Dataset Fingerprints: Exploring Image Collections Through Data Mining

Konstantinos Rematas[1], Basura Fernando[1], Frank Dellaert[2], and Tinne Tuytelaars[1]

[1]KU Leuven, ESAT-PSI, iMinds
[2]Georgia Tech

Figure 1: Given an image collection, our system extracts patterns of discriminative mid level features and uses the connection between them to enable structure specific browsing.

Abstract

As the amount of visual data increases, so does the need for summarization tools that can be used to explore large image collections and to quickly get familiar with their content. In this paper, we propose dataset fingerprints*, a new and powerful method based on data mining that extracts meaningful patterns from a set of images. The discovered patterns are compositions of discriminative mid-level features that co-occur in several images. Compared to earlier work, ours stands out because i) it's fully unsupervised, ii) discovered patterns cover large parts of the images,often corresponding to full objects or meaningful parts thereof, and iii) different patterns are connected based on co-occurrence, allowing a user to "browse" the images from one pattern to the next and to group patterns in a semantically meaningful manner.*

1. Introduction

Information is the result of data processing. Said another way, data only becomes information that you can use to make decisions after it has been processed. It is hard to understand data in bulk. Thus, it's best if data gets summarized. In recent years, the amount of visual data has increased tremendously and computer vision is one of the research fields that benefited from this. Datasets used for object recognition, detection etc. have increased not only in size (*e.g.* Pascal VOC[6], ImageNet[3], COCO[15]) but also in the extend of the visual world that they capture. As the amount of visual data increases, there is an increased demand to summarize visual archives, so that stakeholders can be guided to interesting aspects of the data quickly. A nice example is The New York Times fashion week browser, which it shows highlights of the event through thumbnails reminiscent of slit-scans [23]. These so called *fingerprints* are the zoomed out view of pixels and most dominant colors of each piece and designer. It gives a quick sense of what each designer showed off this year. More recently, another data summarization called "Flickr Clock" has been introduced. It arranges videos as *slices of time*. These examples require manual effort or semi-automated processes to summarize visual archives.

The benefit of summarization is that it not only reduces the amount of data that needs to be digested, but it also increases the ability to interpret the data. If there is a small amount of data, this can be done by hand or one can

Project: `http://homes.esat.kuleuven.be/~krematas/DatasetFingerprints/`

take averages[27]. Otherwise, we need algorithms to summarize and analyze the visual data. The traditional way of browsing such image datasets is based on manual labels that annotators used (*e.g*. the "explore" button on the COCO web-page [15] and the "example images" in Pascal VOC [6]) or by finding correspondences between pairs of images [12, 14, 20].

In this paper we propose a novel way of summarizing and browsing an image collection, based on data mining. We use data mining to chart visual data in a way that is meaningful. It is often helpful to use tables or bar charts to summarize numerical data in order to get a clear overview. Similarly, to obtain the *big picture* of visual data we can rely on summarization and traversal strategies such as connected graphs, trees, Hasse diagrams, bipartite graphs or doubly connected lists. Data mining and pattern mining provide a natural way to summarize and traverse data. Our intuition is that there are sets of images that *see* the same underlying structures, visual patterns and constellations. These structures can be anything from a set of small patches to large structured regions that occur in several images with certain degree of spatial consistency.

We express the problem of finding these structures as an *unsupervised* data mining problem: our image collection is the database, the images are the transactions and mid-level features are the items. We then look for *patterns*, *i.e*. combinations of mid-level features that appear in several images. Based on these we construct various visualizations that allow to explore these collections in various ways.

2. Related work

Snavely *et al*. [20] presented a system for browsing large photo collections of popular sites which exploits the common 3D geometry of the underlying scene. Their focus is on the 3D reconstruction, not on discovering mid and/or high level visual structures. Given a large repository of geotagged imagery, Doersch *et al*. [5] seek to automatically find visual patches of windows, balconies, and street signs, that are most distinctive for a certain geo-spatial area. A graph based location recognition method is presented in [1] where nodes correspond to images. Each edge represents the level of overlap, *i.e*., the geometric consistency between image pairs. These methods are good at recognizing distinctive visual elements in a specific location. They are not necessarily useful for summarizing or exploring a large dataset consisting of semantically different visual entities. Other graph building approaches are [13, 12]. They are based on image-to-image affine feature matching.

Our method is also related to unsupervised object discovery methods. A method that uses image affinity based on "similarity by composition" to discover object categories is presented in [8]. Recently, Doersch *et al*. [4] use context as a supervisory signal to rectify the Exemplar LDA patches of Hariharan *et al*. [11] growing the predicted region. In [17] a geometric latent Dirichlet allocation model is presented, for unsupervised discovery of particular objects in image collections. In contrast, we rely on data mining to discover visual composites and the relationships between them.

Our method is also somewhat related to recent weakly supervised methods such as [21, 22]. Both these and our approach are motivated by the idea that clusters of exemplars are good at encoding some shared visual aspect that one may be interested in. However, Song *et al*. [21, 22] pose this as a sub-modular cover problem and use a greedy algorithm to solve it.

Data mining has also been applied to discover mid-level patterns for image classification [9, 26, 18], image retrieval [10] and bundle adjustment [2]. None of these methods use data mining to do visual data summarization or exploration. Interactive data explorations using pattern mining methods have also been studied recently in the data mining literature [25], albeit for non-visual data exploration only.

An *interactive* method that allows a user to explore and visualize a large image collection using simple average images is presented in [27]. They discover the modes of a dataset by interactive explorations. In visual data exploration, the dynamic arrangement of results was also studied in [16]. These methods can benefit from our method to improve the image exploration and summarization capability.

3. Background

In this section we introduce the data mining terminology and notations. In our framework, each image is described by a set of mid-level clusters [19]. In data mining terms, each image becomes a transaction denoted by t and each mid-level cluster becomes an item denoted by i. The set of all transactions (*i.e*. all images) is denoted by T and $t \in T$. The set of all mid-level clusters, also known as set of items, is denoted by I, *i.e*. $i \in I$. We train n clusters resulting in n items, *i.e*. $I = \{i_1, i_2, \cdots i_n\}$. In our image database we have m images resulting in m transactions, *i.e*. $T = \{t_1, t_2, \cdots t_m\}$. Each transaction is a subset of items I, *i.e*. $t \subseteq I$. For example, if there are five mid-level clusters activated in an image, this results in a transaction t of five items, *i.e*. $t = \{i_a, i_b, i_c, i_d, i_e\}$. A pattern p is a combination of items that appears in large number of transactions. A pattern is also a subset of items, *i.e*. $p \subseteq I$. The length of the pattern is the cardinality of p. The support of a pattern p indicates in how many transactions that pattern appears, *i.e*. $|\{t|p \subseteq t; t \in T\}|$. As a pattern is a set of items, it is also known as an itemset.

For example, let us say three mid-level clusters a, b, c appear together in 10 images. As a result items i_a, i_b and i_c would appear in 10 transactions. This would result in a pattern $p = \{i_a, i_b, i_c\}$. In pattern mining terminology

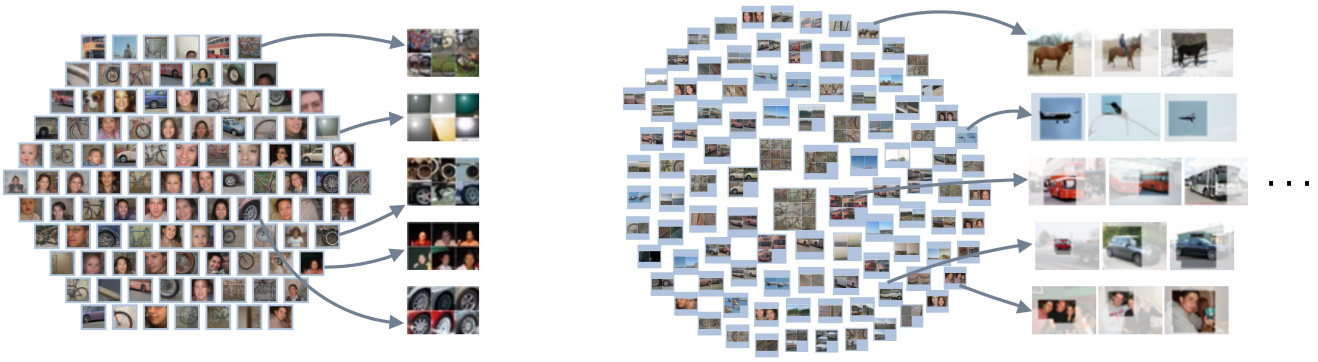

Figure 2: Left: set of Exemplar LDA clusters. Right: Example tiles, with size of bubble representing the area of the tile.

we say the support of pattern $p = \{i_a, i_b, i_c\}$ is 10 and the length of the pattern p is three as there are three items in it.

The objective of *frequent itemset mining* is to discover all itemsets (patterns) having a support greater than some threshold S known as the *minimum support*. An itemset $I_{closed} \subset I$ is called a *closed* itemset, if there exists no *super-set* that has the same support count as this original itemset I_{closed}. Closed itemsets or closed patterns allow to describe the transactions in a compact manner. There are efficient algorithms such as LCM [24] to mine closed itemsets in linear time. Then the set of patterns or itemsets discovered by the LCM algorithm is denoted by P, where $P = \{p_1, \cdots p_l\}$.

Transaction item matrix (M): One way to visualize items and transactions is by using a binary matrix M, where rows represent the transactions and columns represents the items. If transaction t_r has the item i_c, then we set $M(r, c) = 1$, otherwise $M(r, c) = 0$. An example of this binary matrix representing transactions is shown in Figure 3 right hand-side.

4. Mining in the Dataset

Our image dataset summarization consists of three steps. First, we mine patterns using data mining techniques. Then we perform pattern selection to find the most interesting set of patterns that allows to summarize the data. In the final step we connect the selected patterns to find paths across the dataset. Next, we explain each of these steps in detail.

4.1. Pattern set mining

We start our dataset summarization by extracting a large set of random patches similar to [4, 19]. In this set, there are patches that cover representative parts in the images, but there are also background patches. Every random patch is used as a seed to establish a cluster of mid-level patches. However, the collective set of mid-level clusters offers a quite fragmented view on the image collection. Therefore, we use the exemplar LDA framework [11] to select a reduced set of n mid-level clusters.

When a initial seed patch (its exemplar LDA) fires on an image, we call it an activation. Activations are ranked based on the score of the exemplar LDA and we keep only the top k activations per patch as elements of the corresponding mid-level cluster. Then we proceed to the generation of transactions from images.

If a mid-level cluster (*i.e.* item say i) is activated for a given image (among the top k), then we add that item i to the transaction t of that image. We repeat this procedure over all images to create the set of transactions T. Afterwards, we use LCM closed itemset mining algorithm to find an initial set of closed patterns denoted by $P = \{p_i | p_i \subseteq I\}$. We remind the reader that each pattern p_i covers some images (transactions) of the archive (at least S). At the same time, a pattern is a collection of mid-level cluster activations that is observed in these images. These discriminative patterns allow to better capture regularity of visual content than the original mid-level patches. Some of this initial set of patterns are shown in Figure 2.

4.2. Tiling

The patterns we obtain by applying the LCM data mining algorithm correspond to co-occurring mid-level patches. However, the number of the discovered patterns can be enormous. Some of these pattern sets are highly correlated, quite often they represent very similar visual patterns with only slight variations between them. This phenomenon is known as pattern explosion in the data mining literature. Pattern explosion happens mostly because we use frequency as the criterion to discover patterns. Even though all these patterns resemble some meaningful visual aspect, since there are many of them, we need a pattern selection method. In other words, we need a criterion that allows us to quantify the interestingness of a pattern beyond the frequency. Then we can use this interestingness criterion to select a subset of meaningful patterns that allows to explain the entire database of images.

A good pattern should be able to cover a large number of images. Usually, smaller patterns (*i.e.* patterns with smaller number of items) appear in a large number of images. But these smaller patterns do not bring much additional information in comparison to the original mid-level patches. At the same time larger patterns appear in relatively small number of transactions (images), but they are informative and usually very specific. Ideally, we would like to have patterns that are large enough and appear in as many images as possible. Therefore, we define the following pattern interestingness criterion denoted by $V(p)$ considering the length and the support of the pattern.

$$V(p) = length(p) \times support(p). \quad (1)$$

In-fact the interestingness criterion denoted by $V(p)$ resembles a region in the binary transaction-item matrix. For example, in Figure 3, the red pattern (tile C) consists of items D, E and covers three images so it's support is three. This results in a rectangular region denoted by red having area of 2×3 which is equivalent to the interestingness of that pattern. Such a region is also known as a *tile*. A tile is characterized by a set of transactions t, and a set of items p where p is the pattern that generated the tile. We denote a tile by $X = \{t, p\}$[1].

Then our objective is to find the set of tiles that together cover the entire dataset with minimum redundancy. Ideally, we would like to cover the entire dataset with a few non-overlapping tiles. To do this we need to make sure that any of two tiles that are selected should not have common transactions *or* they should not have common items. We emphasize that if two tiles have common items but no common transaction, it is still a valid set of tiles.

Formally we seek a set of non-overlapping tiles Ω that maximize the tiling area. Given any two tiles $X_i, X_j \in \Omega$ generated from two patterns p_i and p_j that cover two sets of transactions $t_i, t_j \subseteq T$, they should satisfy the Boolean constraint

$$\neg((p_i \cap p_j \neq \emptyset) \wedge (t_i \cap t_j \neq \emptyset)). \quad (2)$$

We then have to select the set of tiles (patterns) that maximizes the covered area of the binary transaction item matrix M while satisfying the constraint in equation 2. Then our objective becomes finding the set of tiles that maximizes the following:

$$\Omega^* = argmax_\Omega \sum_{X_{i\{p_i,t_i\}} \in \Omega} V(p_i). \quad (3)$$

Finding the optimal set of tiles, s.t. $\forall X_i, X_j \in \Omega$ that maximizes the above constrained objective function is NP hard. As a result we use the following greedy algorithm(1) to find a good set of tiles. First, we sort all patterns according to the interestingness of the pattern. Then we add the most interesting pattern to the solution Ω. Then we go to the next most interesting candidate tile generated from the next most interesting pattern p_i. If the constraint in equation 2 is satisfied for all patterns in $\Omega \cup \{X_i\}$ (*i.e.* the new candidate X_i can overlap in transactions or items, but not in both), then we add X_i to Ω. We continue until we have processed all patterns.

[1]When t is used without a subscript, it corresponds to a set of transactions.

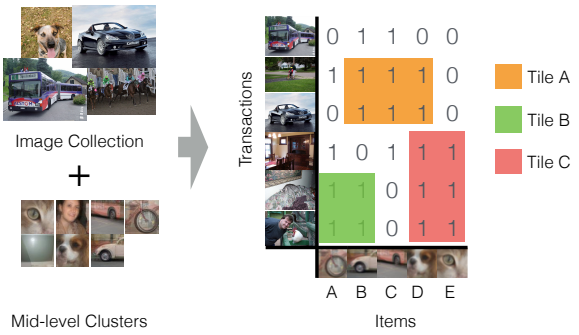

Figure 3: Overview of the tiling procedure: starting from a set of images we extract mid-level clusters and we generate the item-transaction matrix. Our algorithm searches for the set of tiles that together cover the matrix and do not overlap with each other.

Data: transaction item matrix M; set of LCM patterns P
Result: set of tiles Ω
$\Omega = \emptyset$;
$P \leftarrow$ sort patterns P in decreasing order according to interestingness score of equation 1.;
Add the most interesting pattern $p^* \in P$ to Ω ;
Remove p^* from P ;
for $p_i \in P$ **do**
  Generate candidate tile from p_i *i.e.* $X_i = \{p_i, t_i\}$;
  if *Eq. 2 is satisfied for* X_i *and* $\forall X_j \in \Omega$ **then**
    Add X_i to Ω ;
  end
end

Algorithm 1: Greedy tile mining process

This algorithm allows us to discover a set of maximally interesting patterns with minimum redundancy. Given two patterns (tiles) in the set Ω, there can be overlap only in transactions (images) or items (mid-level clusters), but not both at the same time. This allows to reduce the number of patterns without too much affecting the overall coverage of the dataset. In our experiments the initial set of 32,000 patterns covers 98% of the images while after pattern selection using tile mining we manage to cover 95% of the images

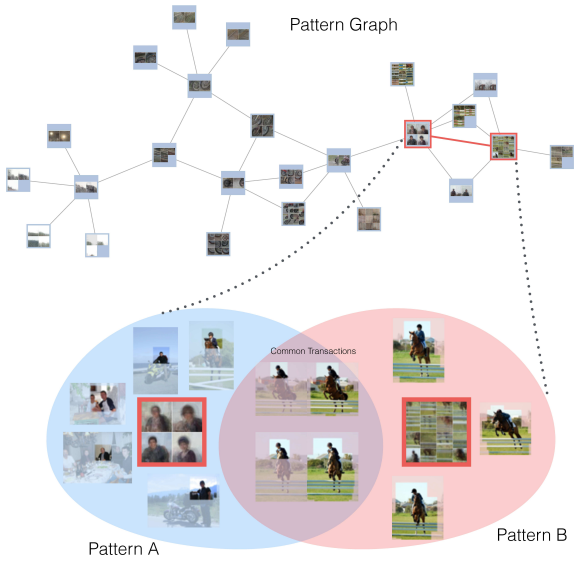

Figure 4: Top: part of the tile graph. Bottom: How two tiles (nodes) are connected based on co-occurrences in some images common to both.

with only 2,800 patterns (tiles).

4.3. Building links between tiles to discover paths and to navigate the archive

Now that we have summarized the image archive with a reduced number of patterns or tiles, we would like to find links between related tiles. Each tile can be seen as a specific view of a collection of transactions or images. If a set of images is covered by several tiles, this implies that those tiles are non-overlapping in the items (so no common mid-level clusters). This means that those tiles explain or see different aspects of the set of images. We use such sets of images shared by several tiles in order to connect the tiles and generate the tile graph. Then our objective is to navigate the dataset via this graph.

Here the generation of links between the tiles is explained. Firstly, we construct an affinity matrix A using the Jaccard-Similarity. Given two tiles $X_i, X_j \in \Omega$, the strength of the link is given by following Jaccard-Similarity defined over the overlap between the sets of transactions covered by $X_i\{t_i, p_i\}$ and $X_j\{t_j, p_j\}$ given by equation 4.

$$A(X_i, X_j) = \frac{t_i \cap t_j}{t_i \cup t_j} \tag{4}$$

In other words, two tiles (patterns) have high affinity if they often co-occur in the same images.

Afterwards we construct a fully-connected graph where nodes represent the tiles and links are weighted based on the affinity matrix A. We visualize this graph as follows:

- Each node shows the set of items (mid-level clusters) of the tile
- Each node is linked to other tiles via the shared transactions (images).

Figure 4 shows a part of the graph (not all links are shown for visibility) and how the links are created.

To discover the common transactions of the dataset, we go through each image of the archive and find which images are covered by two or more tiles. Arguably, those images that have large number of tiles are more representative and perhaps more interesting.

5. Results

Implementation Details In this section we present an analysis of different aspects of our system and some implementation details. We start by randomly sampling 10.000 patches similar to [4, 5, 19] and we train Exemplar LDA detectors for each patch. Every exemplar is applied to the dataset and its maximum response from every image is collected (max pooling). Next, the responses are sorted and for each exemplar detection only the top k scored patches are kept. This results in a sparse activation vector for every exemplar, indicating in which images it is activated. A low k value can result in an activation vector that contains more "pure" patches. However, this strategy may discard correct patches that have lower LDA score. From all the activation vectors we construct the transaction-item matrix M, with each transaction being the set of discriminative patches that are activated (*i.e.* belong to the top k) in an image.

We use the matrix M to generate closed patterns using LCM [24]. In order to keep the number of patterns manageable and without losing any important information we set the minimum number of items (the cardinality of the pattern) to 2 and the minimum support to s. This means that we want patterns with at least 2 exemplars, activated in at least s images. By increasing the minimum support, we impose the patterns to appear in more images, thus reducing the number of patterns found, resulting in a decrease in coverage. As mentioned before, the number of closed patterns can be too large to work with. In our work, therefore, we use the greedy tiling procedure described in Section 4 for generating a pattern set that is compact and describes the data well. In this step, we let the algorithm find the number of patterns. Note that, in contrast to [5, 19], we have not performed any type of spatial de-duplication step based on overlap or any other criterion.

Analysis Framework The free parameters of our system are the number k of the top activations and the minimum support s. We use the same subset of Pascal VOC 2007 as in [4, 19], which contains $\approx$ 1500 images from the classes

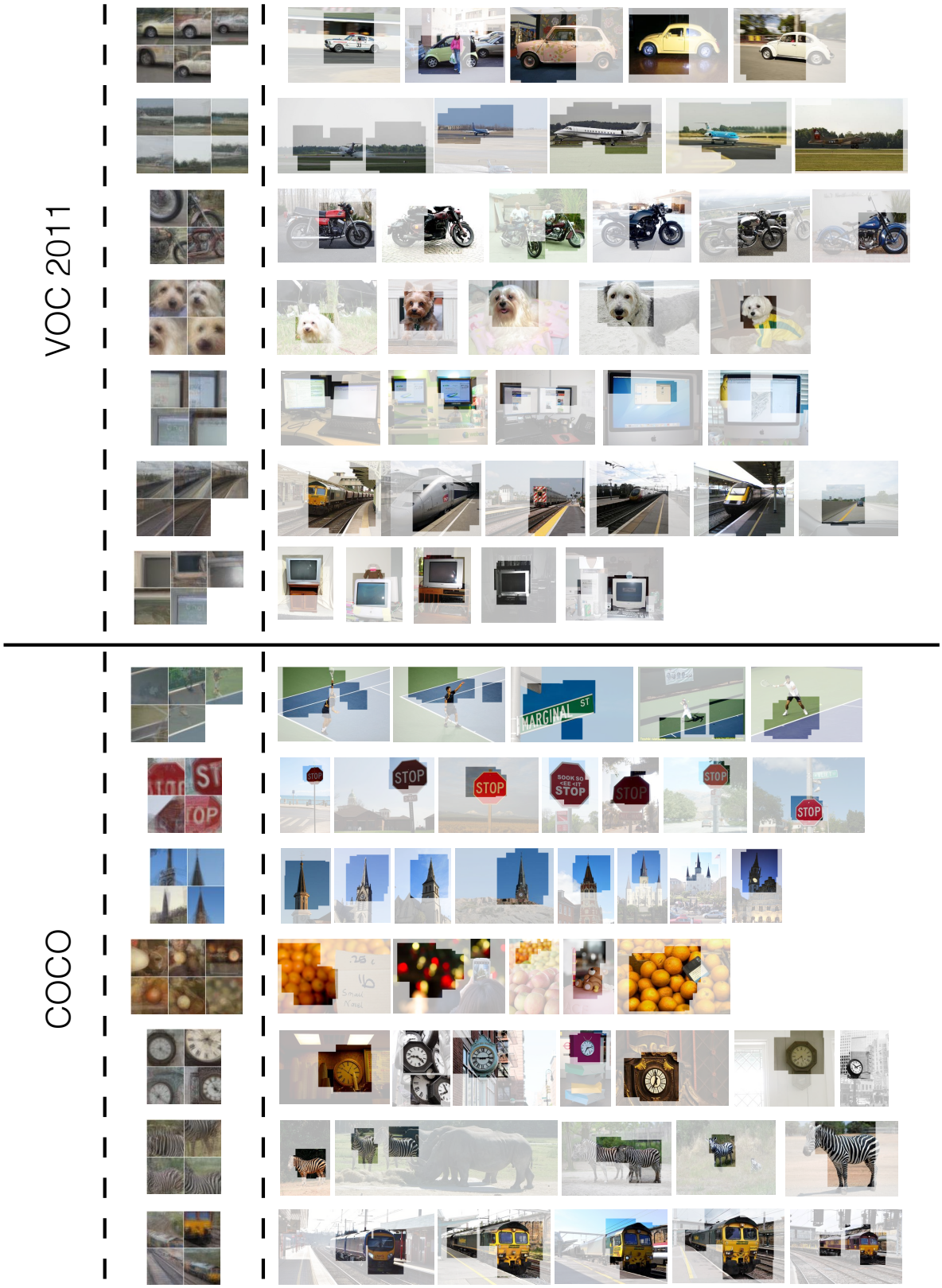

Figure 5: Example tiles together with their support for Pascal VOC 2011 and COCO. The support images are highlighted at the locations that the tile items are activated.

		Closed Patterns		Tiles		
k	s	Total number	Coverage	Total number	Coverage	AUC
20	5	33204	98.2	2874	95.45	0.84
20	6	13783	94.2	1201	87.78	0.75
20	7	6652	88	650	79.4	0.67
20	8	3531	79.5	398	71.59	0.6
25	5	116947	99.6	5801	98.72	0.88
25	6	46840	98.7	2656	96.45	0.83
25	7	22402	96	1345	91.55	0.76
25	8	11903	92.9	849	86.79	0.71
30	5	346914	99.9	8762	99.29	0.89
30	6	134295	99.6	4959	98.58	0.86
30	7	62531	98.5	2573	96.73	0.81
30	8	33059	97.4	1460	94.11	0.77

Table 1: Some quantitative statistics of our method.

horse, dining table, bus, train, sofa and motorbike. In Table 1 we show how the system behaves under different k and s values in terms of number of patterns, pattern set coverage and area under the curve (AUC). The AUC was defined over the purity-coverage plots (similar to [4]) that were calculated in the following way: we order the patterns in terms of class purity (looking at the majority Pascal VOC label in the pattern's support) and the i'th point is the average purity from 1 till i'th pattern against the number of images that patterns 1 to i cover.

From this analysis we can see that our framework is able to reduce considerably the number of patterns while maintaining high coverage. The more patterns we have the higher the purity we obtain. This is an expected behavior since we are able to include more patterns with class-specific nature. For example, a pattern that captures the upper body of a human is considered a "good" pattern, even if it fires in humans+horses and humans+motorbikes. For completeness, [4, 19] have 0.83 AUC score using 1000 discriminative elements, while our method achieves 0.74 AUC by forcing the tiles to be 1000. However, it should be noted that we in fact optimize a different criterion (eq. 3), since our approach is not class specific, like the object vs stuff [4] or vs natural world[19]. By allowing patterns that overlap in images we ensure that a large part of the objects in the images are covered (see for a comparison Figure 5 and 2). For instance (Figure 5) mid-level clusters find relatively smaller parts of televisions in VOC2011 dataset. On the other hand, in the right hand side of Figure 5, we show larger patterns discovered on televisions. In COCO dataset, while mid-level clusters find parts of zebras, our method discovers larger pattern combining several zebra patches. Therefore, our method allows to discover larger semantically meaningful regions of the images and link them with other images as shown in Figure 4. This is useful to browse image datasets in a meaningful manner.

Qualitative Evaluation Due to the difficulty of having a metric that shows how good our framework is, we present a qualitative evaluation of our system. In more detail, we apply our method in two large and realistic datasets, Pascal VOC 2011 [7] and Microsoft COCO [15]. Pascal VOC 2011 consists of 11000 images (training and validation set) from 20 different classes, while COCO is about 40000 images (validation set) with more than 70 classes. In both cases, we used 10000 exemplars, but k was set higher, since now there are more possible matches (to 80 and 100 for VOC and COCO respectively).

Figure 5 shows examples of our discovered tiles. All the examples presented in this paper come from the top 400 tiles w.r.t. to their area. For a more complete view of the discovered tiles, we refer the reader to our website. As mentioned before, our method is able to find constellations of patches and cover larger area of the discovered entity. Moreover, an interesting observation is about the semantically unrelated images that belong to the support of a tile. For example, the sign that is confused with playing tennis (first COCO row), or the bokeh effect in the oranges cluster (4th COCO row) have consistent configuration as the other images of the support. On the other hand, as shown in Figure 6, there are specific structures with high tile area that correspond to background regions (similar observations were made in [4]). These highly textured regions appear very often in the natural images and their structure is interpreted as patterns.

Browsing the dataset Once we establish connections between the tiles as is described in section 4, we are able to navigate and browse the image collection based on mid-level structures. To start the "tour", the system shows a set

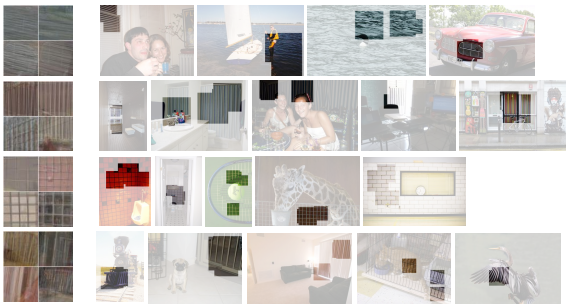

Figure 6: Background tiles in VOC and COCO. These tiles appear in background regions consistently.

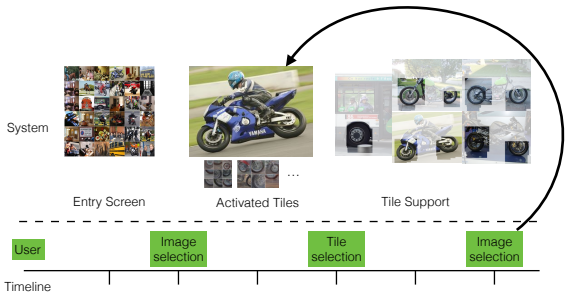

Figure 7: Visualization of the browsing process: a) the user selects one image from the initial set, b) the system shows the patterns that appear in that image, c) the user selects one pattern and d) the user can select the next image.

of images that are covered by many tiles and the user starts browsing by clicking one of the images. The set of tiles that contain this image in their support is shown. When the user selects a tile, its support is shown together with its activation in the images and the user can continue browsing in the same way. Figure 7 illustrates the browsing process.

This way of browsing allows the users to navigate in a semantic way. The discovered tiles of mid-level representations generate meaningful correspondences between images. Additionally, our navigation mechanism recommends multiple paths to continue. This enables efficient data traversal, increasing the ability to explore more visual aspects. We provide both a visual summary and a navigation demo in our project website `http://homes.esat.kuleuven.be/~krematas/DatasetFingerprints/`.

Discussion In all of our experiments, the input of our system is a set of randomly sampled patches without removing highly overlapping patches or the other duplicate-removal heuristics that are applied in [4, 19]. This can result in repetitive structures, such as multiple patches for covering the head. However, note that our approach is complementary to any other discriminative patches based approach, as their output can be directly used for the construction of the item-transaction matrix. In addition, using the randomly initialized set of patches without preprocessing, emphasizes more the effectiveness of the method.

The use of exemplar LDA is very effective for matching the exemplar's structure in different images, but its performance depends on the k parameter, namely the number of activations for the particular exemplar in all images. Setting k too low results in coherent clusters, but in several cases, depending on the data, leads to fragmentation of the true cluster. Setting k too high can result in low purity as random patches may be assigned to a cluster. The solution to this issue would be an adaptive threshold for setting this parameter, based on intra-cluster similarity. This is left as future work.

6. Conclusion

In this work, we have proposed dataset fingerprints, a new way of exploring image collections, based on interesting patterns discovered therein. We start from discriminative patches, which already do a good job focusing the attention on representative parts in the images. However, the collective set of discriminative patches offers a quite fragmented view on the image collection. Therefore, we propose to look at combinations of discriminative patches that often co-occur in the same images. These can be found efficiently using pattern mining techniques. We propose to select a set of patterns that together explain the representative parts of the images with as little redundancy as possible. While it is difficult to quantitatively evaluate unsupervised / subjective tools like these, from visual inspection we conclude that the discovered patterns make sense and indeed allow a user to quickly get familiar with the most important objects in a photo collection. Representative images link multiple patterns together in a graph-structure, allowing a user to browse the images and gradually explore the archive. We encourage the reader to visit our project website, where there are the summarization graphs for the Pascal VOC and COCO datasets as well an interactive browsing tool.

Acknowledgments This work is supported by the FP7 ERC starting grant Cognimund and the PARIS project (IWT-SBO-Nr. 110067).

References

[1] S. Cao and N. Snavely. Graph-based discriminative learning for location recognition. In *CVPR*, 2013.

[2] L. Carlone, P. F. Alcantarilla, H.-P. Chiu, Z. Kira, and F. Dellaert. Mining structure fragments for smart bundle adjustment. In *BMVC*, 2014.

[3] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR*, 2009.

[4] C. Doersch, A. Gupta, and A. A. Efros. Context as supervisory signal: Discovering objects with predictable context. In *ECCV*, 2014.

[5] C. Doersch, S. Singh, A. Gupta, J. Sivic, and A. A. Efros. What makes paris look like paris? *ACM Trans. Graph.*, 31:101, 2012.

[6] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010.

[7] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2011 (VOC2011) Results.

[8] A. Faktor and M. Irani. clustering by composition unsupervised discovery of image categories. In *ECCV*, pages 474–487. Springer, 2012.

[9] B. Fernando, E. Fromont, and T. Tuytelaars. Effective use of frequent itemset mining for image classification. In *ECCV*, pages 214–227. Springer, 2012.

[10] B. Fernando and T. Tuytelaars. Mining multiple queries for image retrieval: On-the-fly learning of an object-specific mid-level representation. In *ICCV*, 2013.

[11] B. Hariharan, J. Malik, and D. Ramanan. Discriminative decorrelation for clustering and classification. In *ECCV*. Springer, 2012.

[12] K. Heath, N. Gelfand, M. Ovsjanikov, M. Aanjaneya, and L. J. Guibas. Image webs: Computing and exploiting connectivity in image collections. In *CVPR*, 2010.

[13] G. Kim, C. Faloutsos, and M. Hebert. Unsupervised Modeling of Object Categories Using Link Analysis Techniques. In *CVPR)*, 2008.

[14] A. Kushal, B. Self, Y. Furukawa, D. Gallup, C. Hernandez, B. Curless, and S. M. Seitz. Photo tours. In *3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT)*, 2012.

[15] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014.

[16] J. Lokoč, T. Grošup, and T. Skopal. Image exploration using online feature extraction and reranking. In *ICMR*, 2012.

[17] J. Philbin, J. Sivic, and A. Zisserman. Geometric latent dirichlet allocation on a matching graph for large-scale image datasets. *IJCV*, 95(2):138–153, 2011.

[18] M. Sadeghi and A. Farhadi. Recognition using visual phrases. In *CVPR*, 2011.

[19] S. Singh, A. Gupta, and A. A. Efros. Unsupervised discovery of mid-level discriminative patches. In *ECCV*. Springer, 2012.

[20] N. Snavely, S. M. Seitz, and R. Szeliski. Photo tourism: exploring photo collections in 3d. *ACM transactions on graphics (TOG)*, 25(3):835–846, 2006.

[21] H. O. Song, R. Girshick, S. Jegelka, J. Mairal, Z. Harchaoui, T. Darrell, et al. On learning to localize objects with minimal supervision. In *ICML*, 2014.

[22] H. O. Song, Y. J. Lee, S. Jegelka, and T. Darrell. Weakly-supervised discovery of visual pattern configurations. In *NIPS*, 2014.

[23] The.New.York.Times. Fashion Week browser through thumbnails. `http://www.nytimes.com/newsgraphics/2013/09/13/fashion-week-editors-picks/`, 2014. [Online; accessed 10-Nov-2014].

[24] T. Uno, T. Asai, Y. Uchida, and H. Arimura. An efficient algorithm for enumerating closed patterns in transaction databases. In *Discovery Science*, 2004.

[25] M. van Leeuwen. Interactive data exploration using pattern mining. *International Journal on Artificial Intelligence Tools,*, to appear:1–14, 2014.

[26] J. Yuan, Y. Wu, and M. Yang. Discovery of collocation patterns: from visual words to visual phrases. In *CVPR*, pages 1–8. IEEE, 2007.

[27] J.-Y. Zhu, Y. J. Lee, and A. A. Efros. Averageexplorer: Interactive exploration and alignment of visual data collections. *ACM Transactions on Graphics (TOG)*, 33(4):160, 2014.