hash and tf-idf weighting. In *BMVC*, 2008.

[7] M. Douze, H. Jégou, H. Sandhawalia, L. Amsaleg, and C. Schmid. Evaluation of gist descriptors for web-scale image search. In *CIVR*, pages 19:1–19:8, 2009.

[8] J. Foo, J. Zobel, and R. Sinha. Clustering near-duplicate images in large collections. In *Proceedings of the international workshop on Workshop on multimedia information retrieval*, MIR, pages 21–30. ACM, 2007.

[9] V. Ganti, R. Ramakrishnan, J. Gehrke, A. Powell, and J. French. Clustering large datasets in arbitrary metric spaces. In *Proceedings of the 15th International Conference on Data Engineering*, pages 502–511. IEEE Computer Society, 1999.

[10] F. Geerts, B Goethals, and J. Van den Bussche. A tight upper bound on the number of candidate patterns. In *ICDM*, pages 155–162, San Jose, California, USA, 2001.

[11] K. Grauman and T. Darrell. The pyramid match kernel: Discriminative classification with sets of image features. In *ICCV*, pages 1458–1465, 2005.

[12] H. Jégou, M. Douze, and C. Schmid. Product quantization for nearest neighbor search. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(1):117–128, 2011.

[13] L.S. Kennedy and M. Naaman. Generating diverse and representative image search results for landmarks. In *Proceedings of the 17th international conference on World Wide Web*, WWW, pages 297–306. ACM, 2008.

[14] B. Larsen and C. Aone. Fast and effective text mining using linear-time document clustering. In *KDD*, pages 16–22, 1999.

[15] W. Liu, Y. Jiang, J. Luo, and S. Chang. Noise resistant graph ranking for improved web image search. In *CVPR*, pages 849–856, 2011.

[16] R. T. Ng, V. S. Lakshmanan, J. Han, and A. Pang. Exploratory mining and pruning optimizations of constrained associations rules. In *proceedings of ACM SIGMOD'98*, pages 13–24, 1998.

[17] D. Nistér and H. Stewénius. Scalable recognition with a vocabulary tree. In *CVPR*, pages 2161–2168, 2006.

[18] F. Pan, G. Cong, A. K. H. Tung, Y. Yang, and M. J. Zaki. CARPENTER: finding closed patterns in long biological datasets. In *proceedings of the 9th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD'03)*, pages 637–642, Washington, DC, USA, 2003. ACM Press.

[19] N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal. Discovering frequent closed itemsets for association rules. In *proceedings of 7th International Conference on Database Theory (ICDT'99)*, volume 1540 of *Lecture notes in artificial intelligence*, pages 398–416, Jerusalem, Israel, 1999. Springer Verlag.

[20] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *CVPR*, pages 1–8, 2007.

[21] T. Quack, V. Ferrari, B. Leibe, and L.J. Van Gool. Efficient mining of frequent and distinctive feature configurations. In *ICCV*, pages 1–8, 2007.

[22] G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. In *Information Processing and Management*, pages 513–523, 1988.

[23] J. Sivic and A. Zisserman. Video google: A text retrieval approach to object matching in videos. In *ICCV*, pages 1470–1477, 2003.

[24] Y. Su, M. Allan, and F. Jurie. Improving object classification using semantic attributes. In *BMVC*, 2010.

[25] T. Uno, T. Asai, Y. Uchida, and H. Arimura. An efficient algorithm for enumerating closed patterns in transaction databases. In *proceedings of Discovery Science (DS'04)*, volume 3245 of *LNAI*, pages 16–31, Padova, Italy, 2004. Springer.

[26] C. Wallraven, B. Caputo, and A.B.A. B. A. Graf. Recognition with local features: the kernel recipe. In *ICCV*, pages 257–264, 2003.

[27] B. Wang, Z. Li, M. Li, and W. Ma. Large-scale duplicate detection for web image search. In *ICME*, pages 353–356, 2006.

[28] X.J. Wang, L. Zhang, X.R. Li, and W.Y. Ma. Annotating images by mining image search results. *PAMI*, 30(11):1919–1932, November 2008.

[29] J. Yuan, M. Yang, and Y. Wu. Mining discriminative co-occurrence patterns for visual recognition. In *CVPR*, pages 2777 –2784, 2011.