Représentation des images au moyen de motifs fréquents et émergents pour la classification et la recherche d'images

Winn Voravuthikunchai

UNIVERSITÉ DE CAEN BASSE NORMANDIE

université de Caen
Basse-Normandie

U.F.R. de Sciences
ÉCOLE DOCTORALE SIMEM

T H È S E

Présentée par

M. Winn VORAVUTHIKUNCHAI
en vue de l'obtention du

DOCTORAT de l'UNIVERSITÉ de CAEN
Spécialité : Informatique et applications

Arrêté du 07 août 2006

Le 11/12/2013

Titre :

Représentation des images au moyen de motifs fréquents et émergents pour la classification et la recherche d'images.

The work presented in this thesis was carried out at
GREYC - University of Caen

Jury

Mme. Élisa FROMONT	Maître de Conferences	Université Jean Monnet	*Examinateur*
M. Thierry PAQUET	Professeur des Universités	Université de Rouen	*Examinateur*
M. Georges QUÉNOT	Directeur de Recherche	CNRS, Grenoble	*Rapporteur*
M. Christophe RIGOTTI	Maître de Conferences (HDR)	INSA de Lyon	*Rapporteur*
M. Bruno CRÉMILLEUX	Professeur des Universités	Université de Caen	*Directeur de thèse*
M. Frédéric JURIE	Professeur des Universités	Université de Caen	*Directeur de thèse*

Abstract

In this thesis, our aim is to achieve better results in several tasks in computer vision by focusing on the image representation part. Our idea is to integrate feature dependencies to the original feature representation. Although feature dependencies can give additional useful information to discriminate images, it is a nontrivial task to select a subset of feature combinations from the power set of the features which has an excessively large cardinality.

We employ pattern mining techniques to efficiently produce a tractable set of effective combinations. Pattern mining is a process that can analyze large quantities of data and extract interesting patterns such as groups of data records (cluster analysis), unusual records (anomaly detection) and dependencies (association rule mining).

The first encountered problem is how to encode image features which are typically real valued as binary transaction items suitable for pattern mining algorithms. We propose some solutions based on local thresholding. The number of extracted patterns is still very high and to use them directly as new features for inferring a supervised classification models leads to overfitting. A solution by aggregating the patterns and have a compact representation which does not overfit to the training data is presented. We have achieved state-of-the-art results on several image classification benchmarks.

Along the path of exploration, we realize pattern mining algorithms are suitable especially for large scale tasks as they are very efficient and scale gracefully to the number of images. We have found two suitable applications. The first one is to detect groups of duplicates in very large dataset. In order to run our experiment, we created a database of one million images. The images are randomly downloaded from Google. We have discovered the duplicate groups in less than three minutes. Another application that we found suitable for applying pattern mining techniques is image re-ranking. Our method can improves the original ranking score by a large margin and compare favorably to existing approaches.

Contents

Abstract **1**

1 Introduction **7**
1.1 Computer vision tasks addressed by the thesis 9
1.1.1 Image classification 9
1.1.2 Object detection 9
1.1.3 Image re-ranking sub-task in content-based image retrieval 10
1.1.4 Groups of duplicate image detection sub-task in mage clustering . . 12
1.2 Image representations 13
1.2.1 Local invariant features 14
1.2.2 Bag of Words 15
1.2.3 Spatial Pyramid Representation 16
1.3 Pattern mining 16
1.3.1 Pattern mining and terminologies 18
1.3.2 Frequent itemset 18
1.3.3 Maximal frequent itemset mining 19
1.3.4 Closed frequent itemset 19
1.3.5 Emerging pattern mining 20
1.4 Motivations of the thesis 21
1.5 Thesis organization and contributions 22

2 Preliminary Investigations **25**
2.1 Introduction 25
2.2 Related work 26
2.3 Three types of feature dependencies 27
2.3.1 Co-occuring features 28
2.3.2 Relative difference between features 32
2.3.3 The spatial relation between features 35
2.3.4 Comparison of the three feature dependencies. 37
2.4 Experiments 37
2.4.1 Pascal VOC 2007 dataset 37
2.4.2 Size of vocabulary 38
2.4.3 Results on the full dataset 39
2.5 Conclusions 40

3 Image Re-ranking **43**
3.1 Introduction 43
3.2 Related Work 45
3.3 Method 47

3.3.1 Binary representation of visual attributes . . . 47
3.3.2 Closed frequent itemset mining and matrix transposition . . . 47
3.3.3 Re-rank images . . . 48
3.4 Experiments . . . 51
3.4.1 Datasets and evaluation protocol . . . 51
3.4.2 Visual attributes . . . 52
3.4.3 Parameters . . . 52
3.4.4 Closed frequent itemset weights . . . 53
3.4.5 Complexity and computational time. . . . 54
3.4.6 Comparison to other methods . . . 55
3.5 Conclusions . . . 57

4 Histograms of Jumping Patterns for Image Classification and Object Recognition 65
4.1 Introduction . . . 65
4.2 Related Work . . . 68
4.3 Method . . . 69
4.4 Experiments . . . 73
4.4.1 On setting the parameters . . . 74
4.4.2 Texture recognition . . . 76
4.4.3 Pedestrian recognition . . . 77
4.4.4 Image classification . . . 78
4.4.5 Object detection . . . 79
4.5 Extension of multiple random projections to image re-ranking . . . 81
4.5.1 Parameters and complexity analysis. . . . 82
4.5.2 Comparison on INRIA Web Queries dataset. . . . 82
4.5.3 Comparison on QUAERO's visual concepts image dataset . . . 83
4.5.4 Summary on multiple random projections for image re-ranking. . . . 84
4.6 Conclusions . . . 84

5 Finding Groups of Duplicate Images 85
5.1 Introduction . . . 85
5.2 Related Work . . . 87
5.3 Method . . . 88
5.3.1 Coding image and transactions of visual properties . . . 90
5.3.2 Mining groups of similar images . . . 91
5.4 Experiments . . . 92
5.4.1 Datasets . . . 92
5.4.2 Representing images as transactions of itemsets . . . 93
5.4.3 Duplicate detection . . . 95
5.5 Conclusions . . . 97

6 Conclusions and Future work 99
6.1 Preliminary Investigations . . . 99
6.2 Image Re-ranking . . . 101
6.3 Histograms of Jumping Patterns for Image Classification and Object Recognition . . . 101
6.4 Finding Groups of Duplicate Images in Very Large Datasets . . . 102
6.5 Conclusions and perspectives . . . 103

List of figures **109**

List of tables **111**

Bibliography **120**

Chapter 1

Introduction

Contents

1.1 Computer vision tasks addressed by the thesis 9
1.1.1 Image classification 9
1.1.2 Object detection 9
1.1.3 Image re-ranking sub-task in content-based image retrieval . . . 10
1.1.4 Groups of duplicate image detection sub-task in mage clustering . 12
1.2 Image representations 13
1.2.1 Local invariant features 14
1.2.2 Bag of Words 15
1.2.3 Spatial Pyramid Representation 16
1.3 Pattern mining 16
1.3.1 Pattern mining and terminologies 18
1.3.2 Frequent itemset 18
1.3.3 Maximal frequent itemset mining 19
1.3.4 Closed frequent itemset 19
1.3.5 Emerging pattern mining 20
1.4 Motivations of the thesis 21
1.5 Thesis organization and contributions 22

When we look at an image, we can easily tell what objects it contains and to which categories they belong. *'This is a picture of two dogs. One of them is my dog, Ben.'* We can also describe their appearance and locations. *'Ben is the big brown dog on the right of the image.'* If the objects are living being, we can also tell their actions and even their emotions. *'Ben is sitting and seems to be happy.'* These abilities are not restricted to only objects, since we can also describe the global nature of scene images. *'This image was taken in a garden during day time.'* Moreover, without prior knowledge, we can still

compare different objects and tell how similar they look. *'The two objects look very similar to each other but the one on the left looks a bit smaller.*

In the field of *computer vision*, the development of an artificial intelligence visual system that allows machines or robots to have the ability to *see* and to *interpret* as humans, is widely regarded as the 'holy grail'. Even though, this ultimate goal is still far, the field has rapidly progressed, and there has been success in finding solutions to particular tasks. Several computer vision applications and products that have appeared in recent years are proof of such claims. They have been aiding humans in various aspects of daily life including 'safety', 'security', 'medical', 'robotics', 'photography', and many more. Some examples of computer vision applications are given in Section 1.1.

Visual systems consist of mainly two parts, the input and the interpreter. As for human, the input is the *light signal* reflecting from the objects and the interpreter is the *brain*. Differently from humans, machines receive an image input typically as a *vector of numerical values (features)* and a *machine learning model* is used to interpret the image from these numerical value features.

In this thesis, one main focus is to improve the image representation by *encoding feature dependencies* (Section 1.4). One big limitation of current image representations is the lack of modeling of features dependencies, such as *co-occurrences* of features, *relative differences* between features, and *spatial relationships* between features. Intuitively, such kind of information is expected to be useful for describing images. For example, in an 'aeroplane' image, usually there is a large amount of uniform patch features from the 'sky' and some rigid shape features from the 'aeroplane'. The dependencies between the uniform patch features and the rigid shape features can be more informative than considering each feature independently. However, encoding feature dependencies is a non-trivial task. Since image features are typically high dimensional (in a range of hundreds to thousands), it is not feasible to encode the dependencies of all feature combinations. Even selecting a subset of the combinations is also very difficult since the *space of feature combinations* is very large.

Our main idea to tackle the problem of encoding feature dependencies is to make use of *pattern mining* algorithms. Pattern mining is the field where the main problem is to analyze large quantities of data and *extract interesting feature combinations*. Besides the possibility to extract feature combinations, the fact that pattern mining algorithms are efficient (they can extract information from millions of data records in a very short time), another goal of the thesis is to *adapt pattern mining techniques to some large scale computer vision tasks* in which efficiency is important.

The organization of this chapter is as follows. Due to the large range of applications, research in computer vision has been split into several topics and sub-topics. In Section 1.1, we describe the computer vision tasks addressed by the thesis, to give some insights on

the different topics and their real world applications.

Since this study is related to two different fields (*i.e.* to *image representations* in the field of computer vision and *pattern mining* in the field of data mining), we provide some background knowledge related to these fields in Section 1.2 and Section 1.3 respectively. Then, in Section 1.4, we emphasize the problems we address in this study. The thesis organization and the outline of the contributions are given Section 1.5.

1.1 Computer vision tasks addressed by the thesis

1.1.1 Image classification

Image classification (mentioned in Chapter 2 and Chapter 4), is one of the major tasks in computer vision. Given an input image, the machine has to be able to recognize the content in the images and to classify the images into categories[1] or sub-categories to which they belong. The (sub-)categories here are predefined. For example, in *scene classification*, the categories could be 'forest', 'beach', 'city', while in *object recognition (object image classification)*, they are generally related to physical objects, such as 'bicycle', 'dog', 'table'.

In object recognition, if an image contains an instance of an object class, the image is considered as belonging to that object class. Object recognition techniques can be used to automatically annotate or tag images in large-scale databases based on the objects present in them. Once images are annotated, they can be efficiently searched with the use of offline created inverted indices on tags. Scene classification systems have been integrated into a number of modern digital cameras. Such cameras are able to automatically categorize the current scene being pictured by the user into one of many scene classes and adjust the camera settings to optimize image quality. *Texture categorization* is also an important ability that can be used by exploration robots. Once robots detect the ground condition based on texture, they can estimate motion parameters accordingly. Figure 1.1 shows some images of real-world applications related to image classification.

1.1.2 Object detection

While first impressions suggest that the tasks of *object detection* (mentioned in Chapter 4) and object recognition are the same, they are in fact differentiated by the final goal of each task. The main difference being, in object detection the locations of the objects in the image have to be given as the system output, while in object recognition it suffices to say whether the object is present in the image or not, regardless of its location. The

[1]The terms 'class' and 'categories' are used interchangeable in the following

(a) Scene recognition in modern cameras can automatically optimize the shooting settings for the current shot. *(Figure from Sony.)*

(b) Texture categorization may be used on exploration robots to navigate in different ground condition. *(Figure from NASA.)*

Figure 1.1: Image classification for a wide range of applications.

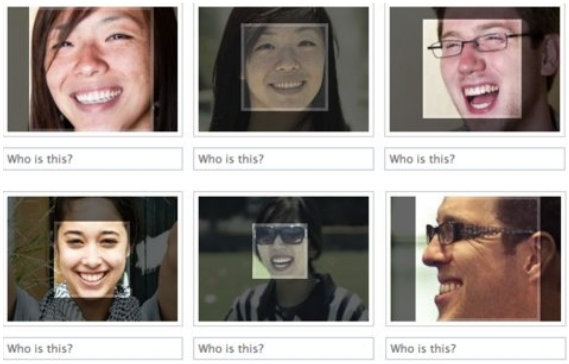
basic approach of object detection is to try to recognize objects in several sub-regions in images, rather than looking at the image as a whole. If a sub-region is classified as an object, its location is given as the position of the object. Typically, the sub-regions are evaluated at different sizes and positions in a *sliding window* fashion [97].

Consider *face detection* as an example of object detection systems. The success in the development of face detection systems is apparent nowadays. Such systems display very advanced capabilities, which have been seamlessly integrated into several daily use devices, such as portable computers and modern digital cameras. Consider the case of the latter device, when pictures are being taken, the camera detects faces of people present in the scene, and automatically adjusts the focus according to the distance to the subjects. Face detection is also prominent in online social networking services. When an image is uploaded, faces in the picture will be automatically detected, allowing to tag people easily.

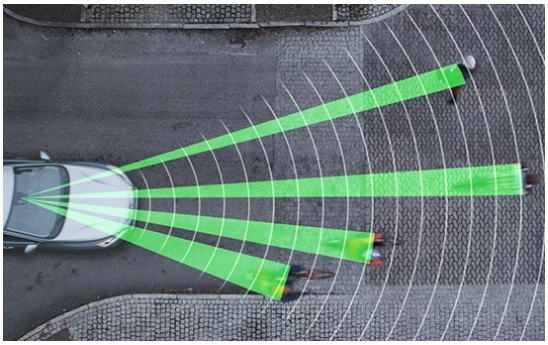
Another very important application of object detection, is *pedestrian detection*. The *World Health Organization (WHO)* has reported that in 2010, 22.4% of the total 1.24 million people killed in traffic were pedestrians [73]. If the development of pedestrian detection becomes advanced enough in the near future, it could be easily expected to have reliable pedestrian detection systems on-board vehicles. These systems could assist drivers in order to avoid accidents, which would hopefully lead to a large decrease in the number of fatalities and injuries. It is evident from such arguments, the potential of how important pedestrian detection could be in modern vehicles.

1.1.3 Image re-ranking sub-task in content-based image retrieval

The emergence of digital images, the advances in disk storage technology, together with the impact of the Internet in later years, have led to huge sizes of image databases in a scale of several millions. The current problem is related to the question of how to



(a) Face detection in social networking services help the users in tagging their friends. *(Figure from Facebook.)*



(b) Pedestrian detection assists driver to avoid accidents. *(Figure from Volvo.)*

Figure 1.2: Object detection is useful in wide range of applications.

efficiently find or retrieve a particular image or a set of images of interest, in a very large image database, such as those of *'Google'* and *'Yahoo'*. Search on such databases is typically based on textual information. For example, given a text query to a search engine, the search engine returns images based on their meta-data, such as keywords, tags, and/or descriptions associated with the image.

Textual meta-data is not always reliable as sometimes the text does not correspond to the visual content in the images. Moreover, there are many image databases where the textual information is not readily available such as untagged photos stored in personal computers or images captured from surveillance cameras. *Content-based image retrieval* is another active research topic in computer vision, which, in simple words, is about how to retrieve images based on their visual content rather than relying on associated textual information. One recent interesting application is to use hand-held devices with integrated cameras to quickly find information of surrounding objects or landmarks that we are interested in. The idea is to take a picture of the interested subject,which will be analyzed by a server, returning a set of visually similar images with associated information (Figure 1.3). Another example could be an application to aid in *crime investigation*. When a crime occurs, it is a nontrivial task to manually search for objects or persons that could be related to the crime in video sequences captured by surveillance cameras. If some clues about object appearances are available in advance, we can use content-based image retrieval systems to find the suspected frames in the sequences.

There is a sub-topic in content-based image retrieval called *image re-ranking* (mentioned in Chapter 3). Image re-ranking aims at improving the results of textual/visual image retrieval systems. To search for images in very large database in a scale of several millions, the retrieval algorithms have to be very efficient. In order to make fast comparisons between the query and the images in the database, the description of the images have to be very compact and for the compactness, a lot of information has to be discarded. This

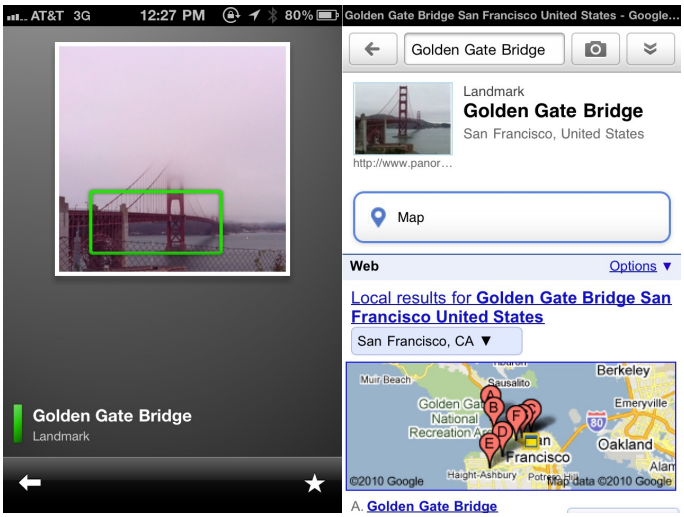


Figure 1.3: A mobile application, named 'Google Googles', returns similar images based on visual content associated with useful information to the user.(Figure from Google.)

is why the search is usually based on keywords or tags. However as the tags (typically the image filename) are noisy, such retrieval systems do not give very good results. The retrievals are usually mixed with noisy irrelevant images which is undesirable. However, the set of images retrieved are much smaller than the entire database, and therefore, a second process can be used to extract visual information from the small set of retrievals and re-rank them according to how relevant they are to the query.

1.1.4 Groups of duplicate image detection sub-task in mage clustering

When the number of images in the database is large, it is important to organize the images in such as way that we can easily search and navigate through the database. *Image clustering* is a key technique for organizing the database by grouping images that are visually similar to each other together. Instead of displaying all of the images in the database, only one representative image per group/cluster can be shown to get the overview of the database. If we want to search for an image, we can narrow down our search by looking into the cluster that we are interested. In the case when the database is organized in a hierarchical clustering derived structure, we can narrow down our search level by level to find the image the we are looking for.

In the context of multimedia search engines, image clustering is also a technique for disambiguating text-based image searches. A classical example is the *'jaguar'* query, which is expected to retrieve at least two groups of images; (i) animals on the one hand and (ii) cars of brand 'jaguar' on the other hand. Leveraging image clustering in this context clearly is a strong added-value for the end user, in that it automatically identifies the two distinct types of 'jaguar' images (Figure 1.4).

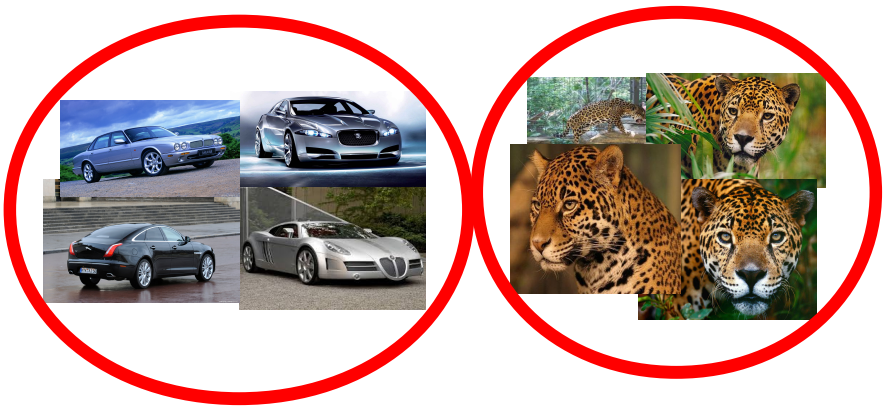


Figure 1.4: Two clusters of 'jaguar' images (the car brand and the animal). Image clustering resolves ambiguity occurring from homonymous queries in text based image searches.

In image databases especially in very large scales database of several millions or billions images, it is difficult to avoid having a large amount of duplicate or near duplicate images in the database. Closely related to image clustering, we are interested in *Groups of duplicate image detection* (mentioned in Chapter 5). The problem of discovering duplicates can be use in applications such as *image collection management* by eliminating duplicates and keeping only one instance per duplicate set. Not only can it improve space efficiency but also allows the user to access their images more efficiently.

1.2 Image representations

As mentioned earlier, visual systems consist of mainly two parts, the input and the interpreter. Machines receive an image input typically as a vector of numerical values (features). The success of the interpreter part relies heavily on the the image features. The key of feature construction is that images belong to the same class of interest should have similar representations, and have different representations than images belonging to other classes. The issue is to construct features that can be robust against the large range of diversities including:

Intra-class variations: instances of an object class, can be very different from one to another.

Orientation: objects can be viewed at a variety of possible orientations with respect to the camera, leading to different orientations.

Pose: non-rigid objects, especially living objects, have non-rigid articulated body. Therefore, global shape can encounter a large range of transformations due to the variety

of possible poses that can be assumed.

Position: objects can be at various position and distances respected to the camera. This also gives confusion between physical sizes of objects and scaling sizes depending on the distance to the camera.

Occlusion: objects can be occluded by one or several other objects.

Illumination depending on the lighting condition when the images are captured, the brightness and color of images of same scene/object can be very different.

Machines typically take inputs as numerical vectors. There are several methods for transforming a digital image what we see, to a vector representation for machines. The simplest method is to stack the intensity value of each pixels in the image sequentially. This method can be applied to controlled scenarios, where the objects in each image are well aligned to each other *i.e.* they are in the same scale, position, and pose. Moreover, there has to be no background clutter or the background has to be almost the same in all images. Also the illumination has to be strongly controlled. This kind of representation has been applied to some face recognition systems [5, 91]. In practice, these constraints are too strong for real life applications, and more invariant representations have to be used.

1.2.1 Local invariant features

As said before, raw-pixel intensity gives poor performance, in the general unconstraint scenarios. Some slight change in the illumination can dramatically change the pixel intensity values. Therefore, images belonging to the same class may have very different image representations. Local features are designed to overcome such problems. Local features are some form of low-level statistics extracted from small patches in images. Low-level features are based on pixel intensity, pixel gradient or pixel color. The statistics are typically captured by histograms. For example, the orientations of pixel gradients which can have values from 0 up to 360 degrees are quantized into a finite set of orientation bins, then the quantized gradients are counted in each image patch.

Several types of local features have been proposed in literature. The most popular ones are *Scale-Invariant Feature Transform (SIFT)* [64] and *Histogram of Oriented Gradients (HOG)* [21] in which both of them are based on edge orientations. While *Local Binary Patterns* (LBP) [90], *Local Ternary Patterns (LTP) [88]*, and *Local Quantized Patterns (LQP)* [45] are based on intensities values. As mentioned before, local features are extracted from small image patches. Moreover, the patches are often sampled at points of interest. For example, in face recognition, the patches are typically located on the eyes, the nose, and the month of the face. However, the patches could also be sampled densely at a constant pixel step size.

After computing local features from each patch, some systems concatenate all the local

features sequentially as the final image representation (*e.g.* [21, 103, 110]). Note that in order to compare feature vectors, the dimension of the vectors need to be the same. Therefore, in this kind of representation, it is necessary to have the same number of patches in all images. Moreover, the images have to be scaled to the same size and aspect ratio so that the patches located at the same position of each images can capture information from the same region of the objects. Although this kind of representation is more robust to illumination changes, and slight changes in object appearances, it is again suitable only for well aligned images. Applications for this kind of representation includes face recognition [14, 54, 68] and object detection [28, 30, 21], where the object of interest is in the center of a fixed size bounding box. Mentioned in Section 1.1.2, related to the object detection task, the visual content in each bounding box can be considered as an image to be classified (this is therefore also related to image classification).

1.2.2 Bag of Words

For the general task of image classification, the object of interest can be positioned anywhere in the image which different sizes and scales. Therefore, encoding an image representation by concatenating local features is not suitable.

The current state-of-the-art image representation for image classification is called the *Bag of Words (BoW)* representation [22]. This representation is inspired from *text document analysis*. In text document classification, a document is represented as an unordered collection of words, without taking grammar and even word order into account. The frequency of each word is used as a feature for training a classifier. The principle is that, it is possible to categorize the type of document by just knowing words appearing in the documents. The concept of the BoW representation is shown in Figure 1.5. The steps for representing an image similar to the BoW representation of documents are as follows,

(i) collect a large amount of local features from many images in the database and perform clustering such as *K-mean* clustering. The centroids of the clusters are considers as *visual words* similar to the words in text documents. The set of centroids is called a *codebook*.

(ii) map each local feature in an image to a visual word in which the visual word is the one which has the closest distance to the local feature.

(iii) at this stage, an image is similar to a document which contains a set of words. The frequency of each word appearing in an image is used as an image feature.

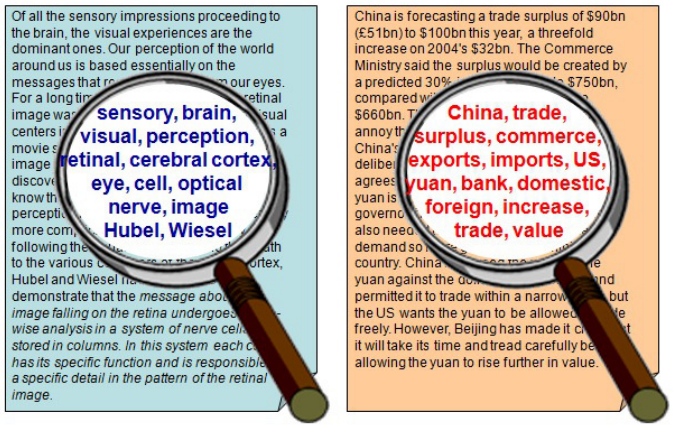


Figure 1.5: Concept of Bag of Words representation (Figure from Google Image). By just looking at the words without taking to account the grammar and word orders, we are still able to tell that the document on the left is related to 'health' while the document on the right is related to 'business'.

1.2.3 Spatial Pyramid Representation

In the BoW representation, the spatial information is totally discarded. However, the spatial information is expected to be important for representing images. For example, in a typical 'car' image, there are some uniform shape features from the 'sky', at the top of the image, some rigid shape features from the 'car' in the center, and some uniform shape features from the 'road' at the bottom of the image.

Several different methods have been proposed to incorporate the spatial information into the BoF representation [57, 62, 66, 80, 106] but the simplest and most commonly used is the *Spatial Pyramid Representation (SPR)* [57]. SPR works by dividing the image spatially with uniform grids at multiple resolutions. BoW vectors are computed in sub-region and the concatenation of all BoW vectors is the final representation of the image. Figure 1.6 is a toy example showing how to compute the SPR.

The rough spatial layout in encoded into the overall representation. This simple way of incorporating spatial information has proved to be very effective and improves the simple BoW by a large margin. Note that not only SPR is applied to BoW features, but also, it has been applied to most of recent image representations (*e.g. Semantic attributes* [85], *Fisher Vector* [78], *Frequent Local Histograms (FLH)* [33]) and show improvement in all cases.

1.3 Pattern mining

As mentioned at the beginning of the chapter, one goal of this thesis is to investigate how efficient pattern mining algorithms could be used to solve computer vision problems.

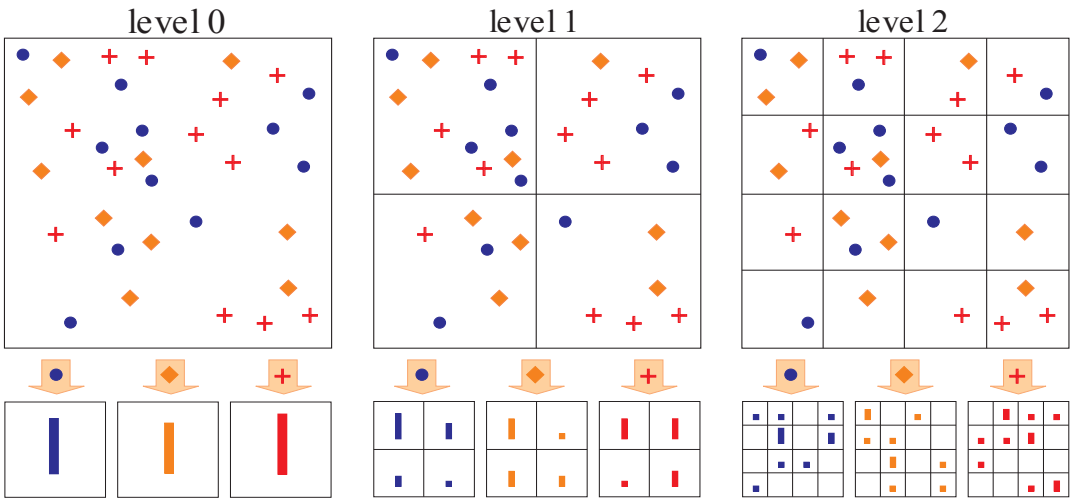


Figure 1.6: Toy example of constructing a three-level spatial pyramid [57]. The image has three feature types, indicated by circles, diamonds, and crosses. The image is subdivided at three different levels of resolution. Next, for each level of resolution, the features that fall in each spatial bin are counted. Finally, the histograms computed at each levels are concatenated.

In this section, we introduce pattern mining, explain the terminology used in pattern mining, and present some types of patterns used in our study.

Pattern mining was originated from the study of business analysis [1]. One of its very first application was to find patterns in the buying behaviors of consumers. A pattern can be, for instance, a set of items often being purchased together. Some common types of patterns that can be extracted from a shopping history database are such as

Frequent itemsets: sets of items that are frequently being bought together (*e.g.* '50 percents of the customers buy bread and milk together.').

Sequential (frequent) patterns: the items that are frequently being bought in an order (*e.g.* '20 percents of the customers buy first cameras with integrated kit lens, then they buy zoom lens, and after that they buy wide angle lens.').

Emerging patterns: sets of items in which the frequency of buying is very different depending on the class of customers (*e.g.* 'from all customers who buy video games and skateboards together, 95 percents of them boys whereas only 5 percents are girls.').

The information from the extracted patterns are used to improve marketing strategies such as to set sale promotions or to arrange supermarket shelves. Since then, patterns have been extracted from various kinds of data from different domains such as 'gene expression', 'chemical compounds', 'social network', 'stock price'. Many more types of patterns have been defined to suit different applications. The patterns are defined based on constraints such as *closed patterns*, *maximal patterns*, *discriminative patterns* or based

on data type such as *itemsets*, *sequence*, *tree*, *graph*.

Over two decades, the research has been progress tremendously in terms of efficiency and scalability. It is very impressive that there are very efficient algorithms that can extract patterns from millions of transactions or data points in a very short time. We introduce here the pattern mining terminology and some important types of pattern mining used in this study.

1.3.1 Pattern mining and terminologies

Let $\mathcal{I} = \{i_1, i_2, \ldots, i_n\}$ be the set of the n items under study, an itemset $\mathcal{X} \subset \mathcal{I}$ is a subset of items from $\mathcal{I}$. A transaction $\mathcal{T}_i \subset \mathcal{I}$, is a set of items being processed/purchased together. The database $\mathcal{D} = \{\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_m\}$ is the set of transactions collected for the study.

The *support* of $\mathcal{X}$, denoted as $\mathcal{S}(\mathcal{X})$, is the set of transactions containing itemset $\mathcal{X}$. More formally, $\mathcal{S}(\mathcal{X}) = \{\mathcal{T}_i \in \mathcal{D}, i = 1, \ldots, m \mid \mathcal{X} \subseteq \mathcal{T}_i\}$. The *frequency* of $\mathcal{X}$, defined as $fr(\mathcal{X})$, is the number of transactions containing $\mathcal{X}$. More formally, $fr(\mathcal{X}) = \mathrm{Card}(\mathcal{S}(\mathcal{X}))$. The frequency corresponds, for instance, to the number of times that the items in $\mathcal{X}$ have been purchased together. The *length* of $\mathcal{X}$, denoted as $l(\mathcal{X})$, is the number of items in $\mathcal{X}$. More formally, $l(\mathcal{X}) = \mathrm{Card}(\mathcal{X})$. In other words, $l(\mathcal{X})$ is the number of items shared between the transactions in $\mathcal{S}(\mathcal{X})$. Note that the more items shared between the transactions, the more similar the transactions are.

1.3.2 Frequent itemset

The most basic type of pattern capturing co-occurrence of items is called *Frequent itemset* proposed in 1993 [1]. Frequent itemset is an itemset that occurs frequently in the database. The term frequent depends on a minimum frequency threshold F_{min} defined from the user. More formally, $\mathcal{X}$ is a frequent itemset if $fr(\mathcal{X}) >= F_{min}$. For instance, in Figure 1.7, $\mathcal{X} = \{i_2, i_3\}$ is a frequent itemset respect to $F_{min} = 2$, since this $\mathcal{X}$ occurs two times in the database which statisfied the condition to be more or equal than the defined $F_{min} = 2$.

It is simple to find some frequent itemsets but the difficulty is to extract the whole set of frequent itemsets in the database. One main property to efficiently find the complete set of frequent itemsets is that every subset of a frequent itemset is also a frequent itemset. This property is called the *apriori property* or also called the *downward closure property*. This property can be one of the clearest example in order to demonstrate how pattern mining can to quickly prune out the unsatisfied feature combinations. If one feature combination does not fulfill the frequency criteria which means that the combination

appear less than the specific threshold in the database, it is unnecessary to evaluate all of its superset because they will not fulfill the frequency criteria as well.

1.3.3 Maximal frequent itemset mining

Due to the fact that every subset of a frequent itemset is also a frequent itemset, a frequent itemset will contain a combinatorial number of sub-frequent itemset which can be very large, especially when the frequent itemset is long. For example, if a frequent itemset contains 20 items, there will be millions ($2^{20} - 1 = 1,048,576$) of sub-frequent itemsets. This large number of frequent itemsets is not only difficult to handle in further processes but also contains a lot of redundant information. Rather than using all frequent itemsets, in many cases, only *Maximal frequent itemsets* are used.

Maximal frequent itemset mining was proposed in 1998 [4]. A maximal frequent itemset is a frequent itemset in which none of its immediate supersets is frequent. Typical frequent itemset mining algorithms use the *downward closure* property and find frequent itemsets starting from frequent itemsets of single items. Each frequent itemset is extended level by level and stop at the level when the current frequent itemset is a maximal frequent itemset. When frequent itemsets are expected to be long, this procedure can be very time consuming because the search space is huge. Specifics algorithms such as *Max-Miner* [4] have been invented to extract only the maximal frequent itemsets without going through all of their sub-frequent itemsets.

Starting from the set of all maximal frequent itemsets, it is possible to generate all frequent itemsets by making combinations between the items in each maximal frequent itemset. However, the drawback is that it is not possible to obtain the frequencies of those sub-itemsets.

1.3.4 Closed frequent itemset

The question of how to contruct a *lossless* condense set of frequent itemsets (*i.e.* a condense set in which it is possible to generate all frequent itemsets from the condense set without losing the frequency counts.) leads to another type of frequent itemset called *closed frequent itemset*. Closed frequent itemset was proposed a year after Maximal frequent itemset in 1999 [75].

Regarding the definition of a closed frequent itemset, if $\mathcal{X}$ is a closed itemset, it is impossible to add any item to $\mathcal{X}$ without decreasing its frequency. To describe the mathematical definition, $\mathcal{X}$ is closed if $\nexists Y \supset \mathcal{X} \mid fr(\mathcal{X}) = fr(\mathcal{Y})$ where $\mathcal{Y}$ is any superset of $\mathcal{X}$. We can also explain the definition of a closed itemset in another way, a closed itemset is the longest itemset in its *equivalence class*. All itemsets which have the same support are considered as members of the same equivalence class. Using all members in the equivalent

Transaction $\mathcal{T}_i$
$\mathcal{T}_1 = \{i_1, i_2, i_3\}$
$\mathcal{T}_2 = \{i_1, i_4, i_6\}$
$\mathcal{T}_3 = \{i_2, i_3, i_6\}$
$\mathcal{T}_4 = \{i_1, i_7, i_9\}$
$\mathcal{T}_5 = \{i_4, i_5, i_8\}$

(a) Database

Itemset x_j	$l(\mathcal{X}_j)$	$S(\mathcal{X}_j)$	$fr(\mathcal{X}_j)$
$\{i_1\}$	1	$\mathcal{T}_1$, $\mathcal{T}_2$, $\mathcal{T}_4$	3
$\{i_4\}$	1	$\mathcal{T}_2$, $\mathcal{T}_5$	2
$\{i_6\}$	1	$\mathcal{T}_2$, $\mathcal{T}_3$	2
$\{i_2, i_3\}$	2	$\mathcal{T}_1$, $\mathcal{T}_3$	2

(b) All closed frequent itemsets with $F_{min} = 2$

Figure 1.7: Example of closed frequent itemsets with $F_{min} = 2$ and the notations. $\mathcal{T}_i$: transactions, i_k: items, $\mathcal{X}_j$: itemsets, $l(\mathcal{X}_j)$: length of each of itemset $\mathcal{X}_j$, $S(\mathcal{X}_j)$: support of each itemset $\mathcal{X}$, and $fr(\mathcal{X}_j)$: frequency of each itemset $\mathcal{X}_j$.

class gives redundant information which is the reason in many cases it is favorable to use only closed frequent itemsets.

To give an example for explaining a closed itemset, in Table 1.7, $\{i_2\}$ and $\{i_2, i_3\}$ belongs to an equivalence class because both of them are extracted exactly from the same transactions which are $\mathcal{T}_1$ and $\mathcal{T}_3$. Although $\{i_2\}$ is also a frequent itemset with respected to $F_{min} = 2$ same as $\{i_2, i_3\}$, it is not a closed frequent itemset because there exists a superset, $\{i_2, i_3\}$, which have the same frequency. We can also see that $\{i_2, i_3\}$ is the longest member in its equivalence class (*i.e.* $\{\{i_2, i_3\}, \{i_2\}\}$) and therefore $\{i_2, i_3\}$ is a closed itemset.

Several efficient algorithms have been proposed for extracting closed frequent itemsets such as *CHARM* [109], *CLOSET* [77], *LCM* [92]. Note that all maximal frequent itemsets are also closed frequent itemsets but not all closed frequent itemsets are maximal frequent itemsets.

1.3.5 Emerging pattern mining

The three types of mining mentioned earlier extracts patterns from a single class of data. They can be used to find combinations of items appear freqeuntly in the database. However, such patterns are rarely discriminant and not suitable for classification. *Contrast pattern mining* is used for extracting patterns from a database containing multi-classes of instances. These patterns are interesting for classification (*e.g.* they appear frequently in one class and not in the others). Several contrast pattern mining techniques have been proposed such as *Class based association rules* [12], *Version spaces* [65], *Odds ratio rules* [59], *Delta discriminative* [49], *MDL based contrast* [98].

However, we focus on the most commonly used contrast pattern mining called *Emerging pattern mining*. Emerging pattern mining introduced by Dong and Li [24] in 1999 is designed for extracting discriminative co-occurrence features which can be used for binary class classification. We consider two classes of data as *positive* and *negative*. A positive emerging pattern is defined as a set of features that appears together frequently in the

positive class and rarely appears in the negative class. In the case of having more than two classes, the class of interest is considered as the positive class and the rest of the classes are considered as the negative class.

There are two important parameters in emerging pattern mining: (i) *growth rate* GR and (ii) the *frequency* fr which has been defined earlier. The positive growth rate GR^+ of a pattern $\mathcal{X}$ is defined as the frequency of $\mathcal{X}$ in the positive data over the frequency of $\mathcal{X}$ in the negative data.

$$GR^+(\mathcal{X}) = \begin{cases} 0 & fr^+(\mathcal{X}) = 0, fr^-(\mathcal{X}) > 0 \\ \infty & fr^+(\mathcal{X}) > 0, fr^-(\mathcal{X}) = 0 \\ \frac{fr^+(\mathcal{X})}{fr^-(\mathcal{X})} & otherwise \end{cases}$$

The overall frequency $fr(\mathcal{X})$ is the frequency of the pattern occurred in the entire dataset.

1.4 Motivations of the thesis

This thesis is mainly motivated by two questions. First, how to use pattern mining to discover feature dependencies and combine them with learning techniques for inferring supervised, semi-supervised, or non-supervised object models. Second, how to fit pattern mining algorithms to some important large scale computer vision tasks where efficiency is critical. We here explain further only the first one (feature dependencies) since the motivation of the second question is already clear by itself.

One big limitation of all current image representations (Section 1.2) is the lack of modeling of features dependencies. The term feature dependencies are such as

- **Co-occurrences.** The co-occurrence of features can be used to for better discrimination. For example, a single semantic feature 'wing' may not be good enough to distinguish between an 'aeroplane' and a 'bird'. However, combining 'wing' and 'metal' features together can make a more powerful discriminative high-order feature to separate the two classes.

- **Relative difference.** The amount of a given visual features in the image, for the same object, can vary accordingly to the size of the object in the image. However, the proportion of a feature with respect to another feature could be more stable. For example, although the amount of 'metal' and 'wheel' in images containing cars are different according to the size of each car, the amount of 'metal' features from the body of the car is larger than the amount of 'wheels' features, in all cases.

- **Spatial relationships.** Similar to the relative difference between the amount of two features, the relationship between the location of two features could be also very

useful. For example, the 'wheel' semantic feature of a car is always under the 'metal' body part.

Feature dependencies can be very be important for discriminating between images. However, it is very difficult to encode such information into image representations. The fact that images have high number of features (typically in a scale of thousands), it is not feasible to evaluate and to encode all possible combinations. To give an example, even the simplest case to find co-occurrence of two features, if an image representation contains 1,000 features, about 500,000 feature pairs have to be evaluated. How to step beyond pairs and encode dependencies among larger sets of features remains a very challenging open problem which we would like to address in the thesis.

1.5 Thesis organization and contributions

Chapter 2

We make a preliminary investigation to obtain some prospective directions for encoding feature dependencies. We first introduce three types of feature dependencies including (i) *co-occurrences*, (ii) *relative differences*, and (iii) *spatial relations*. Starting from a typical image representation which is a real valued feature vector, we design three methods for capturing each type of feature dependencies resulting in three high-order image representation. The representations are compared in an image classification framework.

The main problem of capturing feature dependencies is the dimensionality of the feature combination space. We propose some strategies to select a subset of the whole space including *random selection* and a more sophisticate technique based on *Emerging Pattern mining*.

In order to capture feature co-occurrence, the real value of each feature in the image feature vectors has to be binarized (*i.e.* whether the feature is present or absent). We discuss the issues raised by the binarization schemes. To address them, we introduced the *top-K bins* binarization which is to threshold each vector to the top-K highest value of the feature vector.

In the experimental section of this chapter, we do further experiments on the second introduced method which captures feature relative differences, since promising results are obtained from the initial set of experiments. We validate that encoding relative differences between the features of BoW feature pairs, *pairs differences*, can give better classification score that the original BoW representation. At the conclusion of the chapter, we discuss the problems encountered and directions for further improvement which are useful in the following chapters.

Chapter 3

We propose a technique to *re-rank* images retrieved from text-based search engines. We introduce the problem of image re-ranking and two main prior assumptions that are typically used for image re-ranking including (i) text based search engines provided initially reasonable results in which the top returned images are likely to be relevant to the query, and (ii) negative images are scattered (*i.e.* they do not make clusters, unlike the relevant images). We explain why *closed frequent itemset* has the potential to be used in this re-ranking task.

In order to mine frequent itemsets, it is necessary to convert image representations to data mining transactions. To obtain transactions, we use the top-K bins binarization method discussed in Chapter 2 for binarizaring the *Bag of Words (BoW)* representation. Our re-ranking scoring function is based on the statistics of frequent itemsets found in the images. We also discuss about several *weighting schemes* of the patterns, based on their (i) *frequency*, (ii) *length*, (iii) *area*, and (iv) *original ranking positions*.

Chapter 4

We solve major problems from our preliminary investigation in Chapter 2. We discover a method that can capture both *feature co-occurances* and *feature reletive differences*. Our novel image representation is based on *Histograms of Emerging Jumping Patterns (HJEPs)* and *multiple random projections*. Applying the top-K binarization method on the full original BoW representation leads to a huge amount of information loss. To overcome the problem, we propose to project the original BoW representation into multiple low-dimension sub-spaces before apply top-K binarization method. The emerging patterns mined from each projections are accumulated and the accumulated value is considered as a single feature of the proposed image representation. The features from all random projections are concatenated as the final image representation. We discuss several advantages of the representation including (i) its simplicity and (ii) the fact that no post processing is required for pattern selection. The representation has been validated on several image classification benchmarks and show state of the art results.

Not only for image classification, similar techniques in this chapter can be applied to image re-ranking as well. We show better results in image re-ranking compare to the results in Chapter 3.

Chapter 5

We address the problem of *detecting groups of duplicates* in large-scale unstructured image datasets, such as the Internet, based on closed frequent itemsets. Duplicate images are

defined as the ones 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 of brightness/contrast. We propose a very compact image representation based on the top-K bins binarization of BoW representation and consider the representation as a pattern mining transaction. We show that using only 10 items per image is sufficient to find the duplicates. Our idea is to extract *long closed frequent itemsets* as the length of the itemsets is related to the similarity of the image transactions containing the itemsets. We proposed a new dataset of one million Internet images obtained with random searches on Google image search and validated the efficiency and effectiveness of our approach.

Chapter 6

We summarize the content and contributions of each chapter by discussing the results along with some possible directions for further work.

Chapter 2

Preliminary Investigations

Contents

2.1 Introduction 25
2.2 Related work 26
2.3 Three types of feature dependencies 27
2.3.1 Co-occuring features 28
2.3.2 Relative difference between features 32
2.3.3 The spatial relation between features 35
2.3.4 Comparison of the three feature dependencies. 37
2.4 Experiments 37
2.4.1 Pascal VOC 2007 dataset 37
2.4.2 Size of vocabulary 38
2.4.3 Results on the full dataset 39
2.5 Conclusions 40

2.1 Introduction

As mentioned in the introduction of the thesis (Section 1.4), one aim in this study is to improve current typical image representations by including information related to feature dependencies. In this chapter, we investigate the possibilities to capture three types of feature dependencies including: (i) feature co-occurrences, (ii) feature relative differences, and (iii) feature spatial relations. These different types of feature dependencies are evaluated in an image classification framework.

The chapter begins by providing some related work in encoding feature dependencies in Section 2.2. The three aforementioned methods to capture dependencies are given in Section 2.3. For each method, several small-size experiments are carried out. We explain

the motivations and the implementation details of each experiment. Results showing the influence of the different parameters and the different encoding strategies are also given. We discovered that encoding pair-difference features has the potential to outperform our baseline features. Therefore, we make further experiment on this encoding in Section 2.4. Finally, Section 2.5 concludes the chapter by summarizing the three investigated methods, discussing the encountered problems, and pointing out some potential directions for further investigation in the next chapters.

2.2 Related work

The simplest way to capture feature dependencies is to encode relationships between pairs of features [56, 61, 62, 66, 67]. Most of these methods begin by extracting local features and map them to visual words. The statistics of co-occurrences of visual word pairs which appear closely together in images are used as new image features [56, 61, 62]. As the number of visual pairs grows quadratically to the number of visual words, the representations can have very high dimensions (*e.g.* 500,000 pairs can be constructed from 1,000 visual words). Due to the high dimensionality, not only training classifiers can be very slow , but also the classifiers may over-fit and fail to generalize over testing data since the number of training images is usually much less than feature pairs. Some feature selection methods [61, 62] have been applied to reduce the dimensionality. Unfortunately, the selection methods require additional information such as class labels, and performance is not significantly improved. Morioka and Satoh [66] have proposed *Local Pairwise Codebook (LPC)* to solve the problem of high-dimensional feature pairs. The method is to concatenate pairs of spatially close local features before constructing a codebook of pair of visual words by k-means clustering. Pairs of local features are mapped to pair of visual words before a histogram of pair of visual words is computed and used as the image representation. Since pairs are combined at the local features stage, the dimension of the final representation can be controlled by the codebook size. In LPC, the local features are simply concatenated without encoding more relative spatial information. An extension of this work has incorporated spatial directions between the local features before constructing the codebook [67]. The space of combinations grows dramatically when considering more than simple combinations of two features. Going beyond pairs, [62] constructed triplets, by progressively selecting the best features at each iteration and extend them to the next interation. However, the triplets does not show improvement over the pairs. Some important combinations cannot be discovered as some combinations are discarded at early stages.

Although the space of feature combinations is very large, not all combinations are useful. The problem of encoding feature combinations can be stated as how to efficiently select the best small set of feature combinations. Interestingly, pattern mining allows the dis-

covering of the relevant combinations of items in transaction databases without having to evaluate exhaustively all possible combinations. The use of pattern mining to find combinations of visual feature has attracted several works [33, 39, 80, 102, 108]. The idea of most of these works also begins by extracting local features and map them to visual words. For an image, several lists of visual words are constructed in which each list contains a few visual words which are located closely together. The lists can be seen as data mining transactions and pattern mining algorithms can be used to find interesting visual word combinations.

Several papers then use *frequent itemsets* (*e.g.* [33, 39, 80, 108]) to find sets of features that are often located closely together. The whole set of frequent itemsets obtained are typically huge (several millions) but only a few of them have discriminative power which can be use for image classification. Therefore, it is necessary to select and use only the frequent itemsets that are good at discriminating the classes. The selection is based on contrast measures [71]. The total number of the remaining discriminative patterns are still usually too large and most of the them are redundant to each other. Fernando et al. [33] proposed algorithms to select frequent itemsets that are discriminative, representative, and non-redundant and called the remaining set of the frequent itemsets as *Frequent Local Histograms (FLH)*. Constructing histograms over FLH gives impressive classification score. Currently there is no efficient way to generate a set of patterns that can satisfy global constraints (*e.g.* cover all data while gives good prediction accuracy) [10] and all of the mentioned methods require costly or ad-hoc post-processing stages for selecting the frequent itemsets.

While most of work have tried to capture dependencies (co-occurrences and spatial relation) of between visual words located closely together, Barat et al. [3] show that the dependencies between the global BoW features also contains valuable information. Rather than representing each image in the typical form of BoW representation, each image is represented as a string in which the alphabets and order corresponds the *TF-IDF* weighting BoW histogram. To find distance between the image representation, they proposed a new distance measure based on *Edit distance* and TF-IDF weights. The image representations are based on the ranking of visual words. Therefore, in this work the relative difference dependencies between features are captured.

2.3 Three types of feature dependencies

In this section, we investigate three methods for encoding three types of feature dependencies: (i) co-occuring, (ii) relative differences, and (iii) spatial relation. We make some preliminary experimental validation, on a small part of the *Pascal VOC 2007* dataset. The full dataset consists of 9,963 images of 20 classes objects and is split into three subsets including: (i) the train set, (ii) the validation set, and (iii) the test set. Section 2.4.1

describes the full dataset in more detail with sample images. In this section, we use only the train set containing 2,501 images and the validation set containing 2,499 images. Moreover, we only evaluate the results on a single class which is the 'aeroplane' class. More details on the dataset are given Section 2.4.1.

More precisely, we consider a binary classification task in which all the images containing at least one 'aeroplane' are a positive examples whereas other images are negative examples. In these experiments, we encode feature dependencies and use them as new image features. Our baseline feature is the standard Bag of Words (BoW) histogram representation which is one of the most commonly used image representation [42, 84, 99]. Therefore, the term feature dependencies are the dependencies between visual words. To calculate visual words, we used (i) multi-scale sift as local descriptors from VLFeat library [95] (8 scales ranging from 4 up to 18 pixels with the default patch step size of 2 pixels.) and, (ii) a dictionary size of $1,000$. All representations which are histograms, including the baseline BoW, are L1-normalized and square-rooted.

In all experiments of this section, we used the train set for training binary linear SVMs and the validation set for testing the results. The *Average Precision (AP)* is used as the metric to evaluate the different image representations.

2.3.1 Co-occuring features

In this method, we try to extract sets of co-occurring features. Each of these sets contain some features in which they frequently appear together.

The term co-occurrence in this case only considers that the features are present together and does not take into account the amount of each individual feature. Indeed, the data mining tools we want to use need binary items as inputs. Since typical image representations encode the amount of each feature (e.g. by histograms), a binarization process is required to determine the present or absent of the features.

If the amount of a feature is beyond a certain limit, the feature is considered as present. Once each image is described as a list of present binary features in which we refer to *visual attributes*, we can apply pattern mining techniques to extract sets of co-occurrence features. There are several pattern mining techniques using different constraints. In this chapter, we focus on a particular type of pattern mining called *emerging pattern mining* introduced in Section 1.3.1 which is suitable for classification tasks.

Binary visual attributes representation

Obviously, binarization causes a dramatic loss of information, and the best way to binarize a real valued vector is still an open question (and is probably context dependent). We

investigate several methods to binarize real valued features in particular to image feature vectors which have real values. To illustrate the methods, we use the the Bag of Words (BoW) histogram representation.

In the BoW representation, each image I is described as a histogram of visual words $\boldsymbol{h} = (p(w_0|I), \dots p(w_d|I))$ where d is the size of the dictionary. We binarize the histogram by thresholding the values of each histogram bin. The binarize vector is described as $h_i^b = 1 \iff h_i \geq \tau$, where τ is the binarization threshold. Options for setting τ are described along with the explanation of the advantages and disadvantages of each method in the following paragraphs.

Fixed threshold. The simplest way is to set a fixed constant global threshold $\tau = C$ (Figure 2.1a). The disadvantage of this method is that the images which have flat distributions can be entirely set to 'zeros' or 'ones' (Figure 2.1a-left) depending on the threshold value. Besides having a single threshold for all features, an alternative is to set distinct thresholds for 'each' bin/feature. For example, the threshold for each features, can be set according to its mean value $\bar{f}_i$ or its median value $\tilde{f}_i$. However, the number of attributes after binarization (*i.e.* the number of ones) can vary a lot from an image to the another. This gives a problem when extracting patterns since the images with more items will contain more patterns than the images having less items.

First q-quantile of the sorted histogram. Another method is to set the threshold to the first q-quantile of the sorted histogram $\boldsymbol{h}^s$ (Figure 2.1b). We denote $\boldsymbol{h}^s$ as a vector in which its components correspond to the bins of $\boldsymbol{h}$ sorted in descending order. Then $\tau = h_{i^*}^s$, where i^* verifies $\sum_{i \leq i^*-1} h_i^s < \frac{1}{q} \wedge \sum_{i \leq i^*} h_i^s \geq \frac{1}{q}$. This binarization method seems to be a suitable choice since the total amount of bits being set to one comes from a fix proportion of the total probability distribution of the original histogram. Unlike the fixed threshold method, this thresholding ensures that at least one bit is set to one (an image has at least one attribute), and there will not be any case where all the bits are set to one (an image contains all attributes). This binarization method can avoid having too many or too less items in flat distributions (Figure 2.1b-left) as could occur when using a fixed threshold (Figure 2.1a-left), However, a problem remains when only a few visual words dominate the image. We give here an example of an image of a small plane flying in the sky. This image contains a large amount of a single non-gradient visual word. The amount of this visual word exceeds the first q-quantile which sets only the bit corresponding to this visual word to one. In this case, the visual words representing the plane are discarded. Having only one attribute, the combinations of attributes in this image is limited. Figure 2.1b-right also explains this problem.

Binarization methods	Average Precision (%)
Fixed threshold	19.0
First q-quantile	7.2
top-K bins	27.0

Table 2.1: Comparison of different binarization methods.

Top-K bins. This methods set τ to the top k most frequent visual word value, $\tau = h_k^s$ (Figure 2.1c). Selecting the visual words which have the highest counts is favorable since they are the most representative features in the image. Furthermore, thresholding to the k-highest value allows all binary vectors to have the same l_0-norm that is all image have the same number of attributes. Unlike the two previously described methods, this binarization ensures that all images will have a reasonable amount of attributes. This can avoid the problems of having too high or too low number of attributes as mentioned in the previous examples. Having a fixed number of attributes also gives an advantage to the implementation. For example, we can work on fix array sizes which is much more efficient in terms of both computation and memory (the exact memory usage is known in advance).

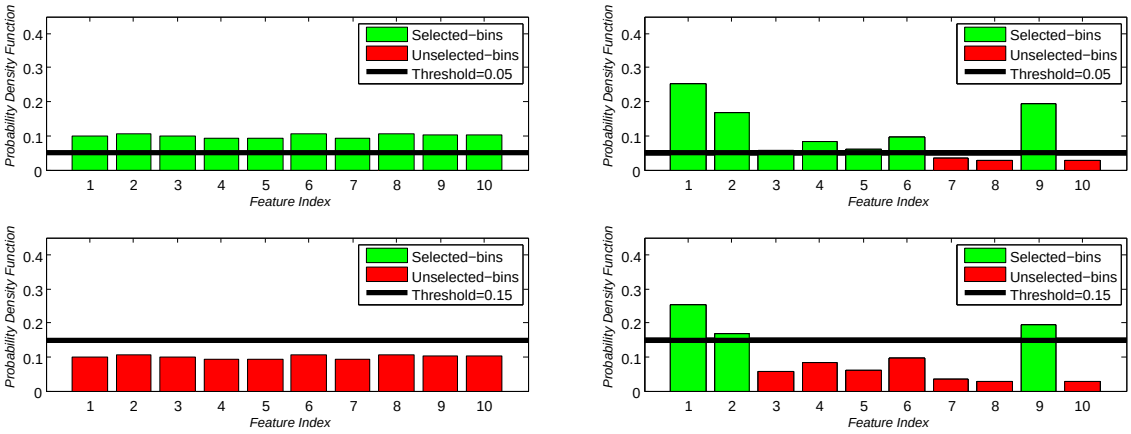
Pattern encoding and classification

After the mining process, the mined patterns are used as new (high-order) features to represent images. Each image is represented as a binary vector having the dimensionality equal to the number of patterns N_p. Each bits corresponds to the presence or absence of one given pattern.

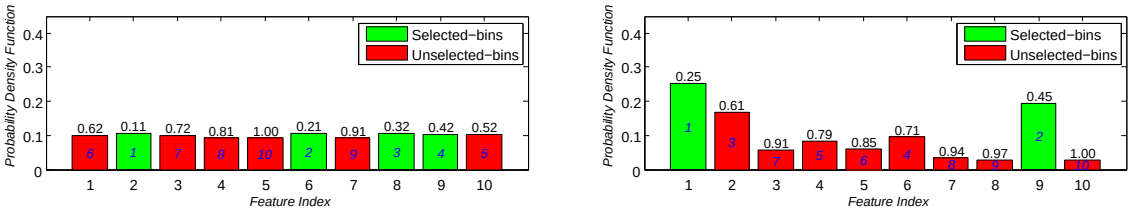
Experiments

Binarization methods evaluation From the BoW histogram representation, we converted histograms into pattern mining transactions by using the three different aforementioned binarizations. For all of the three binarizations, we adjust their binarization thresholds in order to have the same proportion of ones (about 20 percents of the features). We mine out the top 5,000 patterns according to the *growth rate* GR and represent each image by the patterns it contains. Each image is therefore represented as a binary vector of 5,000 dimensions. Using the top-K bins binarization method gives the best results as shown in Table 2.1. Consequently, we are going to use the top-K bins binarization in the remaining chapters of the thesis.

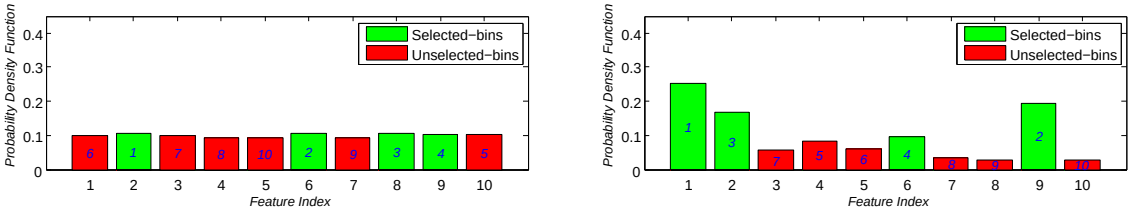
Number of patterns This experiment shows the trend of performance according to the number of patterns. The performance increases according to the number of patterns and



(a) Fixed threshold binarization.The bins having values above the threshold (black straigt line) are selected as transaction items. Problems occurs in flat distributions where all bins are selected (top-left) or none of the bins is selected (bottom-left).



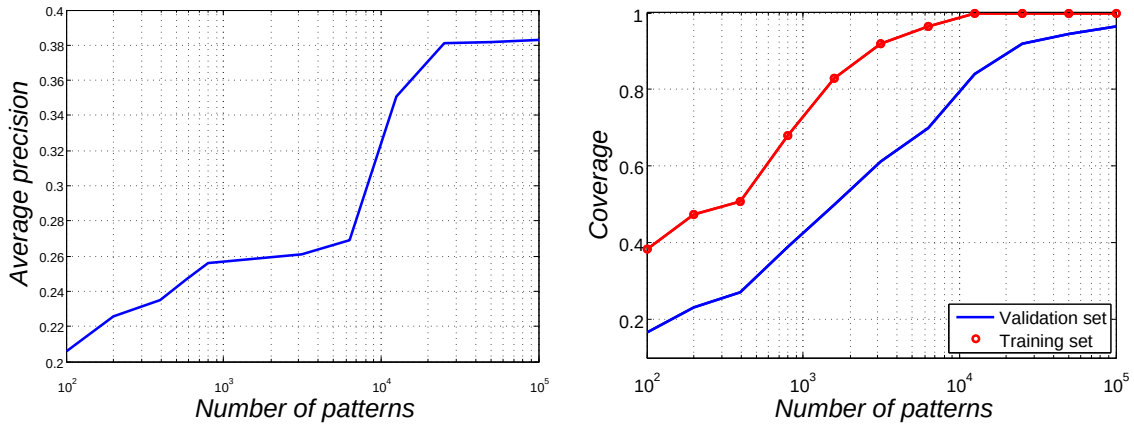
(b) First q-quantile of the sorted histogram binarization. The probability is accumulated starting from the bin with the highest probability to the bin with the lowest probability. (The blue-italic numbers in the bins show the accumulating order, the black numbers on top of the bins show the accumulate probability). The bins having probability up to the first q-quantile of the sorted values are selected as transaction items (in the figures q is set to be 2.5). The first q-quantile is the data value where the cumulative distribution function crosses 1/q. Only few bins are selected if there are high peaks in the distributions (right).



(c) top-K bins binarization. The bins which have values among the top-K values are selected as transaction items (The blue numbers in the bins show the ranking order, K is set to 4). This binarization gives equally a reasonable amount of items for all images (both left and right have 4 items).

Figure 2.1: Toy example of the three types of binarization methods. All figures on the left represent the binarizations of a flat distribution histogram. All figures on the right represent the binarizations of a distribution histogram with high peaks. The green bars are the bins in which their feature indexes are used as transaction items. The red bars are the discared bins. For example, the transaction encoded from the histogram in Figure (c)-left is {2,6,8,9}.

saturates at about 38% AP as shown in Figure 2.2. We can notice from Figure 2.2(a) that increasing the number of patterns from 25,000 to 100,000 does not improve the result. The plot in Figure 2.2(b) shows the coverage of the positive images in the training and validation set according to the number of patterns. The coverage is the percentage of the images having at least one pattern over the total number of images. We can see



(a) Average Precision. (b) Coverage.

Figure 2.2: Average Precision and coverage according to the number of patterns.

from the red curve in Figure 2.2(b) that the coverage in the training set increases rapidly from 100 to 10,000 and cover all of the training set after that. This means that all of the positive training images have been used when extracting patterns the top 10,000 patterns. After that, the patterns are extracted from the same images. The blue curve shows the coverage in the validation set which is lower than the training set. This shows that the patterns extracted from the training set do not generalize well in the validation set. The coverage in the validation set increases rapidly from 100 to 10,000 which has the same trend as the coverage in the training set. After 10,000 patterns to 100,000 patterns the coverage increases less than 20% of the images. Since most of the patterns from 10,000 to 100,000 are extracted from the same images, redundancy occurs. This can explain why the performance starts to saturates after 10,000 patterns because of the redundant patterns give no additional information.

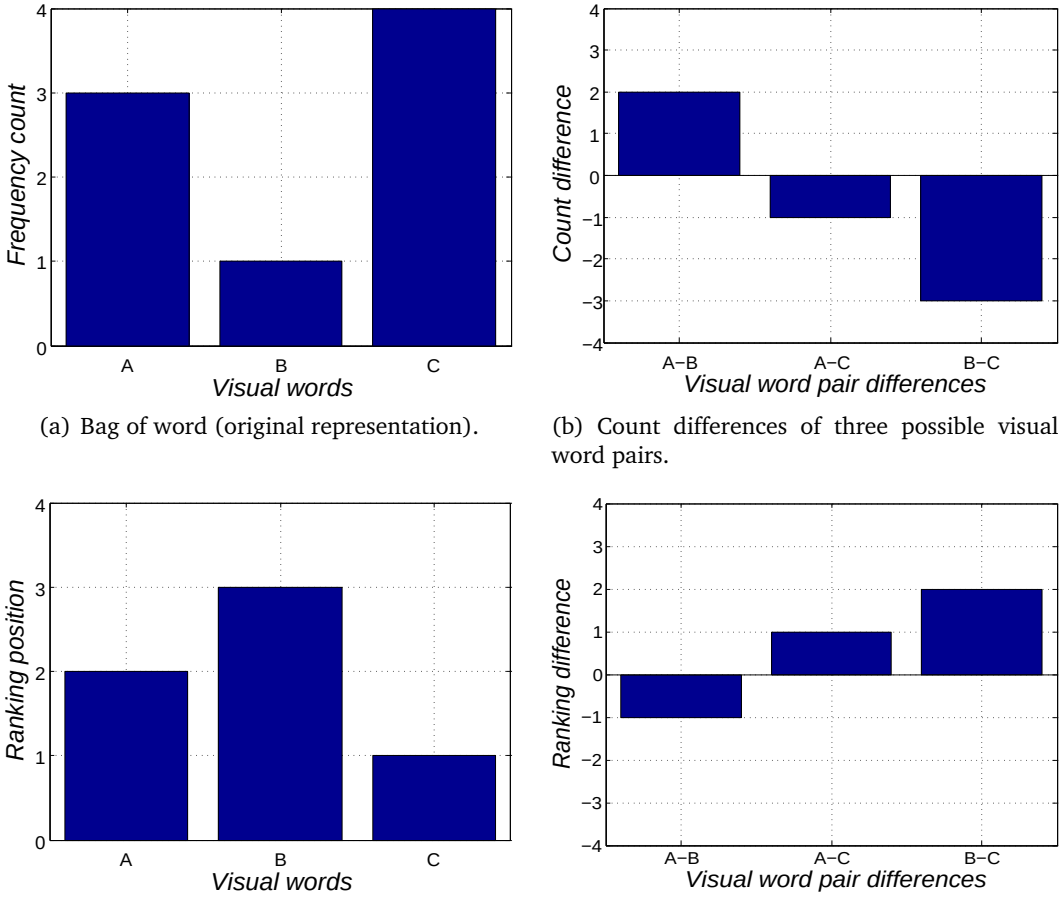
2.3.2 Relative difference between features

In this method, we encode the relative difference between each pair of visual words features. The information is extracted from the global image representation, BoW histograms. The method is described as follows. For each BoW histogram, we calculate the relative difference between all possible pair of features. The difference of each pair is used as a high-order image feature. Each image is represented as a vector with the dimension of all possible pair features.

Experiments

Pair differences The aim of this experiment is to compare two types of pair differences: (i) the different between the bin counts of the pairs shown in Figure 2.3(b) and (ii) the

different between the ranking position of the pairs shown in Figure 2.3(d). The experi-



(a) Bag of word (original representation).

(b) Count differences of three possible visual word pairs.

(c) Visual word ranking.

(d) Ranking differences of three possible Visual word pairs.

Figure 2.3: A toy example to encode a BoW vector 3,1,4 to pair difference representations.

ment setting is as follows. The original image representation is the Bag of Words features with 1,000 visual words. We calculate the pair differences for all possible pair of features (*i.e.* visual words) and use them as new features. The new image representation has 499,500 dimension features. We use a sigmoid function to convert the range of feature values to be between zero and one. The high-order feature vectors are L1-normalized and square-rooted. As shown from 2.2, the pair features using the count differences and ranking position differences give similar performance. Both representations outperform the baseline Bag of Words features by 1 percent. The one based on ranking is less interesting as it requires more computation (to rank the features).

Pair selection The dimensionality of the pair representation is about $\frac{D^2}{2}$ where D is the dimensionality of the original representation. Typical image representations have several thousand features which can give several millions possible pair features. To use all of possible pair features is not practical since the image representation requires too much

Features	Average Precision (%)
Bag of Words	58.1
Pairs of count differences	59.1
Pairs of ranking differences	59.0

Table 2.2: Comparison of pair difference features.

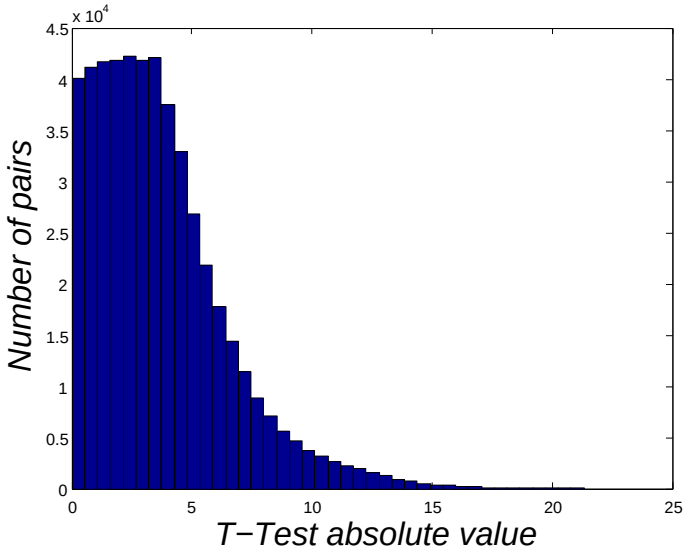
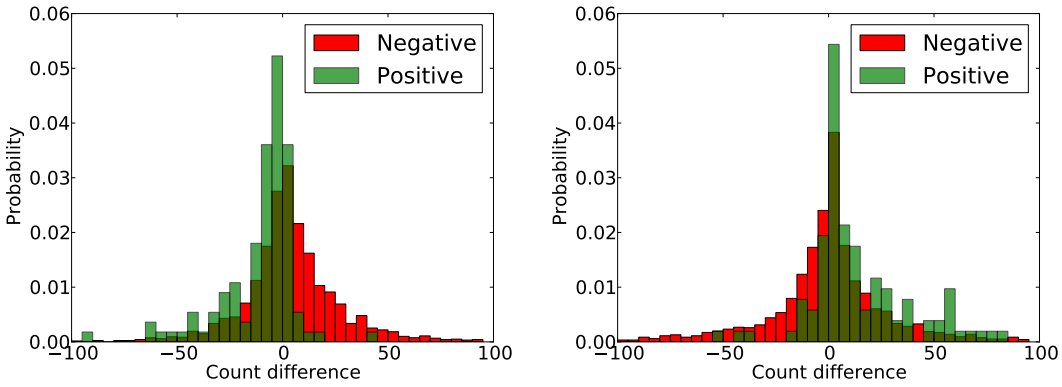


Figure 2.4: T-Test distribution

space also it will be too slow for training a classifier. In this set of experiments, our aim is to reduce the number of feature pairs.

Two different methods for selecting the feature pairs has been investigated including: (i) random selection and (ii) selection based on *t-test*. In the first strategy, we randomly choose some pair features from the total amount of 499,500 feature pairs of count differences which was computed in the previous experiment. In the second strategy, for each pair, we calculate the t-test from the distributions of the count differences over the positive and negative training images. Figure 2.4 shows the distribution of t-test. The pairs which gives high absolute t-test means that the over-lapping between the positive and negative distributions are less that the pairs which gives lower t-test. In other words, the pairs which gives high t-test can better discriminate the positive and negative images. Figure 2.5 shows the distributions of the pair which gives the highest absolute t-test and the pair which gives the lowest absolute t-test. We can see that overlapping between the positive and negative distribution calculated from the highest absolute t-test (Figure 2.5a) is less than the ones calculated from the lowest absolute t-test (Figure 2.5b). In order to evaluate the performance of the two pair selection methods, we compute the average precision for both methods using different number of pairs ranging from 10 to 499,500 (the total amount of all possible pairs) using a logarithmic scale. From the result shown Figure 2.6, using the top 10 highest t-tests pairs gives better result than randomly select-



(a) The pair which gives the highest absolute t-test.

(b) The pair which gives the lowest absolute t-test.

Figure 2.5: Positive and negative image probability density function over the count difference.

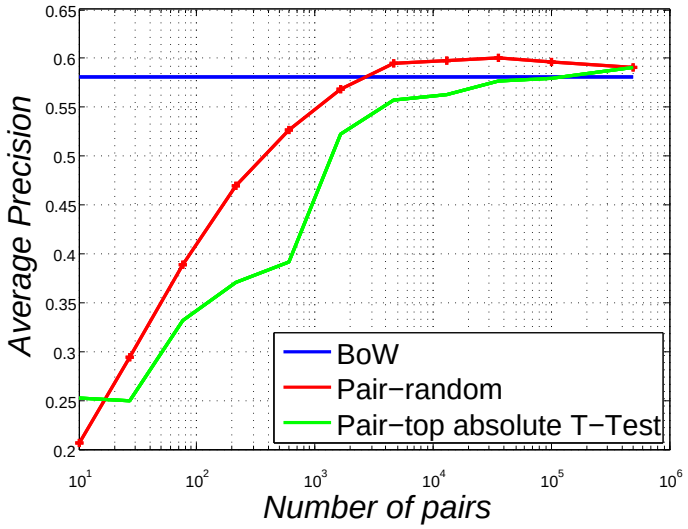


Figure 2.6: Performance according to the number of pairs.

ing the pairs. However, the random selection outperforms the t-test based selection for larger amount of pairs. The explanation could be that the pairs having similar t-tests gives similar information which lies in a small part of the data. On the other hand, randomly selecting the pairs up to a certain amount can better represent the the information from the entire dataset. From Figure 2.6, using about 4,000 random pair features, the optimum performance which is even slightly better than using all the possible pairs is reached.

2.3.3 The spatial relation between features

In this method, our aim is to capture the spatial relation between visual words. For each image, SIFT local features are sampled densely with a fixed step size and are quantized to visual words. The images are now transformed to images of visual words in such a way that each cell of the grid is represented by a visual word. Inspired by [102], for each

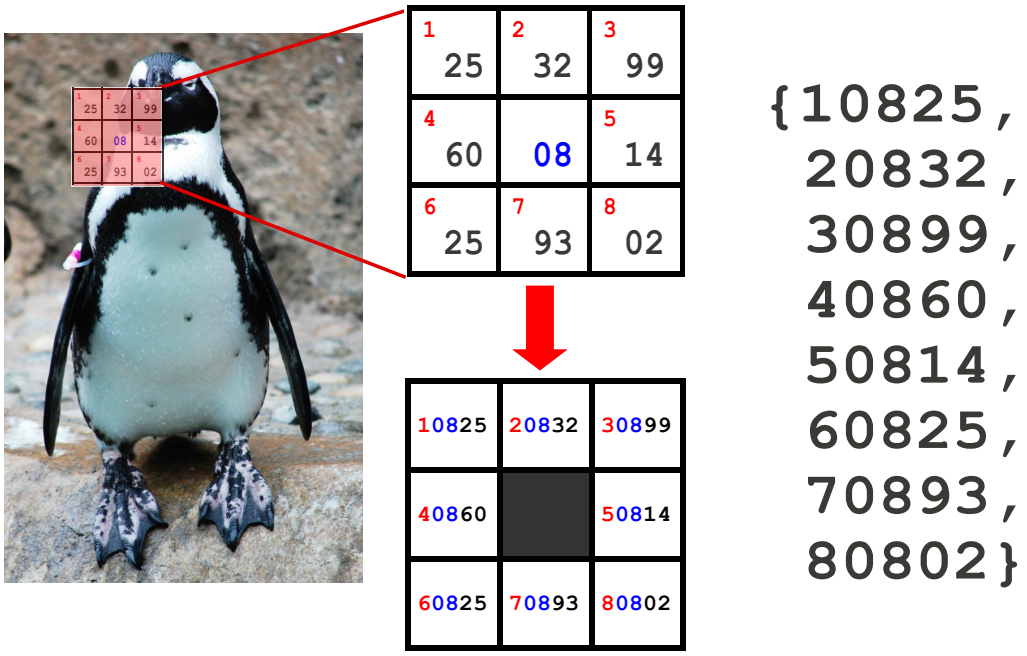


Figure 2.7: Local region filter. One data mining transaction consists of eight data mining items. Each item consists of three components: (i) the location of the neighboring visual word (red), (ii) the center visual word (blue), and (iii) the neighboring visual word (black).

image, we densely slide a a window of 3x3 cells and each window position we encode one data mining transaction consisting of eight items. Each item is formed by two visual words and the spatial relationship between them. The first visual word is the visual word located in the center of the window while the second visual word is one of its eight neighboring visual word. The spatial location is the cell position (encoded by an integer) of the neighboring visual word (Figure 2.7).

Each image will have multiple transactions in which the number of the transactions corresponds to the number of the window location. All transactions extracted from each image have the same label which corresponds to the class of the image. We apply emerging pattern mining algorithm to find interesting combinations of these items.

The transaction database contains all transactions encoded from the training images. Once the emerging patterns are extracted, we represent each images in the training and testing set, as a P dimensional histograms of patterns. P equals to the number of the pattern extracted. A single pattern can appear multiple times in each image, each histogram bin is the count of each pattern in the image.

Experiments

We used 1,000 visual words and encode the transactions as described earlier. We keep the top 10,000 emerging patterns according to the growth rate between the positive and

negative class. The result is obtained in Table 2.3.

Features	Average Precision (%)
Bag of Words	58.1
Spatial relation patterns	8.2

Table 2.3: Comparison of pair difference features.

Since we use 1,000 visual words, we realize that our way of encoding item, the set of all possible items is in a scale of millions. Due to this very large space, it is very difficult to extract meaningful combinations of these items.

2.3.4 Comparison of the three feature dependencies.

Table show the results of the three types of feature dependencies using their best settings on the aeroplane class validation set.

Table 2.4: Image classification results on the 'aeroplane' class validation set.

Features	Average Precision %
Bag of word (baseline)	58.1
Co-occurrences	38.1
Relative differences	60.0
Spatial relation	8.2

2.4 Experiments

In Section 2.3.2, we showed that encoding the relative difference between pairs of features has the potential to outperform the original Bag of Words representation. Previously we have validated the approach using only a small subset of the **PASCAL 2007 dataset** which is the validation set of a single image class, the 'aeroplane' class. In this section, we do further experiments all the classes of this dataset and compare the results with the related literature.

2.4.1 Pascal VOC 2007 dataset

The PASCAL 2007 dataset [27] is one of the most popular dataset used for benchmarking image classification algorithms. It consists of 9,963 images of 20 classes objects including 'aeroplane', 'bicycle', 'bird', 'boat', 'bottle', 'bus', 'car', 'cat', 'chair', 'cow', 'dinning table', 'dog', 'horse' 'motorbike', 'person', 'potted plant', 'sheep', 'sofa', 'train', and 'tv monitor'. The images are daily photos collected from *Flicker*. The large intra-class variations from

Figure 2.8: Example images from the Pascal VOC 2007 dataset.

different object sizes, view points and illuminations makes the dataset very challenging. Figure 2.8 shows some images from the dataset. For benchmarking purpose the dataset is split into three subsets including the train set, the validation set, and the test set. The number of images of each subset is shown Table 2.5. The evaluation protocol is to perform a binary classification task for each of the 20 object classes. For each task the images containing the object of interest is considered as positive samples whereas the rest of the images are considered as negative samples. The results are reported on the test set for each object class using average precision as the scoring metric. To summarize the overall results, the mean over the 20 average precision (mAP) scores is computed.

Train	Validation	Test
2,501	2,499	4,952

Table 2.5: Number of images in each subset.

2.4.2 Size of vocabulary

In this experiment, we would like to measure the effects of constructing pairs from different sizes of vocabulary. Besides 1,000 dimensions, we have evaluated one lower dimension of 100 words and one higher dimension of 4,000 words. In order to have a comparison between the results given Section 2.3, we use the same baseline features which is the Bag of Word features with a vocabulary size of 1,000 dimensions. We still evaluate on the 'aeroplane' class and use only the 'train' and 'validation' data.

Similar to the experiment of Section 2.3.2, we compute the average precision using different number of random pairs. The result is shown in Figure 2.9. The BoW feature of 100 dimensions gives a performance of 49.5% AP. The green circle curve shows that when con-

structing about 1,000 random pair features, comparable results to the BoW feature of 100 dimensions can be obtained. The BoW feature of 4,000 dimensions gives a performance of 59.5% AP which is about 1.5% higher than the BoW feature of 1,000 dimensions used previously as our baseline in Section 2.3. It has been shown that the performance of BoW feature increases according to the number of vocabularies [18]. When constructing pairs from the BoW feature of size 1,000 dimensions (red curve with crosses), the performance saturates at about 4,000 pairs. However, when constructing pairs from the BoW feature of size 4,000 dimensions (blue curve), the performance still does not saturate even when using 50,000 pairs. In this experiment, we have evaluated up to only 50,000 pairs and achieve a performance of 62.2 which is 2.7% higher than the original 4,000 dimension BoW feature.

It has been shown in [18], that increasing the vocabulary size from 4,000 to 25,000 can increase the performance up to 4%. However, the total number of possible pairs will be about 300 millions. In order to use our method to increase the performance of the higher dimension features, it will require a very large number of random pairs which will not be tractable.

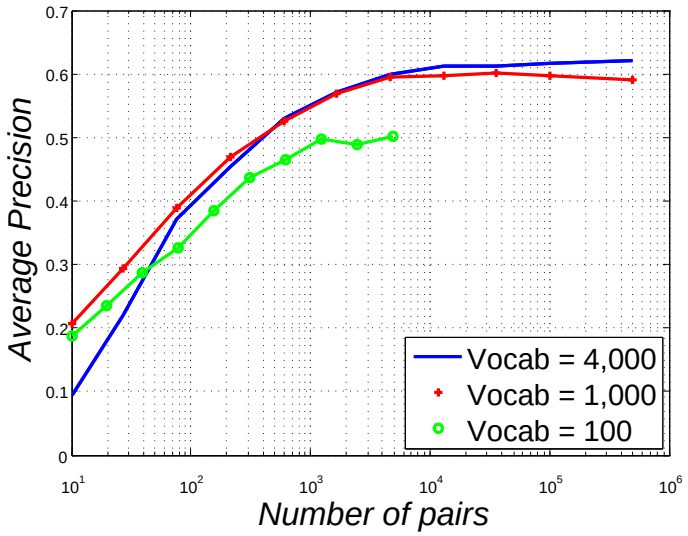


Figure 2.9: Performance according to number of pair features.

2.4.3 Results on the full dataset

In this experiment, we evaluate the random pair features on the whole dataset. The result in Table 2.6 is the results of each class on VOC 2007 test set. The results in Table 2.6 is split into two parts including the results without and with Spatial Pyramid Matching.

The upper part shows the results of the BoW features without spatial pyramid matching. The BoW is computed using our own implementation. The size of the vocabulary is 4,000 dimension and we use 10,000 random pair features. The pair features give 0.9% mAP

improvement over the BoW and when combine the scores the performance increases up to 1.6%.

The lower part in the table shows the results of BoW features with Spatial Pyramid Matching. The BoW feature is the same as used in [18] with a vocabulary size of 4,000 and 8 spatial regions obtained by dividing the image in 1×1, 3×1 (three horizontal stripes), 2×2 (four quadrants). The total dimension of the feature is 32,000. In our settings, we encode 10,000 random pair features in each spatial regions and concatenate all pair features. The total dimension of the pair feature is 80,000. The pair feature gives equivalent mAP as the BoW features and when combining the pair features with the BoW features an improvement of 0.8% mAP. The reason why the pair features does not give improvement unlike the improvement over the BoW feature without Spatial Pyramid Matching could be that when an image is divided into small sub regions, many visual words in the sub-regions are not presented and therefore many pair features are not informative.

Feature	mAP	plane	bird	bike	bottle	boat	bus	cat	car	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
without Spatial Pyramid Matching																					
BoW	44.9	63.9	32.3	48.0	**17.4**	55.2	43.7	44.8	65.3	**47.3**	**34.9**	32.2	34.1	65.5	50.9	77.5	17.3	33.5	36.4	55.8	41.4
Pair	45.8	64.1	33.0	48.2	14.2	59.7	44.0	46.0	**67.0**	46.4	31.1	32.4	36.2	**68.6**	51.4	75.9	**18.3**	**38.9**	38.9	**60.9**	40.2
Bow+Pair	**46.5**	**65.5**	**33.9**	**48.9**	16.2	**60.1**	**45.5**	**46.4**	66.7	47.0	33.7	**33.7**	**37.8**	68.4	**52.8**	**77.5**	18.2	36.7	**39.2**	59.6	**42.3**
wtih Spatial Pyramid Matching																					
[18]	52.7	68.2	**40.5**	54.2	24.9	65.3	58.6	54.1	72.2	49.7	39.1	45.4	36.8	73.3	60.3	80.0	26.2	42.8	**46.9**	70.6	45.4
Pair	52.7	67.6	37.9	55.9	22.9	65.0	58.5	54.4	**73.2**	49.5	40.7	43.9	**40.1**	71.4	**62.0**	80.2	26.2	**44.2**	42.3	69.5	**47.7**
[18]+Pair	**53.5**	**68.8**	40.1	**56.2**	**26.2**	**65.5**	**60.4**	**55.5**	73.2	**50.3**	**40.7**	**45.8**	37.7	**73.5**	61.5	**80.4**	**26.3**	44.1	45.9	**71.1**	47.0

Table 2.6: Full results on PASCAL VOC 2007 dataset.

2.5 Conclusions

In this chapter, we investigated different ways to encode three types of feature dependencies including: (i) feature co-occurrences, (ii) feature relative differences, and (iii) feature spatial relations; by using three different methods for each type of dependencies.

For feature co-occurrences, the method begins with the typical BoW image representation. For each image, its BoW representation is binarized using one of the three binarization schemes including: (i) fixed threshold, (ii) q-quantile, (iii) top-K bins. From our experiments, the top-K approach gives the best performance. After the binarization, the indexes of '1s' in the binary vector can be seen as data mining items while each image can be seen as a data mining transaction. We employed emerging pattern mining to extract discriminative co-occurrence features from the set of positive and negative transactions. The patterns are used as high-order features for training a linear classifier. Although, our method to capture feature co-occurrences does not outperform the baseline, we believe that there are plenty of room for improvement especially to solve the problem of binarization loss.

For feature relative differences, the method also begins with the typical BoW image representation. For each image, the differences between feature pairs are computed and are used as new image features for training a binary linear classifier. Two types of differences were investigated including: (i) histogram bin count, and (ii) ranking position of features. From our experiments the two options give similar results and can slightly improve the baseline. However, when the baseline feature has high dimensionality, the number of possible pairs becomes too large for training classifiers. We investigated possibilities for choosing a subset of the pairs based on: (i) random pair selection, and (ii) t-score. From our experiments, random pair selection gave better results than t-score when, comparing with similar number of selected pairs. The reason is that the pairs selected using t-tests gives similar information which lies only in a small part of the data. In order to outperform the BoW representations, the number of random pairs has to be large respectively to the dimension of BoW. As the number of possible pairs grow quadratically to the number of visual words, random selection might not be the best strategy for feature selection. One possible extension of the work could be to find a method to select only the most interesting pair features. Nevertheless, the results of all 500,000 pair features (BoW features with 1,000 dimensions) can only improve the baseline slightly. Even we find a solution to select feature pairs, the results will be bounded by the slight improvement of using all pairs. Instead of feature selection, this work encourage us to capture relative differences between groups of visual words not only pairs.

For feature spatial relations, the method begins at the stage when local features are mapped to visual words which is the stage before calculating the histogram of visual words in the BoW image representation. For each image, we encode multiple data mining transactions in which each transaction corresponds to a local-region in an image. Each transaction contains 8 visual items correspond to the visual words in the local-region. We employed emerging pattern mining to extract discriminative co-occurrence features from the set of positive and negative transactions. Since an image contains multiple transaction, a single pattern can appear multiple times in an image. An image is represented as a histogram in which the bins correspond to the patterns. We obtained very bad results using this method. We realized that our way of encoding item, the set of all possible items is in a scale of several millions. Due to this very large space, it is very difficult to extract meaningful combinations of these items. Besides the mistake of using high number of items, pattern selection which has been shown to be very important [33] is also missing in our work. Since this method is very similar to several previous works [33, 39, 80, 108], we are not so interested in extending this work in the thesis.

We conclude that feature dependencies contain valuable information and it is worth to further investigate the possibilities to encode them. We propose a method that can capture both feature co-occurrences as well as feature relative differences in Chapter 4. From our observation, we believe that binarize real valued features by using top-K bins binarization scheme has many good properties and will be used in all of the remaining chapters. Prob-

lems encountered during our preliminary investigation include (i) binarization loss, (ii) high dimensionality, (iii) redundant patterns. We show how we overcome these problems in Chapter 4.

Chapter 3

Image Re-ranking

Contents

3.1 Introduction 43
3.2 Related Work 45
3.3 Method 47
3.3.1 Binary representation of visual attributes 47
3.3.2 Closed frequent itemset mining and matrix transposition 47
3.3.3 Re-rank images 48
3.4 Experiments 51
3.4.1 Datasets and evaluation protocol 51
3.4.2 Visual attributes 52
3.4.3 Parameters 52
3.4.4 Closed frequent itemset weights 53
3.4.5 Complexity and computational time. 54
3.4.6 Comparison to other methods 55
3.5 Conclusions 57

3.1 Introduction

Image search has become one key feature of most well-known web search engines such as 'Google', 'Yahoo', 'Bing'. Given a text query to a search engine, millions of images are retrieved based on their metadata such as keywords, tags, and/or descriptions associated with the image. Using tags or keywords is indeed very fast as it can be implemented efficiently using inverted files. However, the metadata do not always correspond to the visual content in the images. For example, when a user searches for images of 'Eiffle' there might be some few non relevant images of restaurants or persons retrieved just because

they are also tagged 'Eiffle'. From this fact, the retrievals are mixed with noisy irrelevant images, which is undesirable.

Normally when a user is searching for images, he/she only looks at only a few tens or hundreds images which are shown first. It is therefore important that the images should be ranked in such that the top retrievals are the most relevant images and the irrelevant images should be ranked at the end. As said that the text data do not always correspond to the visual content in the images, content-based visual information has been used to re-rank the initial text based results from the search engines.

There are two assumptions that are typically used for re-ranking the images. The first assumption is that the text based search provides already a reasonable ranking result that is the top images are containing a majority of images are relevant to the query. The second assumption is that the top relevant images returned by the initial text-based retrieval are visually similar to each other. On the other hand, the few irrelevant images among them, what we call the noise, are unlikely to share visual similarity with the other images. From these assumptions, several techniques can be use to re-rank the images according to the visual similarity of top ranked images. For instance, it is possible to train a classifier by using the top rank images as noisy positive images and the images that are ranked below or even the images from other dataset as negative images. The classifier can then be used to (re)score all the retrievals. Our method mainly also exploits these assumptions, by modeling the connectivity among retrieved images.

Our idea is based on two main components including; (i) *frequent itemset mining*, and (ii) *top-K binary image representation*. Frequent itemset mining can be used to efficiently find sets of items that appear together in many *transactions* in the dataset so-called *frequent itemsets*. Definitions of pattern mining terminologies are described in Section 1.3.1. If we describe each image as a transaction containing a list of *visual attributes* (the bins correspond to '1' in the binary image representation), a frequent itemset in our case will be a set of visual attributes that appear in many images in the database (more precisely in the set of images given by the text based search engine). Note that the images containing the same frequent itemset share some similarity to each other since they contain the same set of visual attributes.

Frequent itemsets are likely to be found in the positive images. Not only because the relevant images are frequent in the dataset but also because they are visually similar (they have similar visual attributes). On the other hand, frequent itemsets are rarely found in the non-relevant images because they do not share similar attributes with other images. After extracting frequent itemsets, our scoring function to re-rank the images is based on the statistics of the frequent itemsets. As mentioned earlier, an image containing a frequent itemset shares some similarity to other images containing the same frequent itemset. Therefore, if an image contains several frequent itemsets, it means that it shares similarity with many groups of images, or, in other words, it is 'connected' to these other

images. From the assumption that the positive images share some visual similarity with some other positive images while negative images are uncorrelated, an image highly connected with other images in the retrievals is likely to be relevant to the query.

This chapter is organized as follows: Section 3.2 discusses previous works related to image re-ranking while Section 3.3 describes our approach in detail including how to encode images as data mining transactions, how to extract frequent patterns, and how to re-rank the images from the frequent itemsets. Section 3.4 provides the experimental results compare to state-of-the-art methods. Finally, Section 3.5 concludes and discusses the possibility to improve the re-ranking approach.

3.2 Related Work

There are several approaches to re-rank the images. They can be based on clustering, such as [6, 8, 50]. Ideally, all relevant images are expected to be visually similar and form a large cluster. The images can then be ranked according to the distance to the cluster center. The irrelevant images should have far distance since they are not supposed to belong to the cluster. However in practice, it is not realistic to assume having a single cluster that can group all relevant images together. Even the relevant images can have large visual variability, for example, side-view bicycles and front-view bicycles look very different to each other. It is possible to have more than one cluster and consider that the images in big clusters are relevant to the query but the difficulty is to determine the number of clusters subject to each query. Moreover, how to compare the relevance between the images in different clusters is an open issue (*e.g.* 'Is an image belonging to a small cluster but close to its center more relevant or less relevant than another image which belongs to a bigger cluster but far from its center?'

Topic models have been used to deal with the visual variation in the relevant images [31, 32, 36]. Brought by the field of natural language processing, two documents may have different sets of words but both of the documents can share the same topic. For example, document $D_1 = \{play, basketball, score\}$ and document $D_2 = \{played, ball, penalty\}$ contain different words but they both are related to sports. Within this framework, each image is basically mapped to a lower dimension of topic space, containing probability values of how likely it belongs to each topic. The images are then ranked on the basis of the dominating topics in the entire retrieval set which means that if an image contains many dominating topics, it is likely to be relevant to the query.

Classification-based approaches have also been used for re-ranking *e.g.* [7, 29, 41]. In this case, a classifier can be trained using the initial top-ranked images as noisy positive images. The trained classifier is used to give a new – and more relevant – score to each image. The problem is that the selected pseudo-positive and pseudo-negative images may

not be truly-positive and truly-negative and can confuse the classification model. Moreover, a new classifier has to be learned for each new query. Krapac et al. [53] proposed a method based on query-relative visual word features to train a single generic classifier that can be used across different queries. It is possible to calculate which visual words are strongly associated with the query set, *i.e.* the majority of the images are the relevant images, the visual words which appear often are the ones strongly associated to the query set. The statistics of the amount of strongly associated visual words can reflect the relevance of the image and can be used as generic features. Thollard and Quénot [89] proposed to combined an unsupervised re-ranking approach with a supervised re-ranking approach. The unsupervised approach is based on the hypothesis that a relevant image is visually similar to some other relevant images, unlike a non-relevant image which does not share similarity between other images. The ranking score of the approach is based on the average distances between K-nearest neighbors. Regarding the supervised re-ranking approach, the idea is to train a single 'junk' classifier to filter out some non-relevant images that are typically found across all queries. These noisy images share some similar characteristics such as they have very small size with less texture (*e.g.* 'icons', 'banners'). The two re-ranking methods also with the original ranking from search engines are complimentary and can improve the overall re-ranking.

Graph-based method performs very well in image re-reranking [43, 48, 101, 63]. A fully connected graph is constructed from a query set in which the nodes are the images and the vertices are the distances between the images. Then a regularization scheme is applied by enforcing the function to be smooth on the graph while keeping the function to be consistent with prior information, *i.e.* the initial text-based ranking. Graph-based approaches have very high computational complexity due to the distance calculation of all image pairs and the computation of the pseudo-inverse of the adjacency matrix.

Frequent itemset mining for removing outliers have been presented in [72]. Each image is described as a pattern mining transaction in which the items are the visual words located on image *interest points*. Frequent itemset mining is applied to find frequent combinations of visual attributes. The frequent itemsets are encoded as new image features. They applied one-class SVM to re-rank the images. This approach is close to our approach. However, one of the main difference is the way images are encoded as transactions. In their approach, images can have different number of visual attributes which is unlike our approach that all images contain the same number of visual attributes. Some non-relevant images might support many frequent itemsets just because have more visual attributes (*i.e.* non-relevant images that are rich in shape and texture). Another main difference is the way the images are ranked. Their approach requires to train a one-class SVM which can be slow if the number of features is large. They have limited the number of frequent itemsets by using a very small codebook size of only 128. Moreover, most of the extracted frequent itemsets are filtered out and only 128 frequent itemsets are remained for training SVM. Compare to our work, we extract frequent itemsets from

very high dimension image features (*i.e.* more than 20 thousand features) and use up to hundred thousand frequent itemsets to rank images. Clearly our approach can capture richer information. This approach has been tested on a 'motorbike' dataset having only 0.02% of non-relevant images. While other approaches have been tested on multiple web query databases with about 20%-30% of non-relevant images. This approach [72] is likely to fail for queries such as 'flags', 'logos' in which the relevant images have less texture and shape.

3.3 Method

3.3.1 Binary representation of visual attributes

In order to extract frequent itemsets from each image query set, the images in the set has to be represented as sets of binary items. Our image representation is a binarized version of the Bag of Words (BoW) histogram, one of the most common approach for representing images [42, 84, 99]. There are several ways to binarize real valued histograms. In our re-ranking approach, we use the *top-K binarization* method. In section 2.3.1, we have shown that this binarization method gives better image classification results than the other two binarization methods (i) Fixed threshold and (ii) First q-quantile of the sorted histogram. Several advantages of the top-K binarization have also been described. The biggest advantage that benefits our image re-ranking approach is that the top-K bins binarization ensures that all images will have the same amount of visual attributes (*i.e.* same number of items).

Having equal number of attributes in each image is important for our image re-ranking approach. Our approach relies on the statistic of the number of frequent itemsets in each image, if images have different number of attributes, images with more attributes (even if they are non-relevant) will have more chance to contain frequent itemsets. The top-K bins binarization prevents this from happening.

3.3.2 Closed frequent itemset mining and matrix transposition

At this stage, we present each image as a list of K binary visual attributes. The terminology used in data mining and computer vision are different. We now refer to the visual attributes a_i as *items*, and images I are referred to as *transactions*. As described in the introduction of the Chapter 3.1, our re-ranking method is based on the statistic of *frequent itemsets*. More specifically, the method is based on the statistic of the number of *closed frequent itemsets* (Section 1.3.1). Mining closed frequent itemset can reduce a large amount of redundant information. Moreover the computational cost is less expensive than mining all frequent itemsets. Rather than closed frequent itemsets, *maximal frequent itemsets* can

be considered. We made some preliminiry experiments and found out that maximal frequent itemsets gave worse performance than closed frequent itemsets. Maximal frequent itemsets is not a lossless condensed representation unlike closed frequent itemsets (*i.e.* it is not possible to derive all frequent itemsets with their frequency information). All terms used as well as the definition of closed frequent itemsets are described in Section 1.3.1, see Figure 1.7 together with the text explanation.

Typical extraction techniques enables to process difficult cases (millions of transactions, hundreds of items) of columns) provided that data is not too dense. However, in some types of data where there are few transactions with respect to the number of items (*e.g.* gene expression databases), mining closed itemsets can fail since the search space grows exponentially with the number of items. Riout et al. [81] proposed to solve the problem of mining closed itemsets from this kind of data by using the *Galois connections* property and extract closed itemsets from the transposed data. The principle is to consider transactions as items and items as transactions. Note that the mining itemsets in the transposition data, the term frequency fr will become the length l and the term length will become the frequency respect to the original database. The same possible set of closed itemsets can be extracted much more efficiently.

Likewise in our case, each image query set contains about 300-500 image transactions while the item space is in a range of 20,000 and mining closed frequent itemset from the transposition database is more efficient. Therefore, after the binarization process, we transpose the matrix before mining closed frequent itemsets. Instead of considering an image as a data mining transaction with binary attributes, each image attribute is now considered as a data mining transaction containing a few images. Figure 3.1 shows the transposition of the data. Note that the data transposition is made only for the efficiency of the mining process. In the following description of the method, we still use the terms from the original database. To mention again, $\mathcal{X}$ is an itemset which is a combination of visual attributes, $fr(\mathcal{X})$ is the number of images having $\mathcal{X}$ and $l(\mathcal{X})$ is the count number of visual attributes sharing between the images supporting $\mathcal{X}$.

3.3.3 Re-rank images

The next step of our approach is to mine sets of visual attributes (itemsets $\mathcal{X}s$) that occur frequently in the image database. As we assumed that the relevant images of a query share similarity among each other, the itemsets that occur frequently (closed frequent itemsets) are likely to come from these images. An image that supports many closed frequent itemsets shares similarity with many other images in the database and therefore is likely to be a relevant image.

Let us give a toy example in Figure 3.2, to show the idea of our approach. In Figure 3.2(a), the images are retrieved from a text based search engine and are sorted by their initial

$Image\ I_i$	$Transaction\ \mathcal{T}_i$
I_1	$\{a_1, a_2, a_3\}$
I_2	$\{a_1, a_4, a_6\}$
I_3	$\{a_1, a_7, a_9\}$
I_4	$\{a_2, a_3, a_6\}$
I_5	$\{a_4, a_5, a_8\}$

(a) Original transaction database.

$Attribute\ a_j$	$Transaction\ \mathcal{T}_j$
a_1	$\{I_1, I_2, I_3\}$
a_2	$\{I_1, I_4\}$
a_3	$\{I_1, I_4\}$
a_4	$\{I_2, I_5\}$
a_5	$\{I_5\}$
a_6	$\{I_2, I_4\}$
a_7	$\{I_3\}$
a_8	$\{I_5\}$

(b) Transposition transaction database.

Figure 3.1: Toy example of the original transaction database in (a) and the transposition of the transaction database in (b).

$Image\ I_i$	$Transaction\ \mathcal{T}_i$	$relevant$
I_1	$\{a_1, a_2, a_3\}$	yes
I_2	$\{a_1, a_4, a_6\}$	yes
I_3	$\{a_1, a_7, a_9\}$	no
I_4	$\{a_2, a_3, a_6\}$	yes
I_5	$\{a_4, a_5, a_8\}$	no

(a) Images retrieved by a search engine in their initial ranking order.

$\mathcal{X}_j$	$fr(\mathcal{X}_j)$	$\mathcal{S}(\mathcal{X}_j)$
$\mathcal{X}_1 = \{a_1\}$	3	I_1, I_2, I_3
$\mathcal{X}_2 = \{a_4\}$	2	I_2, I_5
$\mathcal{X}_3 = \{a_6\}$	2	I_2, I_4
$\mathcal{X}_4 = \{a_2, a_3\}$	2	I_1, I_4

(b) All closed frequent itemsets with $F_{min} = 2$

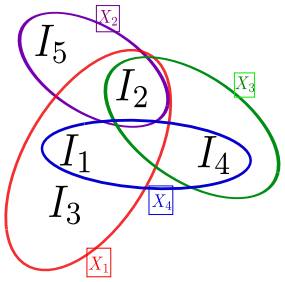
$Image\ I_i$	$\mathcal{X}_j$ supported by I_i	$\#\mathcal{X}_j$ supported by I_i	relevant
I_2	$\mathcal{X}_1, \mathcal{X}_2, \mathcal{X}_3$	3	yes
I_1	$\mathcal{X}_1, \mathcal{X}_4$	2	yes
I_4	$\mathcal{X}_3, \mathcal{X}_4$	2	yes
I_3	$\mathcal{X}_1$	1	no
I_5	$\mathcal{X}_2$	1	no

(c) Images re-ranked according the count of frequent itemsets it supports

Figure 3.2: Toy example illustrating the re-ranking of images according to the count of frequent itemsets. (a) shows that it is likely that the relevant images I_1, I_2 and I_4 share many similar visual attributes to each others whereas the two non-relevant images I_3 and I_4 share very few visual attributes with the other images. Shown in (b), Frequent itemsets are likely to be found in the relevant images and so we can use the number of frequent itemsets found in each images as a criterion to re-rank the images. As shown in (c) all the relevant images are ranked before the non-relevant images.

relevant ranking. The majority of the retrievals are relevant to the query, in this example, there are three relevant images, I_1, I_2, and I_4. However, some noisy retrievals can be obtained, in this example, there are two non-relevant images, I_3 and I_5. In Figure 3.2(b), all the closed frequent itemsets for which $F_{min} = 2$ have been extracted. Notice that the frequent itemsets are obtained mostly from the relevant images. At last, in Figure 3.2(c), re-ranking the images according to the number of closed frequent itemsets improves the results as the relevant images are now ranked before the non-relevant images.

Besides the frequency, the length of an itemset $l(\mathcal{X})$ is also an important property to



(a) Hypergraph - Each ellipse is a hyperedge corresponding to a frequent itemset X_j. Each hyperedge covers a set of vertices, which represent the supporting images I_i.

	$\mathcal{X}_1$	$\mathcal{X}_2$	$\mathcal{X}_3$	$\mathcal{X}_4$
I_1	1	0	0	1
I_2	1	1	1	0
I_3	1	0	0	0
I_4	0	0	1	1
I_5	0	1	0	0

(b) Incidence matrix - The rows are the vertices and the columns are the hyperedges. The "1s" indicates that the hyperedges contain the corresponding vertices.

Figure 3.3: Hypergraph of toy example in Figure 3.2(b) and it corresponding incidence matrix.

determine the relevance of an image. $l(\mathcal{X})$ reflects the similarity of two images containing a common frequent itemset.The longer the itemset is, the more similar to each other the supporting images are. Since relevant images normally appear to be similar to each other, they tend to share a lot of visual attributes. Therefore, the frequent itemset extracted from these images are expected to be long. The images supporting long frequent itemsets are more likely to be relevant images than the images supporting short frequent itemsets.

Scoring function In order to facilitate the way we calculate the ranking score of images, we represent the connections between the images and the closed frequent itemsets as a hypergraph. A hypergraph is a generalization of a graph in which an edge can connect any number of vertices. In our case, we denote $\mathcal{V} = \{I_1, I_2, ..., I_n\}$ as a set of vertices where each vertex I_i is an image and denote $\mathcal{E} = \{\mathcal{X}_1, \mathcal{X}_2, ..., \mathcal{X}_m\}$ as a set of hyperedges where each hyperedge $\mathcal{X}_j$ is a frequent itemset. Figure 3.3(a) shows the hypergraph of the toy example in Figure 3.2(b). The incidence matrix, $\mathbf{A} = (a_{ij})$, of the hypergraph, is an $n \times m$ binary matrix where

$$a_{ij} = \begin{cases} 1 & \text{if } I_i \in x_j \\ 0 & \text{otherwise.} \end{cases}$$

The incidence matrix of the corresponding hypergraph is shown in Figure 3.3(b).

Our ranking score is defined as

$$\boldsymbol{r} = \boldsymbol{w} \cdot \mathbf{A} \tag{3.1}$$

where $\boldsymbol{r}$ is a vector with n elements in which each element corresponds to the score of each image, and $\boldsymbol{w}$ is a vector with m elements in which each element corresponds to the weight of each frequent itemset.

As the importance of the frequent itemsets are not the same, they should have different weights when scoring. We describe four different factors that could affect the importance of each frequent itemset including;

(i) Frequency : A frequent itemset having high frequency should have higher weight since it appears in many images. The high frequency is likely to come from the large number of relevant images.

(ii) Length : Longer frequent itemset should have a higher weight than the shorter because the relevant images tend to look similar to each other and share many visual attributes.

(iii) Area : Since both the frequency and the length are important, we can consider them together.

(iv) Original ranking of the support : The support of a frequent itemset is the images containing the frequent itemset. One prior information is that the original ranking from the text-based search engines is reasonably good. Since the original ranking position of each image is known, we can use this information to give weights to the frequent itemsets.

For different weighting schemes have been experimented. They are described in the experiments section, Section 3.4.4.

Note that in our toy example in Figure 3.2, we do not consider the weights of the frequent itemsets. Our scoring for each image is simply the number of frequent itemsets it contains. This corresponds to $\boldsymbol{w} = \mathbf{1}$ where $\mathbf{1}$ is a vector with all elements are '1s'. It is worth mentioning that the score vector when $\boldsymbol{w} = \mathbf{1}$, is linked to the degree of the hypergraph in graph theory.

3.4 Experiments

This section describes the experimental validation of the proposed approach. We first describe the datasets used for the experiments and then present and analyze the experimental results. For extracting closed frequent itemsets, we use the LCM [93] mining tool.

3.4.1 Datasets and evaluation protocol

The **INRIA Web Queries dataset** was proposed in 2010 by Krapac et al. [53]. The images in the dataset are the top-ranked images from several text queries returned by a web search engine. More precisely, the entire dataset consists of 71,478 images obtained from 353 text queries. All of the images have been scaled to fit within a 150×150 pixels

square while maintaining the original aspect ratio. The queries are very diverse ranging from general terms of object or scenery classes such as 'car', 'bird', 'mountain' to specific names of objects, places, events, or persons such as 'Logo Nike', 'Eiffel tower', 'Cannes festival', 'Cameron Diaz'. For each query, the annotation giving the relevance to the query is provided.

The evaluation protocol is as follows. For each query, the images are sorted according to their ranking score. The average precision AP is calculated per each query and the mean average precision is reported mAP.

Fergus dataset was proposed in 2005 by Fergus et al. [31]. The dataset has in total of 4,091 images from seven text queries returned by Google image search engine. The seven text queries include 'airplane', 'cars rear', 'face', 'guitar', 'leopard', 'motorbike', and 'wrist watch'. This dataset was created by the time image search was not well established giving very poor results from the initial text based retrieval. The majority of the retrieval which is about 70% of the retrievals are not relevant to the query. The metric used for benchmarking is the precision at 15% recall.

3.4.2 Visual attributes

Our visual binary attributes are obtained from binarizing the BoW representation. To compute BoW, we used (i) multi-scale SIFTs as local descriptors from the VLFeat library [95] (12 scales from 3 up to 14 pixels with the default step size of 2 pixels.) and, (ii) a dictionary size of $1,024$. The visual words are pooled from a 4×4 SPM [57] grid which gives in total a vector of $16 \times 1,024 = 16,384$ dimensions for each image. To binarize the BoW representation we use the top-K method which has been described in Section 2.3.1. The method is to set the bins which are among the K highest values of the BoW feature vectors to one while set the rest bins to zero.

3.4.3 Parameters

For finding the parameters, we used a sub-part of the Web Queries dataset. We randomly select 30 queries out of the total of 353 queries. In this experiment, we give $\boldsymbol{w} = \mathbf{1}$ in equation 3.1. The parameters involved in our re-ranking method includes: (i) K, (ii) L_{min} and, (iii) F_{min}. The first parameter K is used in the top-K binarization process. It is related to how many visual attributes are contained in an image transaction. While the second and third parameters, F_{min} and L_{min}, are used in the closed frequent itemset mining process. The closed frequent itemsets must be occurring in at least F_{min} transaction and there must be at least L_{min} visual attributes in each closed frequent itemset. The explanation of these parameters are described in Section 3.3.

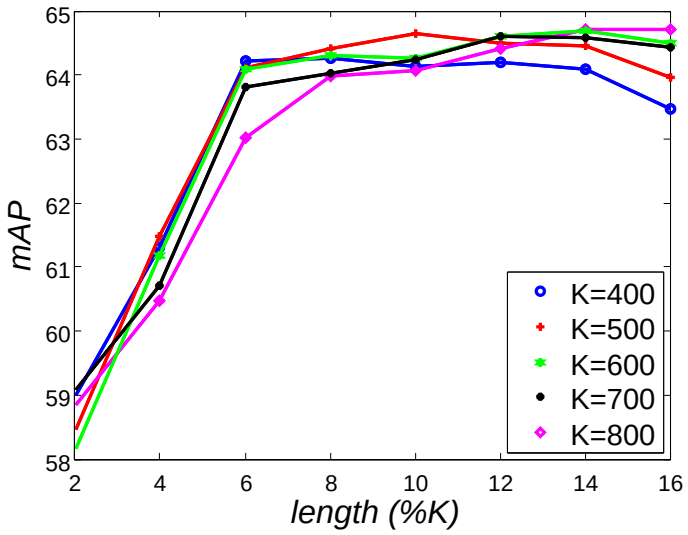


Figure 3.4: Influence of the parameters on the behavior of the algorithm.

We run a grid search on the three parameters by varying K from 300 to 800, L_{min} from 2% to 16% of K, and F_{min} from 2 to 6. The best performance is obtained when setting F_{min} to be 2 and L_{min} to be about 10% of K. We observed that K does not affect the performance so much. Figure 3.4 shows the effect of L_{min} with some K settings. We now set $K = 500$, $L_{min} = 50$, and $F_{min} = 2$ for the rest experiments.

The 30 queries used for this validation are removed from the test set used for the evaluation.

3.4.4 Closed frequent itemset weights

As mentioned in Section 3.3.3, the importance from each closed frequent itemsets might vary according to properties such as the frequency, the length, and the original ranking position of the images which support the frequent itemsets. In this experiment, we use the same data as used in the previous experiment in Section 3.4.3. We investigate four weighting schemes based on :

(i) the frequency $w_i = fr(\mathcal{X})$,

(ii) the length $w_i = l(\mathcal{X})$,

(iii) the area which is $w_i = fr(\mathcal{X}) \times l(\mathcal{X})$,

(iv) the original ranking of the images in the support $\mathcal{S}(\mathcal{X})$ which is $w_i = sr(\mathcal{X})$. To compute sr, we sum the inverse of the original ranking position of the images containing the closed frequent itemset $\mathcal{X}$.

$$sr(\mathcal{X}) = \sum_{i=1}^{l(\mathcal{X})} \frac{1}{rp(\mathcal{S}(\mathcal{X})_i)} \tag{3.2}$$

Method	Search Engine	noWeight	frequency	Length	Area	OriRank
$mAP(\%)$	57.4	64.7	64.5	64.4	64.4	67.1

Table 3.1: Comparison of different weighting schemes.

where $rp(I_i)$ returns the original ranking position of image $\mathcal{S}(\mathcal{X}_i)$. From Equation 3.2, a frequent itemset with original top rank supporting images will have a higher weight.

The results in Table 3.1, shows that weighting the closed frequent itemsets according to their frequency nor their length does not give improvement as we expected. The explanation could be that long closed frequent itemsets have less frequency than the shorter closed frequent itemset and vice-versa. Giving more importance to long closed frequent itemsets, means that we explicitly also give more importance to the less closed frequent itemsets which is not what we want. On the other hand, if we give more importance to higher frequency closed frequent itemsets, we explicitly also give more importance to the shorter closed frequent itemsets. Weighting according to the area also cannot give improvement, it is unlikely to find long closed frequent itemsets with high frequency.

We observed that integrating information from the original ranking can give significant improvement of 2.4%, see Table 3.1. We have tried different functions besides the inverse of the ranking such as exponential functions but could not obtain better results. This weighting scheme is applied in the rest of experiments.

3.4.5 Complexity and computational time.

As mentioned in Section 3.3.2, when the number of images in each query set is less than the possible number of visual attributes, it is much more efficient to transpose the database and consider each visual attribute as a *transaction* and images that contains the attribute as *items* in the transaction. As shown in Table 3.2, on average the time for extracting patterns is about 1.5 seconds which can take several hours to extract the same closed frequent itemsets on the non-transposed database.

The re-ranking computational time differs from one query to another depending on the connectivity of the images (Table 3.2). Query sets such as 'Logo Chelsea' and 'Map World' takes more time to re-rank. The reason is that these query sets contain many near duplicate images and therefore, more closed frequent itemsets are extracted compare to more generic queries such as 'Giraffe' where the images have large variations.

Query	Pattern extraction(s)	Score computation(s)	# Images	# Patterns
Maradona	0.18	0.01	203	1,763
Giraffe	0.30	0.02	219	4,202
Times square	0.49	0.03	249	7,417
Grand canyon	0.81	0.05	204	10,729
Logo Chelsea	3.53	0.40	229	42,536
Map World	8.62	1.03	217	+100,000
Mean 30 queries	*1.58*	*0.17*	*207*	*17,697*

Table 3.2: Computational time for different queries on a single core machine.

3.4.6 Comparison to other methods

In this section, we compare our re-ranking method with other existing re-ranking approaches that has been reported in literature.

INRIA Web Queries dataset. As fas as we know, there are three reported results on this recent dataset. The first approach from Krapac et al. [53] is based on query-relative visual word features. The features are used to train a single generic logistic discriminant classifier that can be used across different queries. They have shown in experiments that their approach gives overall performance better than query-specific classifiers which is to train a classifier for each query set. The second approach from Thollard and Quénot [89] showed that the combination between an unsupervised re-ranking approach with a supervised junk classifier is useful. The third approach is from Liu et al. [63] which is based on a graph ranking approach. Graph based ranking approaches uses a few of top return images from the text based search engine typically 50 up to 100 images as a set of image query samples. The images in the entire query set are reranked based on their relevance to the query samples via a neighborhood graph. As a noisy query set may ruin graph ranking, their main contribution is an approach called *Spectral Filter* that can eliminate the outliers in the image query set in order to feed a cleaner query set to graph rankers. They have shown that *Manifold Ranking (MRank)* [111] gives the best results among other existing graph ranking techniques including *Personalized PageRank (PPageRank)* [34][48]. Combining Spectral Filter with MRank yields higher performance.

As shown in Table 3.3, our re-ranking approach improves the original search engine ranking by about 13% mAP and is better than the the classifier based approaches. Comparing with graph ranking, our approach gives 1% higher than MRank but lower than Spectral Filter with MRank. However as explain in Section 3.2, graph based ranking approaches have very high complexity.

We observed that after applying our re-ranking, the top results are very clean compare to the original ranking especially for queries in which the images have small variations *e.g.* 'logos', 'maps', and 'flags' (Figure 3.6) up to immediate variations *e.g.* 'landmarks'

Method	$mAP(\%)$
Original Search Engine	56.9
LogReg (visual) [53]	64.9
LogReg (visual+textual) [53]	67.3
Query-ind.+Query-dep. [89]	65.5
MRank [63]	69.0
SpecFilter+MRank [63]	73.8
Frequent patterns (ours)	68.0
Weighted frequent patterns (ours)	70.1

Table 3.3: Comparison to other existing re-ranking approaches.

(Figure 3.5), 'celebrities' (Figure3.8). For queries where the images have a large range of diversity *e.g.* 'Generic objects' (Figure 3.7), the top images are less clean. Nevertheless, our re-ranking approach can still improve the original ranking from the text-based search.

Fergus dataset. Most of previous re-ranking approaches have been compared on this dataset [31]. As shown in Table 3.4, our approach improves the original ranking precision by 33%. The result is better than the two classification based approaches [31][53] and the Mrank graph based approach without spectral filtering [63] by about 20%. Compare to the topic model approach in [31], the precision of our method is 7% higher. However, the approach gives less precision than the Mrank with spectral filtering [63] by 4% and less than the LDA topic model approach [36] by 15%. The main motivation of the LDA topic model proposed in [36] is object detection and not image re-ranking. They calculate HOG features on fixed grids and can detect objects that are highly structured and centered with less background. Applying their method to re-ranking fits well with this dataset in which the positive images are very similar to positive bounding boxes in object detection task. Some illustrative images are shown Figure 3.9. However, for image re-ranking, the INRIA Web Queries dataset [53] – in which the images come from diversified text queries and have more intra-class variation – is much more adapted. This dataset contains 70% of non-relevant images in each query. Moreover, the non-relevant images tend to be very similar to each other and many are duplicates. Since our approach relies on the connectivity of the images in the dataset, our approach cannot outperform the two mentioned methods. Nevertheless, our approach outperforms several existing methods.

Qualitative results are shown in Figure 3.9 and Figure 3.10. Notice that for 'Airplane', 'Cars rear', 'Face', and 'Wrist watch', the top images are very clean (this corresponds to the quantitative results in Table 3.4). This is because the images in these dataset do not have a lot of variations. Our approach fails to the 'Guitar' query because a lot of 'music sheets' appear in the dataset and they share a large amount of similarity.

Precision at 15% recall	*Mean*	airplane	cars rear	face	guitar	leopard	motorbike	wrist watch
Google	*43*	50	41	19	31	41	46	70
Query specific - SVM [31]	*54*	35	-	-	29	50	63	93
Query relative - LogReg [53]	*56*	65	55	72	28	44	49	79
Topic model - pLSA [31]	*69*	57	77	82	50	59	72	88
Topic model - LDA [36]	*91*	100	83	100	91	65	97	100
Graph - Mrank [63]	*56*	39	53	66	32	50	79	75
Graph - SpecFilter+MRank [63]	*80*	86	100	75	58	63	79	100
Frequent patterns (ours)	*74*	100	97	91	22	50	64	94
Weighted frequent patterns (ours)	*76*	100	97	91	26	55	68	94

Table 3.4: Fergus re-ranking results

3.5 Conclusions

In this chapter, we proposed a method to re-rank the images obtained from text-based image search engines. Our method relies on the assumption that the majority of the images returned by the initial text-based retrieval are relevant to the query, and that relevant images are visually similar to some other relevant images (*i.e.* they form clusters). Building on this assumption, our approach mines closed frequent itemsets which are groups of image attributes sharing by sets of images. Closed frequent itemsets are likely to be discovered in the relevant images which have visually similar appearance, and not in the non-relevant images which do not share similarity to other images (negative images' appearances are very diverse). In order to apply closed frequent itemset mining, we used the top-K binarization method described in Section 2.3.1. After extracting closed frequent itemsets, the images are ranked based on the amount of closed frequent itemsets found in each of them. We also investigated giving different importance to each closed frequent itemset. We showed that weighting closed frequent itemsets based on the initial ranking position of the images supporting the itemsets can improve the results of the non-weighted scheme. Two dataset including **INRIA Web Queries dataset** [53] and **Fergus dataset** [31] were used to validate our approach. The re-ranking gave significant improvement over the original text based ranking (13% mAP improvement on INRIA Web Queries dataset and 33% mean precision at 15% recall improvement on Fergus dataset.) Our approach takes on average less than 2 seconds to re-rank about 200 images. It compares favorably with state-of-the-art results on the two datasets.

Note that we re-visit image re-ranking at Section 4.5 by applying similar technique described in Chapter 4, to solve the problem of binarization loss and high dimensionality.

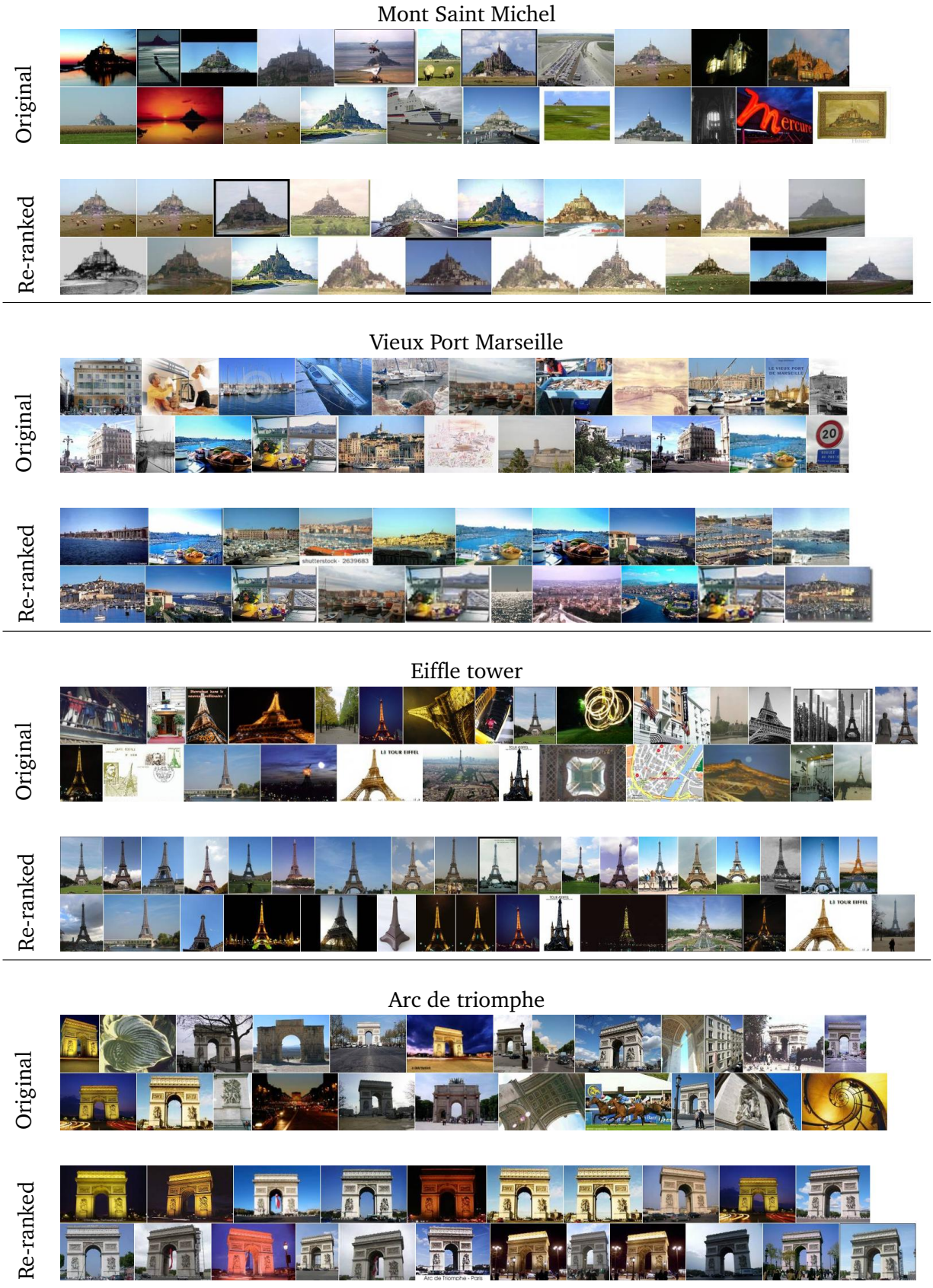


Figure 3.5: Landmark re-ranking results

Figure 3.6: Logos, maps, and flags re-ranking results

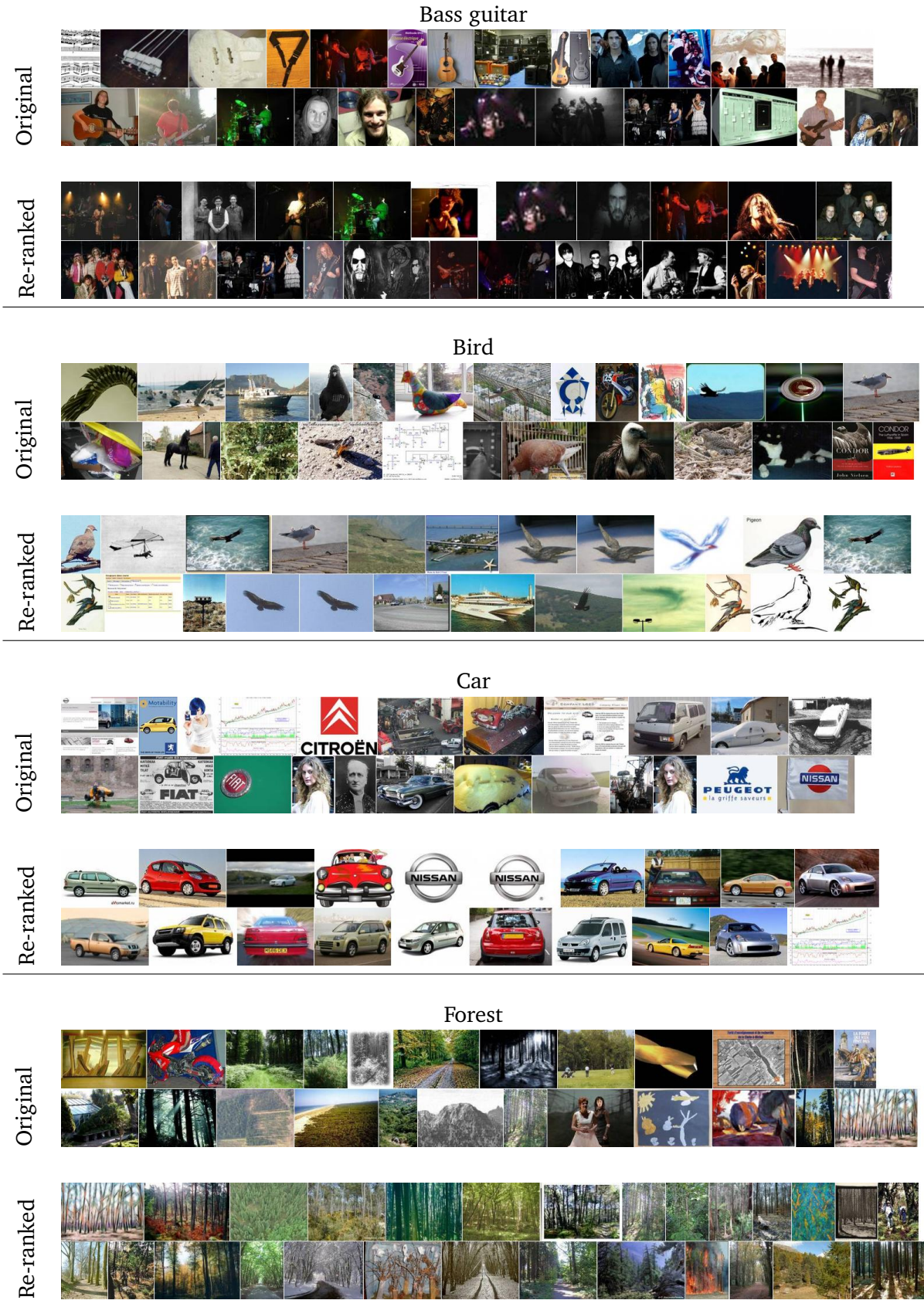


Figure 3.7: Generic object/scene classes re-ranking results

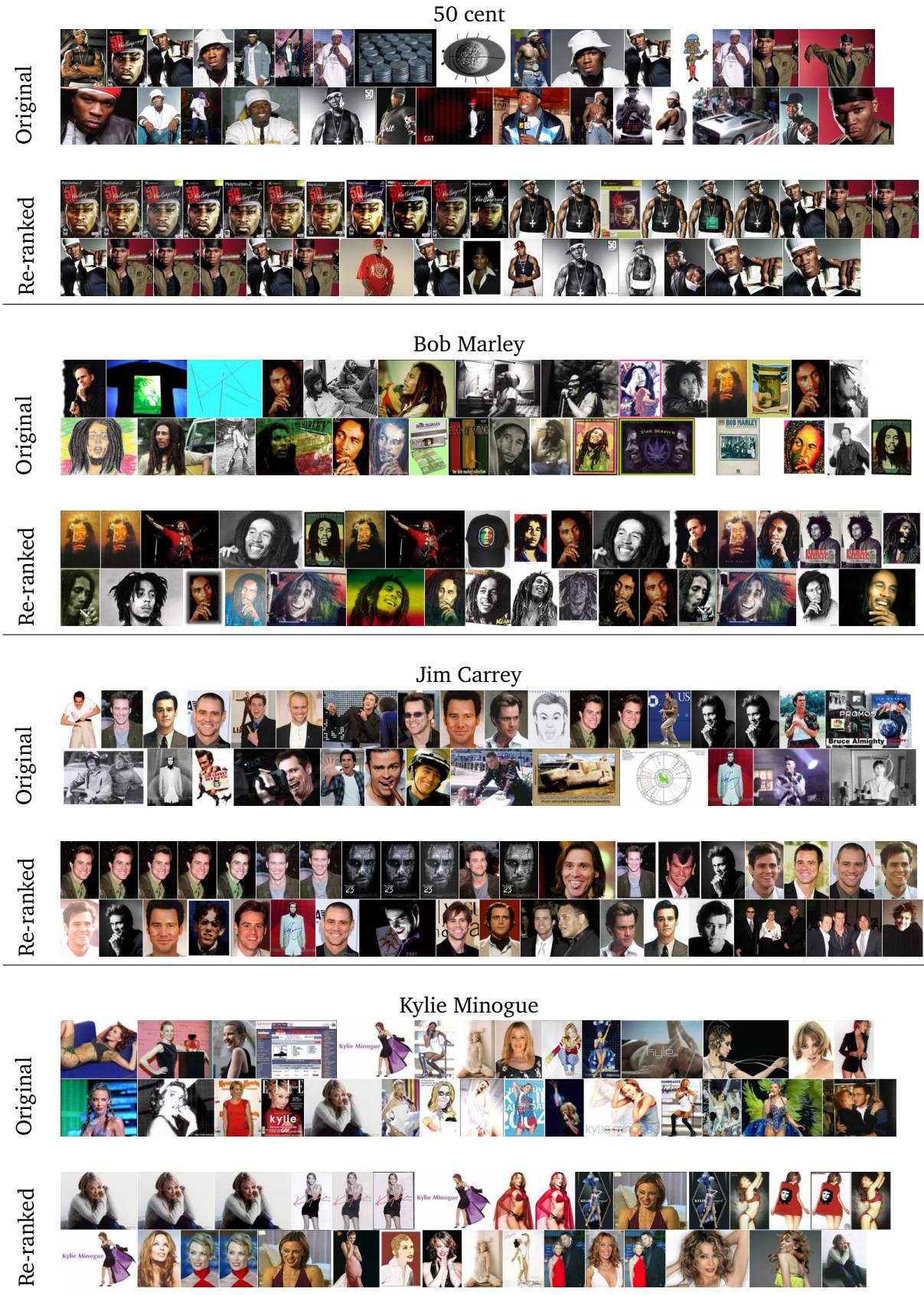


Figure 3.8: Celebrities re-ranking results

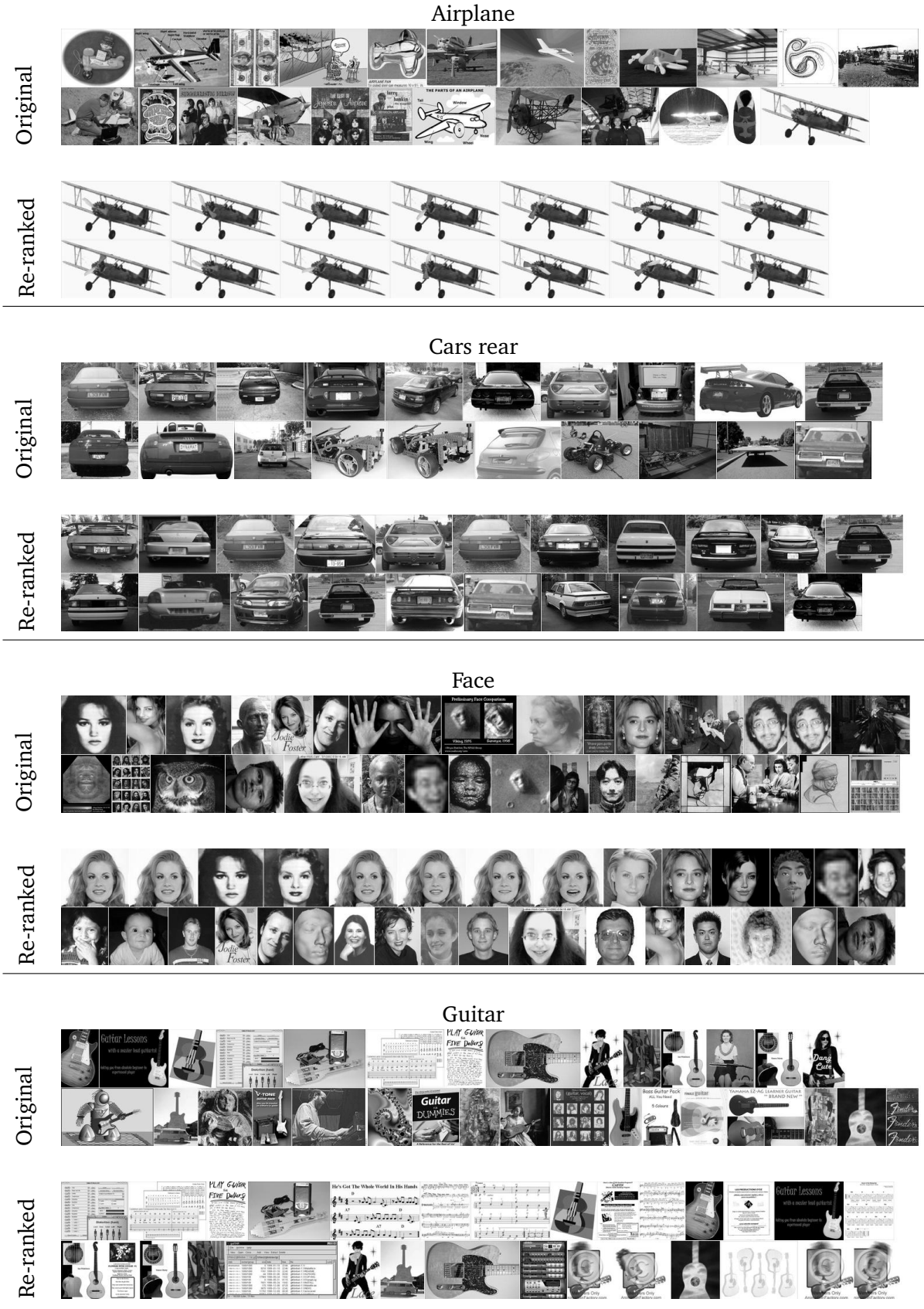


Figure 3.9: Fergus re-ranking results (a)

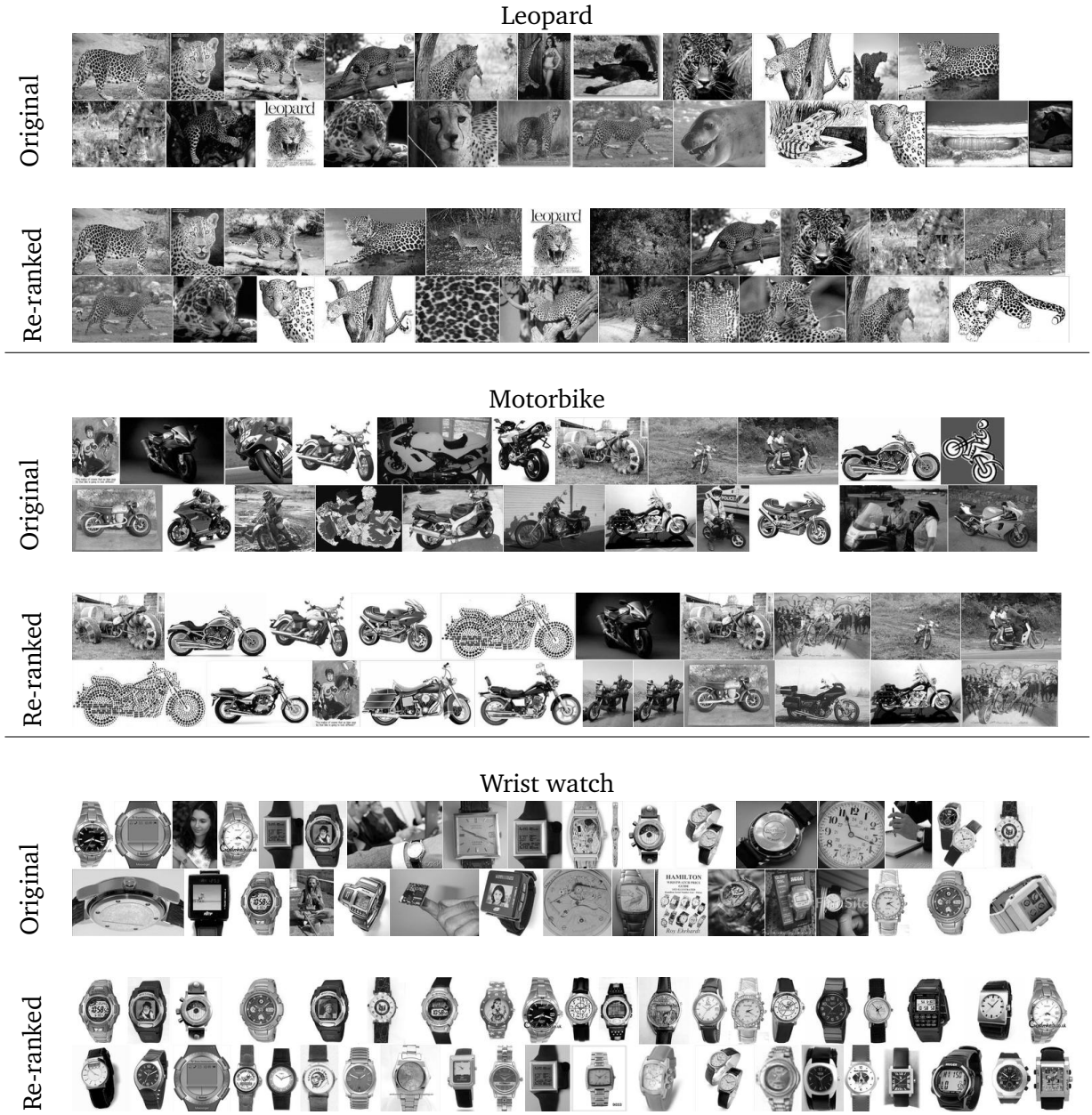


Figure 3.10: Fergus re-ranking results (b)

Chapter 4

Histograms of Jumping Patterns for Image Classification and Object Recognition

Contents

4.1 Introduction 65
4.2 Related Work 68
4.3 Method 69
4.4 Experiments 73
4.4.1 On setting the parameters 74
4.4.2 Texture recognition 76
4.4.3 Pedestrian recognition 77
4.4.4 Image classification 78
4.4.5 Object detection 79
4.5 Extension of multiple random projections to image re-ranking . . . 81
4.5.1 Parameters and complexity analysis. 82
4.5.2 Comparison on INRIA Web Queries dataset. 82
4.5.3 Comparison on QUAERO's visual concepts image dataset 83
4.5.4 Summary on multiple random projections for image re-ranking. . 84
4.6 Conclusions 84

4.1 Introduction

In this chapter, we propose a new image representation that can capture both feature co-occurrences as well as feature relative differences. This work can be seen as an extension

of the work in Chapter 2 by solving two major problems encountered previously. The two problems include: (i) the binarization loss when converting image representations to pattern mining transactions, and (ii) the high dimensionality of the new image representation when using each pattern as a new image feature. The motivation for capturing feature dependencies has been mentioned in Section 1.4. In the following of this section, we describe the motivation in more detail.

One of the central issue of computer vision is the representation of images, which raises the question of how to define sets of visual attributes making tasks such as image classification or object recognition/detection possible.

While the recent literature abounds with examples of such representations, it is worth pointing out that the most successful ones are based on some form of elementary image statistics, often represented by histograms. To mention only a few, SIFT [64] and HOG [21] are histograms of edge orientations, Bag of Words [22] are histograms of visual words, while histograms of LBP [90] have been used successfully for texture and face recognition and color histograms [94] are commonly used in classification tasks.

One big limitation of these representations is the lack of modeling of features dependencies. While discovering such dependencies, in a way analogous to the N-grams used for modeling natural languages, has been recently addressed (*e.g.* [62, 66]), the question of how to discover discriminant groups of co-occurring features remains an open problem. This problem is made intrinsically difficult by the dimensionality of the input space. In most of the case, it is not possible to evaluate all possible combinations of visual words for typical vocabularies size (*e.g.* 10k words). Even if some heuristics can be used to reduce the number of groups of candidates (*e.g.* using locality constraints[80]), the selection process is in general very costly (in [33] over one million candidates have to be evaluated). Another issue is the risk of overspecialization of such combinations, as the number of training image is very small compared to the number of possible combinations.

We address the question of how to discover groups of discriminative features – while keeping in mind the above-mentioned issues (complexity and generalization) – by building on recent advances in data mining.

Our contribution is threefold (see Figure 4.1 for an overview of the approach). First, we propose a new algorithm for transforming histogram based representations into multiple sets of binary items. It makes use of multiple random projections for reducing the dimensionality of the input space while preserving the distance between histograms. Histograms, for each random projection, are transformed into sets of items, which are the features whose probabilities are the top-K components of the projected histogram (Section 2.3.1). It is worth pointing out that, due to the way items are produced, they encode *relative* probabilities of attributes [105]. Second, inspired by the contrast measures proposed by [71] for classification purpose, we introduce the concept of *Histograms of*

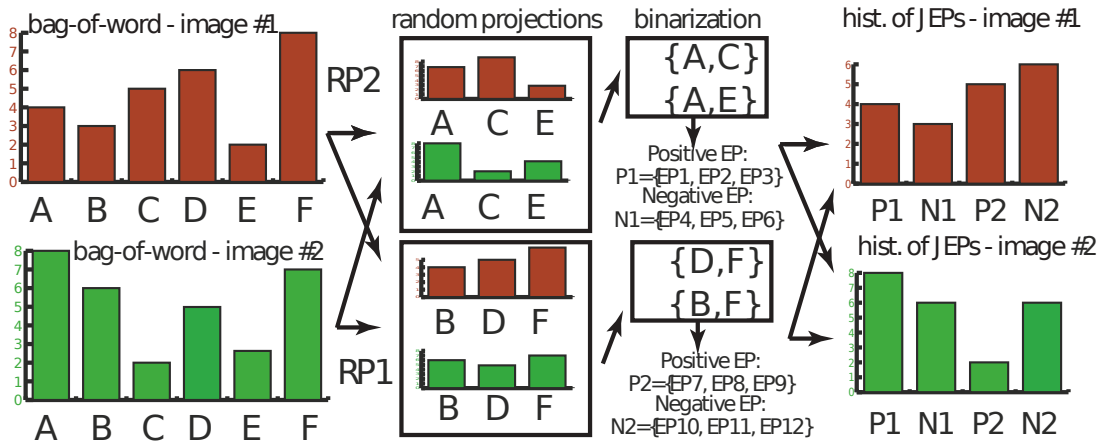


*Figure 4.1: Overview of the approach: feature histograms (*e.g. *bag-of-words) are randomly projected before being binarized (thresholded). Each image is represented, for each projection, by a set of items. The final representation is the distribution of the Jumping Emerging Patterns. See text for more details.*

positive and negative Jumping Emerging Patterns (HJEPs). Mining frequent Emerging Patterns (EPs) [24] allows to provide exhaustively the combinations of features that are both discriminant (*i.e.* good for recognizing one category versus another) and having a good coverage of training examples, without having to produce and test all possible feature combinations. In addition to reduce the complexity, random projections limit the risk of having too specific combinations (low coverage of training examples), by producing smaller set of items. Sets of 'positive' (respectively 'negative') JEPs, the JEPs having 'positive' (respectively 'negative') growth rates [24], are mined from each random projected space (see definitions in Section 5.3). These two steps are done offline during a training stage. Third, we propose a new image descriptor. We represent images by the distribution of their positive and negative JEPs (the ones computed offline). Therefore, if we do P (typically, order of 100) random projections, we obtain a very compact $2P$-dimensional representation (one distribution of positive and of negative patterns per projection).

In order to validate the approach, we report experiments on four different image classification and object detection tasks. We show that the proposed representation not only gives better performance than neglecting feature dependencies, but that it also allows to go significantly beyond state-of-the art results on these different tasks.

This chapter is organized as follows: Section 4.2 discusses previous related works while Section 4.3 describes our approach in detail. Section 4.4 provides the experimental validation of our ideas including a part to describe about the parameters and a part that compares our approach with related work. Finally, Section 4.6 concludes the chapter.

4.2 Related Work

In order to construct high-order features, several recent works have been trying to capture spatial relationships between local features. For example, [56, 61, 62] have encoded the co-occurrences of visual word pairs. Due to the quadratic number of possible pairs, and trying to limit the size of the representation, [66] has proposed to build a *local pairwise codebook* by aggregating pairs whose descriptors are close. An extension of this work has incorporated spatial directions between the local features [67]. Going beyond pairs, [62] managed to construct triplets in a greedy way, by progressively selecting the best features at each order and extend them to the next order level. However, the triplets did not show improvement over the pairs. This may be due to the fact that, some important high order patterns cannot be discovered as the components has been cut off at early stages. Yao and Fei-Fei [107] proposed to discover and encode spatial configurations of discriminative image regions.

As said in the introduction, data mining – which is the process of discovering complete sets of patterns in large datasets according to user defined interests – is highly relevant to the problem we are interested in. Several recent papers have used *frequent itemsets* (*e.g.* [33, 39, 80, 108]), for addressing computer vision tasks. However, frequent patterns are rarely discriminant, explaining why in these works a costly or ad-hoc post-processing stage is necessary for selecting the patterns that are good at discriminating the classes. This filtering step remains an open issue and currently there is no efficient way to generate a set of patterns that can satisfy global constraints (*e.g.* cover all data while gives good prediction accuracy) [10]. In addition to frequency, contrast measure has been applied for obtaining patterns that are more interesting for classification [71]. They can provide complete sets of high-order features giving good performance in several traditional classification tasks [9].

Despite their efficiency (as shown by [102]), none of these methods have been applied to object recognition or image classification. This is what we do here by producing patterns with high growth rate (defined next section), instead of high frequency, in the context of object recognition. One interesting open issue is how to combine efficiently the fragmented information carried by each pattern [52]. We address this issue by proposing a new representation, based on Histograms of positive and negative Jumping Emerging Patterns (or JEPs) – JEPs are Emerging Pattern with additional properties – which summarize efficiently the discriminative power of the JEPs.

While data mining can make the discovery of patterns more efficient than evaluating exhaustively all possible combinations, it requires representing images by sets of *items*. Quack et al. [80] first extracts keypoints, whose centers and scales define local references. The relative position as well as the appearance of keypoints are vector quantized, and one item is defined as the presence of a given keypoint appearance at a given relative position.

A similar approach is presented in [39]. Instead of using keypoints, [102] used a dense grid to represent vector quantized visual attributes based on edges and optic flow features. The approach of [33] is also related to the previous ones. The image is represented by extracting keypoints and by computing local bag-of-words in the neighborhoods of the keypoints. Each local bag-of-words gives a transaction (*i.e.* a set of items), which contains the visual words occurring at least one time. Different from these approaches, we believe (as demonstrated by [105]) that the relative frequency of visual words (*i.e.* the ranking) is more important than the absolute values, and show how it can be represented by sets of items.

Finally, the risk of mining very long transactions, in addition to the prohibitive computational complexity, is that it can produce too specific patterns. Working with local neighborhoods (as proposed by most of the previous approaches) implicitly reduces the length of the transactions, but it does not allow to discover relationships between features that are away from each other. In contrast, we base our approach on random projections of the global distribution of the features, bearing some similarities with the *product quantization* of [46]. It allows us to reduce the size of the transactions without the above mentioned limitations.

4.3 Method

Problem statement. The proposed method addresses binary image classification tasks and assumes each image (or image region) is represented by a collection of loose discrete image features denoted $I = \{x_0, \ldots, x_f\}, x_i \in L$, where $L = \{w_0, \ldots, w_d\}$, is a vocabulary of size d. A prototypical example, taken here to illustrate the method, is the bag-of-features model where images are represented by histograms of visual words $h = (p(w_0|I), \ldots p(w_d|I))$[1]. The motivation of this work is to discover combinations of items (*e.g.* pairs, triplets, etc. of visual words) that would result in a representation (as a distribution over such combinations) more discriminative than the distribution over single independent items. In the following, we denote H^+ (respectively H^-) as the sets of histograms h of 'positive' (respectively 'negative') training images, and define H as $H = \{H^+ \cup H^-\}$. We also assume that $\forall h \in H, h \in \mathbb{R}^d$.

Random projections followed by local thresholding. Pattern mining algorithms cannot be applied to high-dimensional real-valued representations as (i) pattern mining algorithms handle binary items, and (ii) although pattern mining algorithms are very efficient in extracting patterns from a large number of transactions (*e.g.* millions), they can only

[1]However, as shown in the experiments section, other representations falls in this formalism such as histograms of oriented gradients (HOG) for which the vocabulary is obtained by quantizing gradient orientation and position.

tackle a moderate number of items, typically up to a few hundreds, depending on the density of the data. This is due to the search space which grows exponentially with the number of items. As image representations usually have several thousand components, mining patterns from such high-dimensional image data is not only slow but also memory consuming.

Multiple projections of the original representation into small dimensional spaces is one of the key ideas for addressing this issue. The rational for doing this is that (i) representing high-dimensional data by multiple projections leads to good approximations of the data *e.g.* [47] and (ii) mining is very efficient when the dimensionality is low. In practice, the projection is done by simply randomly selecting p visual words from the original BoW. This can be seen as projecting the original d-dimensional data to a p-dimensional subspace where the projection matrix is obtained by randomly selecting p basis vectors among the d ones from the original space (more complex projections have been investigated, without improving the performance). Let R denote the $d \times p$ projection matrix, such that $h^p = h \times R$, $\forall h^p \in H^p$. H^p is denoted as the set of p-dimensional projected histograms h^p. Once the histogram is projected, the bins whose values are among the top-*K* highest values are set to '1' while other bins are set to '0' (*e.g.* if the histograms are projected into a 9-d space and if $k = 3$, we will obtain 9-bit histogram signatures having 3 ones and 6 zeros). The binarized version of the projected histogram is denoted as h^{bp}. More formally, $h^{bp}_j = 1 \iff h^p_j \geq \tau$ where $\tau = h^p_{rank^k(h^p)}$, and $rank^k(h)$ returns the index of the histogram bin with the k^{th} highest value. This process to obtain binary representation bears some similarity with local sensity hashing (LSH) [40]. However, the main difference is that in LSH, a fixed threshold is used instead of top-K. The motivation in using top-K binarization instead of a fixed threshold is to reduce and control the size of the search space. Top-K binarization ensures that all images will be represented by exactly K items. Moreover, we believe (as demonstrated by [105]) that the relative frequency of visual words is more important than their absolute frequency.

We repeat this two-step process several times. After a sufficient number of projections, the relative order of feature probabilities from the original representation is captured. To illustrate this, let us consider the toy example in Figure 4.2. The original BoW representation contains four different real-valued features namely p(A), p(B), p(C), and p(D). The first random process selects visual words A, B, and C. When applying top-K binarization with $K = 2$, the visual words A and C are kept (and considered as a transaction). This infers that both p(A) and p(C) have higher values than p(B). After the second random projection, the fact that p(C) is smaller than p(A) and p(D) can also be derived. From these two iterations, we are able to conclude that (i) both p(A) and p(D) are higher than p(C), and (ii) p(C) is higher than p(B). In this example, it is not possible to determine which one of p(A) or p(D) is higher. However, this information can be discovered if we negate the features $h^{p*} = (1 - h^p_0, \ldots, 1 - h^p_p)$ and apply the same binarization. As the absence of a feature can be as discriminative as the presence of another, we also do the

Input BoW BoW relative order	p(A)=.4 p(B)=.1p(C)=.2 p(D)=.3 p(A) > p(D) > p(C) > p(B)
Random projections	Top-K, $K = 2$
R_1 : p(A) p(B) p(C) R_2 : p(A) p(C) p(D)	{A C} {A D}
Discovered from R_1 Discovered from R_2 Discovered from R_2	p(C) > p(B) p(A) > p(C) p(D) > p(C)

Figure 4.2: Toy example showing that after two random projections followed by top-K ($K = 2$) binarizations, relative relations between the BoW features are preserved (compare the discovered relation with the original BoW relation) and implicitely encoded within the representaion.

same top-K binarization (same K) on the negated version of each projected histogram h^{p*}, $h^{p*} = (1-h_0^p, \ldots, 1-h_p^p)$. This gives a second list of items, the list of *infrequent visual words*. The concatenation of the two lists (*i.e.* the top-K most frequent visual words and the top-K most infrequent visual words) is the *transactional representation* of the image *i.e.* transaction used by the mining process.

As an alternative to random projections, we evaluated the use of principal component analysis (PCA) to reduce the dimensionality of input histograms. However, we obtained much worse results, which can be explained by the fact that it produces a single (low-dimentional) representation per input vector and therefore loses a lot of information once thresholded. In contrast, the proposed method to transform real-valued vectors into multiple sets of binary transactions, reduces dimensionality as well as limits the loss caused by the binarization process.

Jumping patterns. At this stage, we have seen how to represent histogram projections by transactions of frequent/infrequent items. The sets of training images can therefore be seen as sets of transactions. Each set of transactions comes from one single random projection while patterns are extracted from each set of transactions. Our initial motivation, which was the discovery of discriminative groups of vocabulary items, can be stated as the discovery of vocabulary patterns occurring more in one class than in another. This is precisely the key property of *emerging patterns* (EPs) introduced by [24]. EPs are local patterns mined from a two-class dataset. Positive EPs are patterns that appear frequently in the positive class but appear rarely in the negative class. More formally, they are patterns with high *postive growth rates* (GR^+). The positive growth rate GR^+ of a pattern $\mathcal{X}$ is defined as the frequency of $\mathcal{X}$ in the positive data $fr^+(\mathcal{X})$ over the frequency of $\mathcal{X}$ in the negative data $fr^-(\mathcal{X})$. On the other hand, negative EPs have a large negative growth rate GR^- meaning that they appear frequently in the negative class but appear rarely in the positive class. Emerging pattern mining has been explained in more detail in Section 1.3.5.

I	Class	Vis. words distrib.						After proj. R1				Transactions
		A	B	C	D	E	F	A	C	E	F	
1	H^+	.2	.3	.2	.0	.1	.2	.2	.2	.1	.2	{A,C,F}
2		.0	.3	.4	.1	.1	.1	.0	.4	.1	.1	{C,E,F}
3	H^-	.3	.3	.0	.2	.1	.1	.3	.0	.1	.1	{A,E,F}
4		.2	.2	.0	.3	.1	.2	.2	.0	.1	.2	{A,E,F}
5	Test	.0	.3	.1	.1	.4	.1	.0	.1	.4	.1	{C,E,F}
6		.1	.3	.0	.2	.1	.3	.1	.0	.1	.3	{A,E,F}

Figure 4.3: Toy example showing how patterns are obtained.

Jumping Emerging Patterns (JEPs) are special case of EPs with $GR^+ = \infty$ or $GR^- = \infty$. We distinguish positive and negative JEPs. The positive JEPs are those with $GR^+ = \infty$ which means that they appear frequently in the positive class and absent from the negative class (*i.e.* never occurs in the negative class). While the negative JEPs are those with $GR^- = \infty$. In our method, Jumping Emerging Patterns are inferred from closed patterns (Section 1.3.4), which condense the whole set of patterns w.r.t the frequency measure [49]. Indeed, the whole set of frequent patterns can be generated from the closed patterns only, meaning that all the information is present in the closed patterns. Closed patterns are optimal in sense of growth rate [49]. Moreover, finding them is *in average* polynomial in the number of items [58] whereas usual techniques are exponential in the worst case [24]. This is an interesting result because in practice average analysis is closer to real situations than the worst case. Up to now, there is no result in the average-case complexity for mining frequent patterns. Previously in Section 1.3.1, pattern mining terminology and the definition of emerging pattern have been described in more detail.

Representing images by histograms of jumping patterns. One key idea of this work is to compute statistics of patterns found for each random projection to build the new image representation. The complete representation is obtained by aggregating histograms of the whole set of patterns coming from the different projections. More precisely, for each random projection, a set of positive JEPs (found only in the positive images) and a set of negative JEPs (found only in the negative images) are extracted. We therefore build a histogram with two bins, the first is the count of positive JEPs (those found in positive images) and the other is the count of negative JEPs. For P random projections, we hence obtain a $(2 \times P)$-dimensional histogram image representation, the so-called HJEPs representation. The HJEPs representation is intended to be used with classifiers. The underlying idea is to condense the information given by the set of patterns, rather than using each individual pattern as an individual image feature. As mentioned in the introduction, the number of patterns is high and using each pattern as a feature would result in a too large representation.

Toy example. Table 4.3 shows a toy example used to illustrate the method, containing 6 image histograms (4 training ones for discovering the JEPs, and 2 for testing). The dimensionality of the input space is $d = 6$ while the dimensionality of the projected space is $p = 4$. The number of random projection is $P = 1$ and this projection is such that the projected vectors are made of the 1st, 3rd, 5th and the 6th component of the original ones. In this example $k = 3$, meaning that the 3 visual words having the highest probabilities are kept in the transactions. After obtaining the transactions (also given in Table 4.3), the JEPs with minimum frequency threshold $F_{min} = 1$ are computed. In this example, positive JEPs are {A,C,F}, {C,E,F}, and {C,F} while {A,E,F} is the single negative JEP. Note that e.g. {C} is not generated by our method since {C} is not a closed pattern. ({C} is covered by {C,F}.) Finally, the two test images are coded by counting the number of positive and negative JEPs, the representation consisting in these two values, *i.e.* $(2, 0)$ for image 5 and $(0, 1)$ for image 6. Please note that in this toy example, we do not show the patterns representing infrequent items.

Computational time. For each projection and for each image to classify, obtaining the histograms of JEPs requires; (i) binarizing feature histograms, and (ii) computing distribution over JEPs in transactions. The histogram binarization procedure involves the search of the k^{th} highest values, which can be done in $\mathcal{O}(k \times log(p))$ using a partial sort algorithm. To give some insights about the run time, encoding 100,000 images with 500 JEPs (which is in average the number of JEPs in our experiments), on a single core of CPU, takes around 0.3 second for the binarization and around 1 second for counting JEPs. Regarding the computation for extracting the patterns which can be done offline, for our typical setting, the run time is about 1 second.

4.4 Experiments

These experiments validate our approach in the context of different recognition tasks and with different types of image features (such as HOGs, SIFTs, LTPs or bag-of-features), by showing how images can be efficiently represented by the proposed Histograms of Jumping Emerging Patterns (HJEPs). More precisely, these tasks are: (a) *texture recognition* on the KTH-TIPS2a dataset [15], (b) *pedestrian recognition* on the Daimler Multi-Cue, Occluded Pedestrian Classification dataset [26], (c) *image classification* on the Oxford-Flowers 17 dataset [69], and (d) *object detection* on the PASCAL VOC 2007 dataset [28]. These experiments show that our HJEPs significantly outperform the histograms of independent features, leading above state-of-the-art results on these four tasks.

In all experiments, we use DPM-delta [49] for extracting the JEPs. The algorithm mines all closed patterns from the entire database using a FP-tree like algorithm. Then the

positive and negative jumping patterns are derived from the closed patterns. In our case, the jumping patterns are the ones that appear only in one class of the data.

4.4.1 On setting the parameters

To get some insight about the proposed method, we present several experiments on a simple dataset and show the influence of the parameters on the behavior of the algorithm (especially the performance). This simple dataset is a sub-part of the KTH-TIPS2a [15] dataset, in which the *'lettuce leaf'* category is considered as the positive class while the remaining categories are considered as the negative class. The training set contains 3,420 images with 324 positives. The category of test images are predicted with a linear SVM classifier taking HJEPS as input, and the performance is measured by the average precision (AP). As low-level features, we use single scale circular sampling of uniform-LTP, as proposed by [87]. Consequently, the HJEPs are computed from 118-d histograms of LTP.

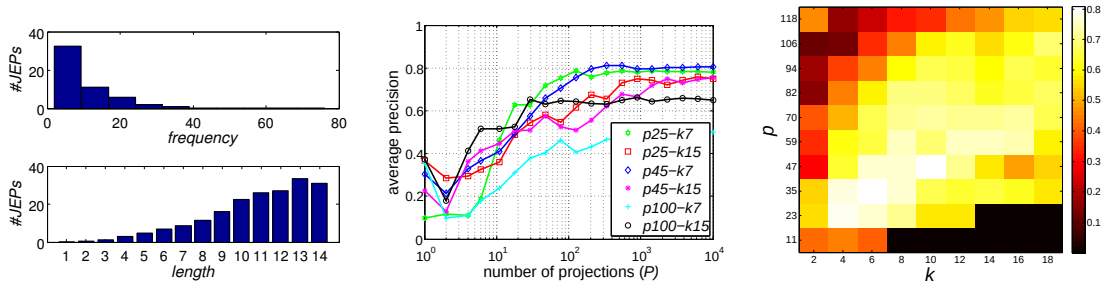
The three key parameters are the number of projections (P), the dimensionality of the projection space (p) and the number of items (k) per transactions obtained by thresholding the histograms. Once these parameters are set, the number of JEPs can vary from one dataset to another. Figure 4.4a gives typical number of JEPs per random projection having a given frequency (number of image containing the JEP) and length (number of items in the JEP), for $p = 25$, $k = 7$. In practice, we limit the total number of JEPs per projection in the range of 200-2,000 by keeping only the ones whose frequency is above a given threshold which is 1% of the number of positive training images. The reason why the threshold is fixed to 1% is that for many categories the number of positive images is about one hundred and 1% is the minimum frequency that can be set. For categories which have more number of positive images, the threshold can be set lower. However, we still set the threshold to 1% in order to limit the number of JEPs. If the number of JEPs is too large, the whole process can be slow.

Number of projections (P). Figure 4.4b shows the AP as a function of P at different values of K and p. The AP increases with P until it saturates, in the range 100-1,000. Indeed, if having enough JEPs is necessary to have a good coverage of training images, adding more helps marginally. P also depends on the dimensionality of the original feature (d). The dimensionality of the feature used in Figure 4.4b is $d = 118$ while P saturates, in the range 100-1,000. However, when the original feature has higher dimensions, more number of P is required in order to cover all information of the original feature. Note that there is no plot to show the relation between P and d. However, we draw this conclusion from other experiments that used other types of features with different dimensionality. The dimensionality of the FLH feature used in Section 4.4.4 is 10,000 and the performance saturates at about 1,000 random projections, while the

dimensionality of the based features using in other experiments are in a range of 100 up to 2,000, we observed that the performance saturates at about 100 random projections.

Binarization threshold (K). Figure 4.4(c) shows the performance according to the number of items (K) and the dimensionality of the projection space (p). K has been set from 2 up to 18 while p varies from 10% to 100% of the size of the input dimensionality (in this experiment $d = 118$). We observed that K is related to p. And the optimal score is reached when K is set to about 20% of p. This K value seems to be the optimal number for considering the present or absent of the visual attributes. In addition, it is important that p should not go further than a certain limit because K will have to be set large as well. Although more emerging patterns are generated since there are more items (K is also larger), the performance drops, which is probably due to over-fitting. When the images contain many visual attributes, it is easy to extract many long JEPs from the training set. However, these JEPs will not generalize well to the test data.

Dimensionality of the projection space (p) Figure 4.4(c) shows that p can be set at different values but not too high and not too low. The reason is that setting p too low or too high leads to very few posible sets of P projected features since in our case the projected features can be seen as sets of random selection of features. To give as an example, suppose the original feature has $d = 4$, if $p = 1$ or $p = 3$ the posible number random feature combinations will be 4 (4 choose 3, 4 choose 1) compare to when set $p = 2$ the posible number of combinations is 6 (4 choose 2). A clearer plot can be seen in Figure 4.4(b). Notice that among the different p values, $p = 100$ which is as close as the dimenion of the original feature ($d = 118$) gives poor results compare to other settings.



(a) Number of JEPs of having a given support and length

(b) Average precision according to the number of projections.

(c) AP as a function of p and k

Figure 4.4: Influence of the parameters on the behavior of the algorithm.

Setting the parameters. We also find the parameters for other datasets. For the datasets used in Section 4.4.4 and in Section 4.4.5, the datasets are split into three parts: (i) train, (ii) validation, and (ii) test. We used the train and validation data to search for the parameters. While the dataset used in Section 4.4.3, only train and test data are provided.

Kernel	1st order	HJEPs	1st + HJEPs
Linear	68.8	74.1	75.0
RBF-χ^2	71.2	73.7	74.0

Table 4.1: Comparison of linear and non-linear classifiers (mAP).

In this case, we find the parameters using 3-fold cross-validation on the train set. From both parameter tuning protocols, we observed that p, k have the same behavior for all datasets and we can set p and k to be the same for all experiments. The only parameter that has to be changed is P which depends on the dimensionality of the original features. We have observed that when the dimensionality of the original feature is large, P has to be set larger. The explanation is that more random projections are required to cover all information from the original representation. In the remaining of the experiments, the parameters are set to $p = 25$, $k = 7$, $P = 100$. Only the FLH feature with 10,000 dimensionality (Section 4.4.4), we set $P = 1,000$.

Non-linear SVM and combination of first and higher order features. As the proposed representation encodes some non-linearities of the features, it is interesting to compare it against a non-linear classifier applied to the features taken as independent (*i.e.* the original histogram). In this experiment, we compared the performance of a linear SVM using our HJEPs with a RBF-χ^2 SVM (known to perform well on histograms) using histograms of the original features. The parameters of the RBF-χ^2 kernel were set by cross validation. We computed the average precision on the 11 classes of the KTH- TIPS2a dataset and report in Table 4.1 the mean average precision (mAP). It is worth pointing out that the HJEPs used with a linear classifier outperformed the first order features with the non-linear SVM. As combining, by a simple concatenation, first-order features (histograms of features) and HJEPs gave slightly a better performance than HJEPs alone, this combination has been done in all the experiments presented in the next sections.

4.4.2 Texture recognition

The experiments on texture recognition have been done on the KTH-TIPS2a [15] dataset, which is a dataset including images of 11 different materials (*e.g.* wool, linen, cork). For each material there are 4 image samples in which the samples were photographed at 9 scales, 3 poses and 4 different illumination conditions. Figure 4.5 shows some example images from the dataset. The evaluation was done by following the protocol proposed in [15], which consists in reporting the mean accuracy(mA) over the 4 runs (multi-class classification). During each run, all images of one sample were taken for testing while the remaining images of the 3 samples were used for training. As in the previous section, the features used for the experiments were 118-d histograms of uniform-LTP features [87].

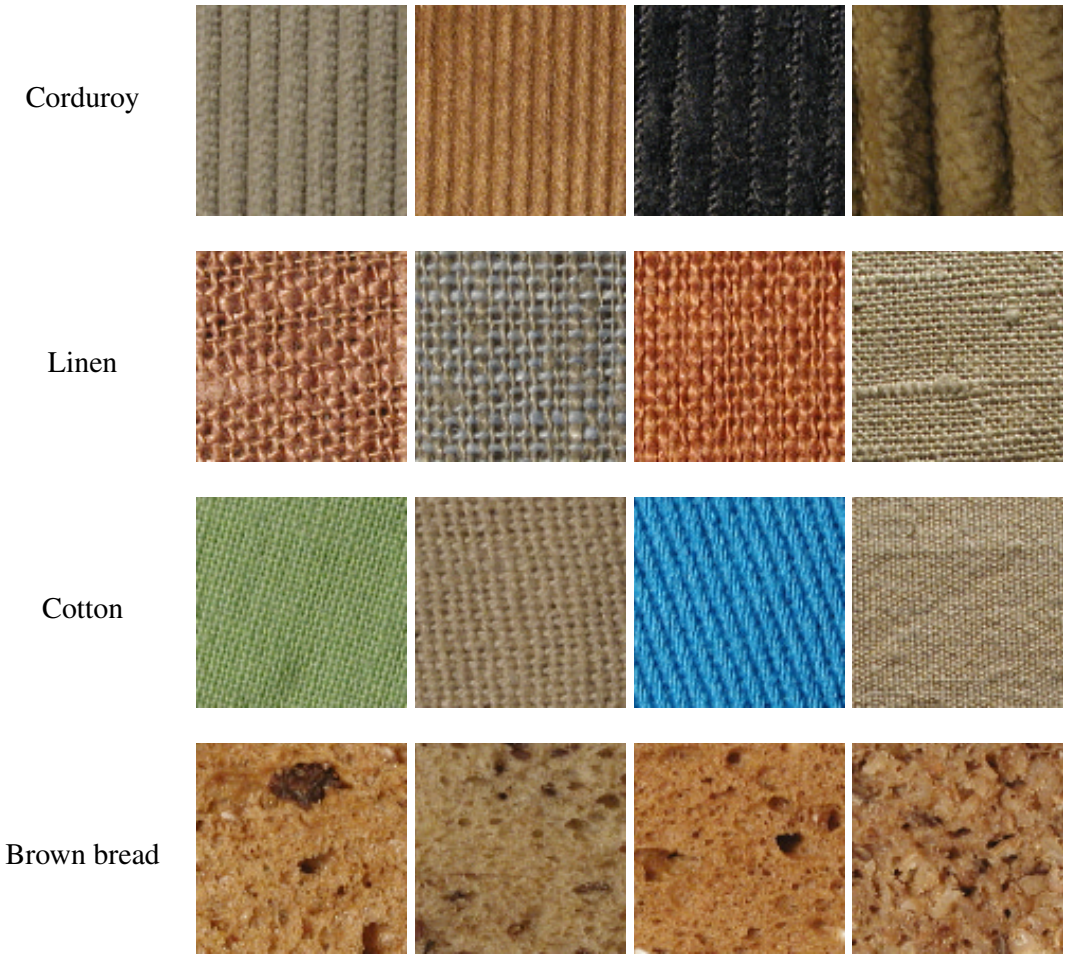


Figure 4.5: Example images from the KTH TIPS 2a texture dataset [15]

The classifier was the linear SVM and the parameters of our method were as given before. As shown in Table 4.2, our method improves the performance of histograms of LTP by more than 5%, validating the relevance of modeling the dependency between features. In addition, our method significantly outperformed other approaches.[2]

4.4.3 Pedestrian recognition

The Daimler pedestrian classification dataset [26] consists of manually labeled pedestrian and non-pedestrian bounding boxes in images captured from a vehicle. The images have a resolution of 48 x 96 pixels with a 12-pixel border around the pedestrians. The dataset has been split into three subsets (i) 52,112 non-occluded pedestrian samples for training (ii)

[2]Please, note that we did not report the results of [15] using multi-scale features and complex decision trees (with non-linear classifiers at every node), as our point is to demonstrate the strength of our features for a given classifier.

Method	mA(%)
Chen et al. [19]	64.7
Caputo et al.[15]	71.0
LHS [83]	73.0 ± 4.7
Color+MLBP[60]	73.1 ± 4.6
histogram of LTP (baseline)	69.9 ± 3.0
HJEPs (ours)	**75.0±3.3**

Table 4.2: mean accuracy(mA) on the KTH-TIPS2a dataset.

25,608 non-occluded pedestrians for testing and (iii) 11,160 partially occluded pedestrian samples, also for testing. Note that there are no partially occluded pedestrian samples available for training. [26] used segmentation based on stereo and optical flow images to help identify occluded pedestrians. Since there is no partially occluded pedestrian samples available for extracting emerging patterns and also we are only interested in comparing features, we only evaluate the results on (ii) subset which is the non-occluded pedestrian test set. Figure 4.6 show some samples from the dataset. For our baseline, we extracted histograms of oriented gradients descriptors (HOG) from intensity gray-scale images, using the same setting as in [26], which is 12 orientation bins and 6x6 pixel cells, accumulated to overlapping 12x12 pixel blocks with a spatial shift of 6. However, we have *L1*-normalized and square rooted the features. As shown in Figure 4.7 this baseline is already better than the best result of [26] which combined intensity, stereo, and optical flow. The proposed HJEPs improves significantly over the baseline as well as other state-of-the-art approaches [103].

4.4.4 Image classification

Our primary motivation for doing these experiments was to provide comparisons with the approach of [33] which bears some similarity with ours. The interesting point is that the authors made the image features (FLH) for the Oxford-Flowers 17 database publicly available[3], allowing us to give some meaningful comparisons (as our patterns are made from strictly the same low-level image features as theirs).

The Oxford-Flowers 17 database[4] contains 17 flower categories with 80 images per class. Figure 4.8 show some samples of the database.

To compare results, we followed the same protocol as [33], *i.e.* 20 folds cross validation, and reported the mean classification accuracy (mA). Results are given in Table 4.3. Note that we did not give comparisons with the features of [33] obtained by combining shape (FLH_S) and color (FLH_C), as the authors has only publicized FLH_S features.

[3] $http://homes.esat.kuleuven.be/\ bfernand/eccv_flh/index.html$
[4] www.robots.ox.ac.uk/ vgg/data/flowers/

(a) Pedestrians.

(b) Non-Pedestrians.

Figure 4.6: Some samples from the Daimler pedestrian classification dataset [26].

With the mA of $\mathbf{93.8 \pm 1.4}$, our approach outperformed the performance of all the recent approaches we are aware of, including [33].

4.4.5 Object detection

The experiments on object detection were done on the PASCAL VOC 2007 dataset, consisting in 9,963 images of 20 different object classes with 5,011 training images and 4,952 testing images. The task is to predict bounding boxes of the objects of interest if they are present in the images. It is the most well known dataset for object detection and several

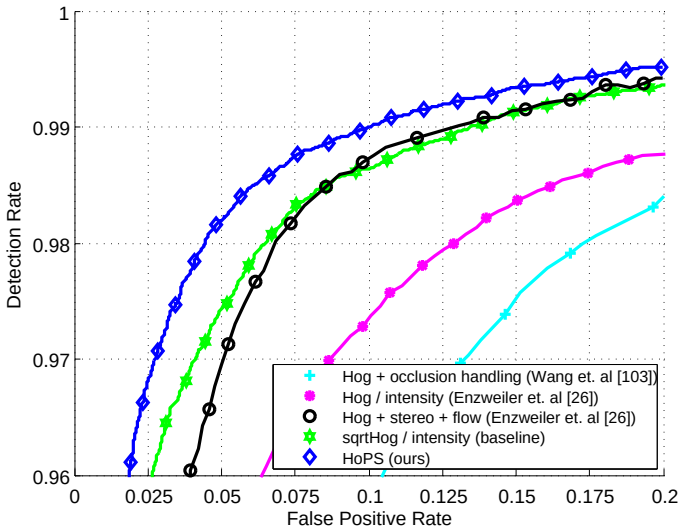


Figure 4.7: Pedestrian recognition on the non-occluded testset of Daimler pedestrian dataset.

Figure 4.8: Example images from the Oxford-Flowers 17 database.

competitive methods have been proposed for this dataset.

We built on the well known part-based detector [30] version 5 (the latest released). As our goal is to validate the proposed image representation (and not to propose a new detector), we used the detection framework of [30] as it is but replaced the original image features (HOG) by ours in the scoring function. Following several works showing that combining texture and shape together gives better results (e.g. [44, 110]), we combined LTP and HOG features. In practice, the representation was obtained as follows. We first rescaled the bounding boxes of training object (roots and parts obtained by using

method	mA(%)
Nilsback [69]	88.3 ± 0.3
CA [51]	71.0
L1-BRD [105]	89.0 ± 0.6
FLH_S [33]	92.0 ± 1.5
HJEPs (ours)	$\mathbf{93.8 \pm 1.4}$

Table 4.3: Comparison with state-of-the-art results on the Oxford Flower 17 dataset.

the standard pre-trained detector) to the size of 128x128 pixels. Uniform LTP features were then extracted from a 6x6 non-overlapping grid, giving a total dimension of 4,248. With the combination of (HOG+LTP), we obtained a gain of 1.0% mAP over the original detector. Then, the distribution of HOGs and LTPs are used as the input of our algorithm for computing the HoPS. As shown in Table 4.4, the gain of our full system is of 1.7% mAP over the original detector. We believe than the improvement is less on this dataset as JEP suffers from the very high imbalance of the positive/negative samples in this task (*i.e.* few hundreds positives vs hundred thousand negatives). Other types of patterns which can handle this issue have to be investigated. Since our method is independent of the type of patterns used, JEPs can easily be replaced by other types of patterns. Nevertheless, JEPs improve the baseline. Most of all, we achieved state-of-the-art result on this extremely competitive dataset.

Method	plane	bird	bike	bottle	boat	bus	cat	car	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	mAP
Best 2007 [28]	26.2	40.9	9.8	9.4	21.4	39.3	43.2	24.0	12.8	14.0	9.8	16.2	33.5	37.5	22.1	12.0	17.5	14.7	33.4	28.9	*23.3*
UCI [23]	28.8	56.2	3.2	14.2	**29.4**	38.7	48.7	12.4	16.0	17.7	24.0	11.7	45.0	39.4	35.5	15.2	16.1	20.1	34.2	35.4	*27.1*
LEO [112]	29.4	55.8	9.4	14.3	28.6	44.0	51.3	21.3	20.0	19.3	25.2	12.5	50.4	38.4	36.6	15.1	19.7	25.1	36.8	39.3	29.6
Oxford-MKL [96]	**37.6**	47.8	**15.3**	15.3	21.9	50.7	50.6	**30.0**	17.3	**33.0**	22.5	**21.5**	51.2	45.5	23.3	12.4	23.9	28.5	45.3	48.5	*32.1*
LBP-HOG [110]	36.7	59.8	11.8	17.5	26.3	49.8	58.2	24.0	**22.9**	27.0	24.3	15.2	58.2	49.2	**44.6**	13.5	21.4	34.9	47.5	42.3	34.3
CN-HOG [2]	34.5	**61.1**	11.5	**19.0**	22.2	46.5	58.9	24.7	21.7	25.1	27.1	13.0	59.7	**51.6**	44.0	**19.2**	**24.4**	33.1	48.4	49.7	*34.8*
HOG [30] (baseline)	33.2	60.3	10.2	16.1	27.3	54.3	58.2	23.0	20.0	24.1	26.7	12.7	58.1	48.2	43.2	12.0	21.1	36.1	46.0	43.5	*33.7*
HJEPs (ours)	37.0	60.7	11.2	18.6	27.8	**54.5**	**59.1**	26.9	20.5	25.8	**29.0**	15.3	**59.9**	49.8	43.0	13.4	23.2	**38.4**	**48.8**	**45.1**	***35.4***

Table 4.4: Comparison with state- of-the-art approaches on the PASCAL VOC 2007 dataset.

4.5 Extension of multiple random projections to image re-ranking

The main idea in this chapter which is to: (i) project the high dimensional feature space into several low dimensional sub-spaces by multiple random projections, (ii) extract patterns in each sub-space, and (iii) merge the results, can also be applied to image re-ranking, addressed earlier in Chapter 3.

In Chapter 3, in order to obtain good results, the possible items (visual attributes) were about 20 thousands while each dataset contains about 200 transactions (images). Since the search space grows exponentially with the number of items, extracting closed frequent itemsets can take several hours. In Chapter 3, we solved the problem by extracting closed

frequent itemsets in the transposed datasets. The drawback of mining closed frequent itemsets in the transposed data is that the search space now grows exponentially with the number of images. Therefore, re-ranking images in larger datasets can be slow.

Instead of transposing the data, we applied similar technique as used in this chapter which is to project the data into low dimensional sub-spaces. And for each sub-space features, we binarized the projected histograms by top-K binarization and extract closed frequent itemsets. We used the same scoring function as used in Chapter 3 which is the weighted count (based on the original ranking) of the closed frequent itemset. Then, we simply averaged the ranking scores of all projections.

4.5.1 Parameters and complexity analysis.

The same 30 queries of the INRIA Web Queries dataset as used in Chapter 3 for validation were used for finding the hyper-parameters including: (i) the number of projections $P = 50$ (performance saturates after 50), (ii) the number of items per transactions $K = 20$, (iii) the dimensionality of the projection $p = 800$, (iv) the minimum frequency threshold $F_{min} = 2$, and the minimum length threshold $L_{min} = 2$. Regarding the complexity, our new method is linear according to the number of images. Although, we have not tested on large re-ranking datasets but we can see from Chapter 5, in Figure 5.4, in which the computation of extracting closed frequent itemsets is varied according to the number of images. Note that the representation of the images in our new re-ranking method and the image representation in Chapter 5 are the same (*i.e.* all images in each database have the same fixed amount of items).

We compare the computational time of pattern extractions process as well as the score computation process between the re-ranking method in Chapter 3 and our new approach in Table 4.5.

Notice that for the new approach, the computation as well as the number of closed itemsets do not vary so much according to the queries which is not the case in Chapter 3. Moreover, in our new method, the sub-processes (*i.e.* transaction encoding and pattern extraction of each random projection) are independent to each other and therefore we can parallelize the computation. Note that in Table 4.5, for the new approach, the number of CPUs used is equal to the number of random projections which is 50.

4.5.2 Comparison on INRIA Web Queries dataset.

Similar results on the INRIA Web Queries dataset compared to the results in Chapter 3 were obtained. Moreover, we managed to further improve the results by choosing the best few projections out of many random projections. Each pattern set extracted from each random projection can be used for re-ranking images, we evaluated the result of

Query	Pat. Extract(s)		Scoring(s)		#Pat.	
	Tr.	*Rp.*	*Tr.*	*Rp.*	*Tr.*	*Rp.*
Maradona	0.18	0.10	0.01	0.03	2k	7k
Giraffe	0.30	0.10	0.02	0.04	4k	8k
Times square	0.49	0.14	0.03	0.07	7k	14k
Grand canyon	0.81	0.12	0.05	0.06	10k	11k
Logo Chelsea	3.53	0.13	0.40	0.06	42k	11k
Map World	8.62	0.13	1.03	0.07	100k	12k
Mean 30 queries	*1.58*	*0.11*	0.17	0.04	17k	9k
Std. 30 queries	*6*	*0.02*	0.02	0.02	30k	3k

Table 4.5: Comparison of computational time between the method of transposing data (Tr.) and the method of multiple random projections (Rp.) on the validation data.

Method	$mAP(\%)$
Weighted frequent patterns (Chapter 3)	70.1
Weighted 50 multi-random proj.	70.5
Weighted best 20 multi-random proj.	72.2

Table 4.6: Comparison of the re-ranking approach in Chapter 3 and the re-ranking approach based on multiple-random projections

each random projection and kept the best few projections which gave the highest results. We made 3-fold cross-validation on the validation data and found that the performance saturates at about 20 best random projections. After we selected the best 20 out of 5,000 random projections on the validation data, we used them on the test data. We observed that combining the results from the best 20 projections yields higher results than using all projections. The results are shown in Table 4.6.

4.5.3 Comparison on QUAERO's visual concepts image dataset

The **QUAERO's visual concepts image dataset** [86] is a very recent dataset for evaluating web image re-ranking. At the time Chapter 3 was being developed, we were not aware of this dataset, and therefore, it has not been used in Chapter 3. The dataset is similar to the INRIA Web Queries dataset in the way the images are obtained. The diversity of the queries, type of images, as well as the intra-class variations, are also similar with the INRIA Web Queries dataset. The difference is that the number of images per concept is larger which is about 950 images per concept. Moreover, since the dataset was created after the INRIA Web Queries dataset, websearch engines had been improved, resulting cleaner data. For each concept, about 55% of the images are relevant images while the INRIA Web Queries dataset has 40% of relevant images. The number of concepts is also larger consisting of 519 concepts. However, we follow the same evaluation protocol as used in [89], by using 100 concepts as test data to report the mAP over the 100 test concepts. As shown in Table 4.7, our approach significantly improves the original ranking

Method	$mAP(\%)$
Original ranking	70.4
Query-ind.+Query-dep. [89]	72.7
Weighted best 20 multi-random proj. (ours)	**76.1**

Table 4.7: Comparison to other existing re-ranking approaches on QUAERO's visual concepts image dataset.

as well the state of the art result reported in [89].

4.5.4 Summary on multiple random projections for image re-ranking.

To summarize, the re-ranking approach based on the idea of this chapter can give better results than the re-ranking approach in Chapter 3. Moreover it can reduce the computational time by an order of magnitude (about 14 times faster) and have a complexity linear according to the number of images both in time and memory.

4.6 Conclusions

In this chapter, we proposed a new method for discovering the discriminant dependencies between image features, and encode them through Histograms of Jumping Emerging Patterns(HJEPs). The key characteristics of the proposed approach lies in (i) *random projections* making the computation tractable while avoiding over-fitting, (ii) *local thresholding* of feature distribution allowing to represent images by compact sets of meaningful transactions, and, finally, (iii) in the introduction of histograms of positive and negative jumping emerging patterns that are shown to be a very efficient and compact way to represent images. The proposed approach has been validated on fours different datasets, leading to state-of-the-art results on each one of these datasets. The idea of multiple random multiple random projections can be also used for image re-ranking.

Chapter 5

Finding Groups of Duplicate Images

Contents

5.1 Introduction 85
5.2 Related Work 87
5.3 Method 88
5.3.1 Coding image and transactions of visual properties 90
5.3.2 Mining groups of similar images 91
5.4 Experiments 92
5.4.1 Datasets 92
5.4.2 Representing images as transactions of itemsets 93
5.4.3 Duplicate detection 95
5.5 Conclusions 97

5.1 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: (i) the number of images on the Internet is unimaginable and (ii) 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 this chapter, our interest is thus in two related questions: (i) how high is this redundancy and, (ii) 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 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. However, these method are not efficient for finding groups of duplicates as each image of the dataset has to be considered in turn as a query, and the duplicates for each image query 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 since it is not neccessary 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 because these algorithms are designed for extracting information such as itemsets from extremely large datasets. In particular, we show in this chapter 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: (i) we establish the link between mining closed itemsets and the discovery of groups of duplicate images, (ii) propose a novel image representation built in terms of data mining transaction, allowing us to make use of efficient data mining algorithms, and (iii) provide a new dataset consisting of one million images to experiment on the problem of detecting groups of duplicates on large-scale image datasets.

This chapter is organized as follows: Section 5.2 discusses previous related works while Section 5.3 describes our approach by explaining how to encode images as data mining transactions and how to obtain the groups of duplicates. Section 5.4 provides the experimental validation of our ideas. We first validate our image representation in an image retrieval scenario, then evaluate the quality of the group detection as well as the complex-

ity of our approach in terms of efficiency and memory usage in large scale experiments. Finally, Section 5.5 concludes the chapter.

5.2 Related Work

As explained in the introduction, finding groups of similar images can be seen as a clustering problem. As an illustration, [11] 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, [50] 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 [63], assuming relevant images belonging to large clusters. Closer to our work, [16] 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 [37]. 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 [17, 20, 25, 79]. However, as explained in the introduction (Section 5.1), 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 [35] and [100]. In [100], 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 their top-K most significant bits are identical and the *Hamming distances* between their 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, [35] 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 possible 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 counts are 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 [104], a mining process identifies salient terms from textual descriptions of search results. In [108], 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 [80], 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 chapter.

5.3 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 5.1. 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 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 (Section 1.3.1), we refer to the visual attributes as *items*, and groups of items as *itemsets*. An image is referred to as a *transaction*, a set of items being processed together.

From our definition that groups of duplicates are groups of images in which the images share a large amount of attributes, we can link the problem in finding long frequent itemsets. A long frequent itemset is a list containing many visual attributes where the list

$Database$	
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

$FrequentItemset$	fr	l	$\mathcal{S}$
$\{a_1\}$	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 $F_{min} = 3$

Figure 5.1: Example of frequent itemsets and the notations. I: images, a: visual attributes, fr: number of occurrences of an itemset in the database, l: length of an itemset, $\mathcal{S}$: the support of the itemset (set of transactions containing the itemset).

occurs in a set of images. The set of transactions (i.e. the group of images) containing these itemsets are considered as groups of duplicates.

In order to find long frequent itemset, algorithms for finding frequent *closed itemsets* are suitable. From the definition of closed frequent itemsets described in Section 1.3.4, a closed frequent itemset is the longest frequent itemset in its equivalence class where an equivalence class is defined as a set of itemsets in which all of the itemsets have the same support (appear in the same set of images). Since we are interested in finding long frequent itemsets, we consider only the closed frequent itemsets. In terms of efficiency, mining closed frequent itemsets is faster than mining all frequent itemsets.

Maximal frequent itemsets described in Section 1.3.3 are the longest among all frequent itemsets. A simple definition is that none of its supersets is frequent. Since we are interested in finding long itemsets, one interesting question could be 'Will *maximal frequent itemsets* be more suitable than closed frequent itemsets?'. The answer is 'no' because maximal frequent itemsets mining can miss to detect some groups of duplicates. We demonstrate a toy example in which maximal frequent itemsets mining fails to detect a group of duplicates. The three images in Figure 5.2 are supposed to be a group of duplicates under a specified condition than $F_{min} = 2$ and $L_{min} = 3$. Using maximal frquent itemset mining, one maximal frequent itemset which is {A,B,C,D} can be discovered. The two images containing this itemset which is I_1 and I_2 form a group of duplicates. While I_3 does not contain this itemset and therefore it does not belong to the group. On the other hand, using closed frequent itemset mining, we can discover two closed frequent itemsets which is {A,B,C,D} and {A,B,C}. The first itemset gives a group of duplicates consisting of I_1 and I_2. While the s econd itemset gives another group of duplicates consisting of I_1, I_2, and I_3 which is also a correct group according to our settings. Notice that the length (the number of visual attributes shared among the images containing the itemset) of the first group is 4 and the length of the second group is 3. Therefore, the images in the first group share more similarity among each other compare to the second group.

Only the sets of images containing long closed frequent itemsets are considered as duplicate groups. Therefore, the process for mining closed frequent itemsets is not enough,

$Database$	
I_1	A,B,C,D
I_2	A,B,C,D
I_3	A,B,C E

Figure 5.2: Toy example of three Images that suppose to be a group of duplicates when setting $L_{min} = 3$.

and a post processing process to filter out the short closed frequent itemsets is required.

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.

5.3.1 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) [84, 42, 99] 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 [82], and has been also successful for normalizing BoW in vision tasks as well [20, 70, 84]. Following this line of work, we represent images by the list of their top-K tf-idf weighted visual words. The top-K binarization method has been mentioned in Section 2.3.1.

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 5.3.2), the mining complexity grows exponentially in the number of visual attributes. In practice, K has been determined experimentally (see Section 5.4.2); 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.

5.3.2 Mining groups of similar images

After representing images as transactions of items, we aim at extracting *all* closed frequent itemsets whose length is greater than a given threshold L_{min} (the number of sharing attributes has to be high enough to be considered as duplicates) and whose frequency is greater than $F_{min} = 2$ (the itemset has to be appeared in at least two images). 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 [13] restrict the mining to specific itemsets like the free or the closed patterns [76]. 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 [13]. Therefore, mining closed frequent itemsets using anti-monotonicity and specificities of the closure operator is much more efficient than mining all frequent itemsets [76].

Our mining strategy is based on LCM [93], which is one of the most efficient algorithms for mining closed frequent itemsets. As we are looking for groups of duplicates, each itemset has to be supported by at least two images ($F_{min} = 2$). Even if this value seems low and may lead to a large number of itemsets, it still 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 5.4).

Since we are interested in finding long itemsets more than frequent itemsets (*i.e.* l_{min} has to be set to a high value whearas F_{min} has to be set to 2 since there can be groups of 2 duplicate images), it might be more efficient if we can impose the length constraint during the mining process rather than the frequency constraint which is used in closed frequent itemset mining.

One solution could be to transpose the data [74] by considering each attribute as a transaction not an item, and each image as an item not a transaction. When transposing the data, the frequency becomes the length respect to the original data and we can set F_{min} to be the value as what we wanted to set for the L_{min} in the original data representation. Mining closed frequent itemsets in the transposition data has been discussed in more detail in Chapter 3 (Section 3.3.2). Once the data is transposed, it might seems that we can

fully make use of closed frequent itemset mining (*i.e.* We can set a high F_{min} threshold). However, due to the fact that the search space grows exponentially with the the number of items [38] and since the items in the transposed data are refer to images, the search space would grow exponentially with the number of images. This is not desirable because the number of images is in a scale of millions while the number of attributes is in a scale of tens.

5.4 Experiments

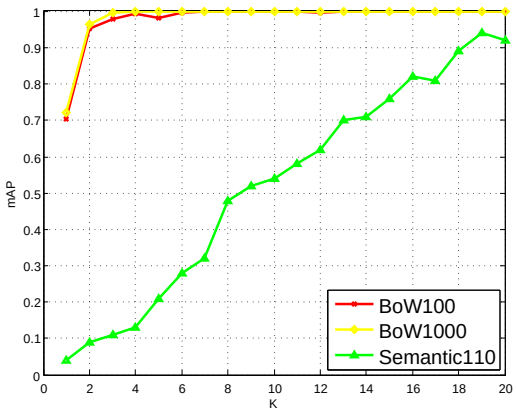
This section presents the experimental validation of the proposed approach. We first describe the datasets used for the experiments, validate the proposed binary representation by comparing it with a baseline representation, and, finally, give quantitative and qualitative results for large-scale groups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

5.4.1 Datasets

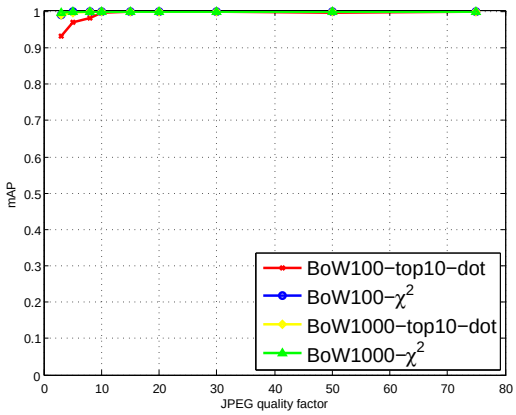
We have created the *One million random web images database* to use in our experiments. Unlike other datasets with a small set of specific queries, we used about 100,000 random alphabet strings as query inputs to Google image search engine. The images in this database are very diverse including many 'man-made objects', 'sceneries', 'logos', 'sketches', 'animals'.

The *Copydays dataset* was proposed in [25] 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.

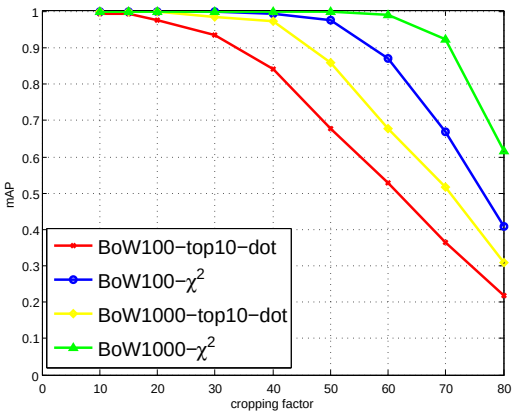
Each of the 18 attack sets consists of exactly 157 images, i.e. each original image has been transformed only once for each attack. Together with the original image set, there are 2,983 images in total.



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

5.4.2 Representing images as transactions of itemsets

We have introduced (Section 5.3) 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 retrieval 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

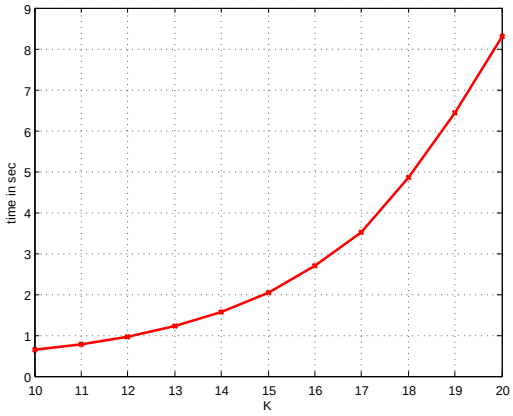
representations. Regarding the representation by itemsets, the items representing an image are the top-K visual words, after tf-idf weighting (see Section 5.3). Itemsets representations for images are given by binary vectors in which '1' (respectively '0') means that the corresponding visual word is (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 [85] and represent image by the K semantic attributes having the highest scores.

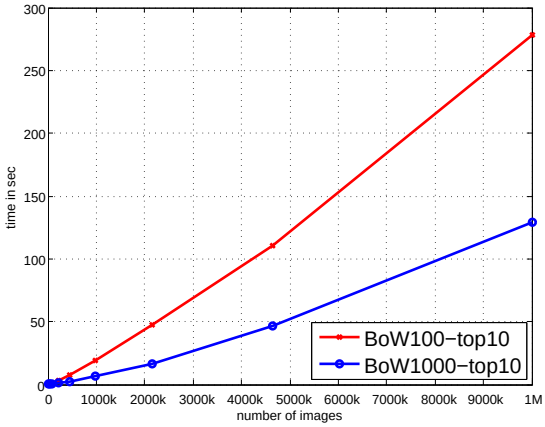
We first present some experiments done on with the JPEG75 attack (this attack represents typical duplicates we can find on the Internet). Three conclusions can be drawn from these experiments, whose results are given in Figure 5.3a. First, representing images by their top visual words performs better 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 5.3b. 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 5.3c. Larger vocabulary performs better again, and the binary representation is almost as good as the BoW one, for cropping factors below 30%.

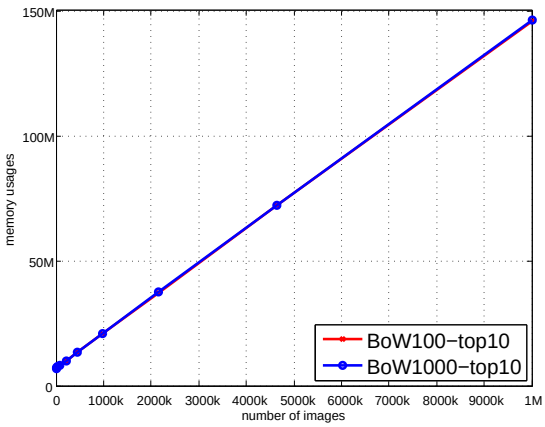
In conclusion, these experiments demonstrate that this representation is sufficient for detecting near duplicate images, while being very compact (each image is encoded by only about 13 bytes). Furthermore, as this representation is made of lists of items, it can be used efficiently for finding frequent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



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



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



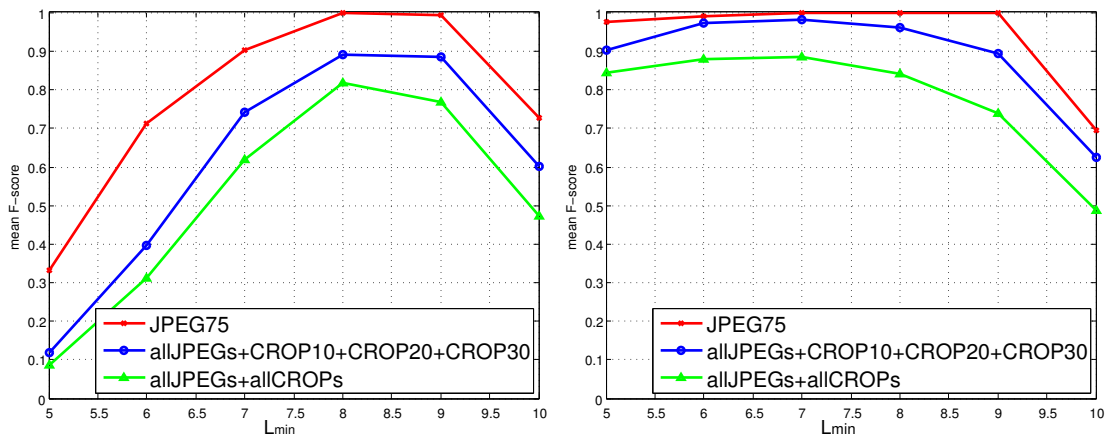
(c) Memory usage as a function of the size of the database.

Figure 5.4: Duplicate detection: quantitative results.

5.4.3 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



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

Figure 5.5: Mean F-score as a function of L_{min}

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 [55]. 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 L_{min} (defined in Section 5.3.2). This is actually the number of attributes two images have to share for being considered to be duplicates. Figure 5.5 shows the F-Score as a function of L_{min}, for representations made from 100 and 1,000 visual words dictionaries. $L_{min} = 7$ 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 5.4b and Figure 5.4c 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 5.4a 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 [46] with its fastest setting, requires 1.5 milliseconds per query for searching in 1,000,000 images. This 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.

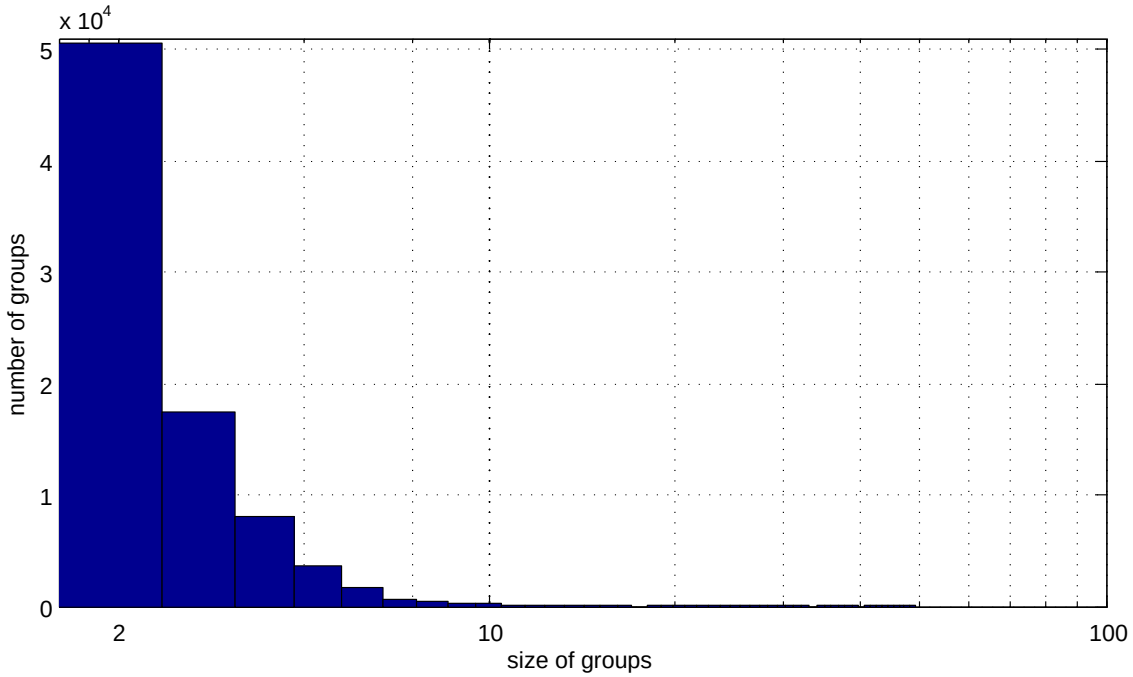


Figure 5.6: Size statistics found on the One million random web images database

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 5.7 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 5.6. However, as we do not know how many actual duplicates are within the dataset, we cannot measure the quality of those groups.

5.5 Conclusions

We have 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.

Figure 5.7: Some of the groups of duplicate/similar images found on the One million random web images database

Chapter 6

Conclusions and Future work

The fact that pattern mining algorithms can efficiently discover interesting combinations of items (patterns) from transaction databases, motivated us to apply such algorithms in computer vision tasks. We made use of them in two aspects. The first aspect is to discover feature dependencies (by seeing images as transactions and image features as items) and encode the information to obtain better image representations (Chapter 2 and Chapter 4). While the second aspect is to benefit from the efficiency of the algorithms to quickly find correlations between images (Chapter 3 and Chapter 5).

We believe that we have achieved interesting results through our exploration. Along this thesis, we conducted several sets or experiments, in order to understand several key issues when applying pattern mining to computer vision. Moreover, the extensive sets of experiments allow us to validate our new proposals. The most important aspects of each chapter, are summarized from Section 6.1 to Section 6.4. While in Section 6.5, we present a general conclusion over the whole study. We discuss as well key issues which needed to be considered when applying pattern mining techniques in computer vision, along with some of our perspectives.

6.1 Preliminary Investigations

We investigated three types of feature dependencies including: (i) *co-occurrences*, (ii) *relative differences*, and (iii) *spatial relations*. Starting from the Bag of Words (BoW) image representation, which is a real valued feature vector, we designed three methods for capturing each type of feature dependencies resulting in three high-order image representations. These representations are compared in an image classification framework.

Co-occurrences The first method captures *co-occurrences* of visual words by using *Emerging Pattern mining* to select the visual words which co-occur frequently in the positive class and rarely occur in the negative class. The emerging patterns are used as

new image features. In order to transform the real valued features, we investigated 3 binarization schemes: (i) fix threshold, (ii) q-quantile, and (iii) top-K bins, and showed that the top-K binarization gave the best results. We analyzed and discussed several advantages of the top-K binarization scheme. Later, we used this binarization scheme in all of the three main chapters in this thesis, Chapter 3, Chapter 4, and Chapter 5. Using this method, we obtained 38.1% AP compared to the baseline 58.1% AP. Although the result is lower, we believe the result is reasonable, considering that a large amount of information has been lost during the binarization process. Solving the problem of binarization loss can surely improve the result.

Relative differences The second method captures *relative differences* between visual word pairs by computing the differences between the values of each visual word feature, and uses the differences as a new feature. Since the number of pairs can be very high, we investigated two methods to select the pairs: (i) according to t-test and (ii) random selection. Random selection gave better results than t-test and comparable results to using all pairs was achieved. This method gave 60.0% mAP compared to the baseline of 58.1% mAP. Although uninvestigated feature selection techniques could be better than random selection, the result would still be bounded to the slight improvement of using all pair features. Therefore, we did not continue on the direction of feature selection. Nevertheless, the results of relative differences between pairs encouraged us to find a method to encode relative differences among groups of visual words; not only pairs.

Spatial relation The third method captured *spatial relation* between visual words in local regions. The method begins at the stage when local features are mapped to visual words, which is the stage before calculating the histogram of visual words in the BoW image representation. For each image, we encoded multiple data mining transactions, in which each transaction corresponds to a local-region in an image. We used emerging pattern mining to extract discriminative sets of spatial relation combinations of visual words. Since an image contains multiple transaction, a single pattern can appear multiple times in an image. An image is represented as a histogram in which the bins correspond to the presence or absence of the patterns. We obtained very bad results using this method. We realized that the way the items were encoded, the set of all possible items was in a scale of several millions. Due to this very large space, it was not possible to extract meaningful combinations of these items. Besides the mistake of using high number of items, pattern selection which has been shown to be very important [33] was also missing in our work.

We showed that feature dependencies contain valuable information and it is worth to further investigate the possibilities to encode them. Later, in Chapter 4, we proposed a method that captures both feature co-occurrences as well as feature relative differences together, and also solves the problems encountered in the preliminary investigation in-

cluding binarization loss, high dimensionality, and redundant patterns.

6.2 Image Re-ranking

We proposed a method to re-rank the images obtained from text-based image search engines. Our method relies on the assumption that the majority of the images returned by the initial text-based retrieval are relevant to the query, and that relevant images are visually similar to some other relevant images (*i.e.* they form clusters). The top-K binarization method described in Section 2.3.1 is used for transforming image features to binary pattern mining transactions. Closed frequent itemsets are mined from the set of transactions. The closed frequent itemsets are likely to be extracted from the positive images, so the images can be ranked based on the amount of closed frequent itemsets found in each of them. We showed that weighting the closed frequent itemsets based on the initial ranking position of the images supporting the itemsets can improve the results of the non-weighted scheme. The search space for finding closed frequent itemsets grows exponentially with the number of items (image attributes) and since in our problem, there are more image attributes compared to the number of images, we proposed to mine closed frequent itemsets from the transposed database which is much more efficient. Two datasets including **INRIA Web Queries dataset** [53] and **Fergus dataset** [31] were used to validate our approach. The re-ranking gave significant improvement over the original text based ranking (13% mAP improvement on INRIA Web Queries dataset and 33% mean precision at 15% recall improvement on Fergus dataset.) Our approach takes on average less than 2 seconds to re-rank about 200 images. It compares favorably with state-of-the-art results on the two datasets.

6.3 Histograms of Jumping Patterns for Image Classification and Object Recognition

Our preliminary investigation in Chapter 2 gave us useful insights which led to our novel image representation that captures both feature co-occurrences as well as the relative differences between features. This representation is both compact and discriminative, and can be used for image classification or object detection/recognition.

To solve the problem of binarization loss in Chapter 2, we projected the original features to multiple low dimension subspaces using *random projections*. For each subspace, we applied the top-K binarization to convert the low dimension features into pattern mining transactions. Since we are using top-K binarization, not only the occurrences of features but also the relative differences between the features are captured. Positive and negative emerging patterns are mined. Rather than using each pattern as an individual feature

(leads to the problem of very high dimensionality in Chapter 2), we used the statistics of the amount of patterns found in each image. Each projection gives two bin histograms (*i.e.* the count of positive patterns and the count of negative patterns). The histograms of all projections are concatenated as the final image representation. Our approach has been applied to various image features such as 'BoW', 'HOG', 'LTP' showing significant improvement over the baseline features. Moreover, we achieved state-of-the-art results on several classification benchmarks including Daimler Pedestrian Classification, Oxford Flowers Classification, KTH Texture Categorization, and PASCAL VOC2007). The results can be seen in Chapter 4 (Section 4.4).

The idea of doing multiple random projections and binarization allows us to encode images as data mining transactions without losing so much information during the binarization process. We have applied similar techniques to *image re-ranking* and showed interesting results in Chapter 4, Section 4.5. The idea is to apply the main idea of chapter 4 which is to: (i) project the high dimensional feature space into several low dimensional sub-spaces by multiple random projections, (ii) extract patterns in each sub-space, and (iii) merge the results. Since we reduced the number of items in the transactions, it is not necessary to transpose the data as done in Chapter 3. We can mine closed frequent itemsets in the non-transposed data, which allows the algorithm to have a complexity linear according to the number of images. Obtaining similar results as the method presented in this chapter (Chapter 3), the new approach takes on average 0.15 seconds for re-ranking about 200 images which is about 14 times faster. Moreover, we improved the results further by 2% mAP using a process to select the best random projections.

It would be interesting to investigate the application of this technique in other tasks not only tasks in computer vision but also tasks from other domains. Another extension could be to explore different projection techniques which might be better than simple random projections.

6.4 Finding Groups of Duplicate Images in Very Large Datasets

We addressed the problem of detecting groups of duplicates in large-scale unstructured image datasets such as the Internet. We proposed an efficient approach based on the search of *closed patterns*. The key idea of the approach consists in encoding images as a compact list of very few visual attributes. Our encoding is based on tf-idf weighting BoW followed by top-K binarization. We showed that this compact representation is sufficient for detecting duplicates in an image retrieval scenario. The duplicates are the ones that share most of the attributes with each other. This corresponds to the discovery of long closed frequent itemsets. We validated our approach on a new dataset of one million Internet images obtained with random searches on 'Google' image search. Using the proposed method, we found more than 80 thousand 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 the complexity is linear in time and space with respect to the number of images. Furthermore, the approach does not need (to build or use) any precomputed indexing structure.

Future work could be to use and combine other types of features; not only BoW representations. Since the images are very diverse, different type of features might be more suitable for different kind of images. As the technique can be seen similar to clustering techniques, it might be possible to extend this work for large scale image clustering, and not restricted only to the detection of duplicates groups.

One million random web images database. We also introduced a new database. 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. Moreover, the numbers of duplicate groups will 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 about 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, ranging from man-made objects, sceneries, logos, sketches, animals.

6.5 Conclusions and perspectives

Pattern mining in the domain of data mining is the field than can extract interesting combinations of items (features) from large quantities of data. Over two decades, several types of patterns and specific pruning techniques to search for patterns in extremely large combination spaces have been proposed. Computer vision is a domain related to the development of artificial intelligence visual systems giving the ability to machines to see and to interpret as humans. Typically, machines take input as real valued feature vectors and use machine learning algorithms to interpret the images. The drawback of these approaches is that they do not incorporate the dependencies between each individual feature. Obviously, incorporating feature dependencies is nontrivial, since images usually have high dimensional features and the space of feature combination is extremely large. Our main idea in this thesis is to use pattern mining techniques to extract useful feature combinations.

Applying pattern mining techniques are not straightforward. First, the algorithms are designed for discrete value attributes, while the image features are real valued. Therefore, binarization is required. Binarization leads to a large amount of information loss. We proposed a binarization scheme based on top-K and discussed several advantages of it. However, how to binarize image features is still an open question that needs to be further investigated. Second, the algorithms can efficiently process millions of transactions, but they can be very slow when the number of items are more than hundreds. This is obvious, since the combination of items grows exponentially with the number of items. Image features typically have very high dimensions in order to capture the large range of variation from *e.g.* viewpoint changes, lightning conditions, size and pose, intra-class variations. We observed that pattern extraction is very slow when the number of items is more than hundreds. In order to benefit pattern mining techniques, we need to find a compact binary representation which has to be robust against the large range of variations. We proposed to project the features to lower dimensions using multiple random projections followed by top-K binarization. This solves the problem of binarization loss as well as the mining complexity in the same time. Future work can be to investigate different kinds of dimensionality reduction techniques.

How to make use of the patterns is the question coming after the extraction process. Typical pattern mining process takes local constraints such as the minimum frequency or the growth rate. The patterns are independent to each other and only needs to fulfill the local constraints. This leads to the problem of having too many patterns redundant to each other. The number of patterns can be several millions which is obviously not possible to use each pattern as a new feature. The typical solution is to use a pattern selection process to first filter out redundancy and apply global constraints such as 'the coverage' or 'prediction accuracy' to select a small set of patterns before use these patterns as the new features. In this thesis, we proposed another solution by using the statistics of the number of patterns instead. This way, our representation is a compact representation that combines the information of several local patterns. This solution is very fast compare to pattern selection with global constraints (*i.e.* one by one pattern selection). We believe that there are more sophisticated methods to compute pattern statistics and can give better results than simply count of the number of patterns as used in this thesis. This could be another future work direction.

Besides feature combinations, we showed that we can find correlations between images directly (*i.e.* we do not encode new image features and learn class models). Pattern mining algorithms are more less linear to the number of transactions (images) and if we can have compact image representations, we could be able to find similarity between images very efficiently. We have shown in Chapter 5 that groups of duplicate images can be discovered from a database of one million images under 3 minutes. This is possible because our image representation is very compact. We have shown that encoding each image by only 10 visual words is sufficient for detecting duplicates. A very interesting

direction is to combine our method in Chapter 4 by doing multiple random projections and combine the detected groups. This might lead to an efficient way for image clustering.

In summary, we believe that pattern mining is interesting for computer vision, and are also confident that there is a wide range of opportunities for improvement, since this discipline has been until now largely unexplored within the computer vision field. While our contributions have mainly resided in the application of itemset mining in order to provide original, practical and performance solutions in a wide variety of computer vision tasks, it remains as future work the investigation of several other data mining techniques. Sequential and graph mining are only a couple of examples of possible implementations in computer vision tasks, that would be worth exploring in depth in the future.

List of Figures

1.1 Image classification for a wide range of applications. 10
1.2 Object detection is useful in wide range of applications. 11
1.3 A mobile application, named 'Google Googles', returns similar images based on visual content associated with useful information to the user.*(Figure from Google.)* 12
1.4 Two clusters of 'jaguar' images (the car brand and the animal). Image clustering resolves ambiguity occurring from homonymous queries in text based image searches. 13
1.5 Concept of Bag of Words representation *(Figure from Google Image)*. By just looking at the words without taking to account the grammar and word orders, we are still able to tell that the document on the left is related to 'health' while the document on the right is related to 'business'. 16
1.6 Toy example of constructing a three-level spatial pyramid [57]. The image has three feature types, indicated by circles, diamonds, and crosses. The image is subdivided at three different levels of resolution. Next, for each level of resolution, the features that fall in each spatial bin are counted. Finally, the histograms computed at each levels are concatenated. 17
1.7 Example of closed frequent itemsets with $F_{min} = 2$ and the notations. $\mathcal{T}_i$: transactions, i_k: items, $\mathcal{X}_j$: itemsets, $l(\mathcal{X}_j)$: length of each of itemset $\mathcal{X}_j$, $S(\mathcal{X}_j)$: support of each itemset $\mathcal{X}$, and $fr(\mathcal{X}_j)$: frequency of each itemset $\mathcal{X}_j$. 20

2.1 Toy example of the three types of binarization methods. All figures on the left represent the binarizations of a flat distribution histogram. All figures on the right represent the binarizations of a distribution histogram with high peaks. The green bars are the bins in which their feature indexes are used as transaction items. The red bars are the discared bins. For example, the transaction encoded from the histogram in Figure (c)-left is {2,6,8,9}. 31
2.2 Average Precision and coverage according to the number of patterns. . . . 32
2.3 A toy example to encode a BoW vector 3,1,4 to pair difference representations. 33
2.4 T-Test distribution 34

2.5 Positive and negative image probability density function over the count difference. 35
2.6 Performance according to the number of pairs. 35
2.7 Local region filter. One data mining transaction consists of eight data mining items. Each item consists of three components: (i) the location of the neighboring visual word (red), (ii) the center visual word (blue), and (iii) the neighboring visual word (black). 36
2.8 Example images from the Pascal VOC 2007 dataset. 38
2.9 Performance according to number of pair features. 39

3.1 Toy example of the original transaction database in (a) and the transposition of the transaction database in (b). 49
3.2 Toy example illustrating the re-ranking of images according to the count of frequent itemsets. (a) shows that it is likely that the relevant images I_1, I_2 and I_4 share many similar visual attributes to each others whereas the two non-relevant images I_3 and I_4 share very few visual attributes with the other images. Shown in (b), Frequent itemsets are likely to be found in the relevant images and so we can use the number of frequent itemsets found in each images as a criterion to re-rank the images. As shown in (c) all the relevant images are ranked before the non-relevant images. 49
3.3 Hypergraph of toy example in Figure 3.2(b) and it corresponding incidence matrix. 50
3.4 Influence of the parameters on the behavior of the algorithm. 53
3.5 Landmark re-ranking results 58
3.6 Logos, maps, and flags re-ranking results 59
3.7 Generic object/scene classes re-ranking results 60
3.8 Celebrities re-ranking results 61
3.9 Fergus re-ranking results (a) 62
3.10 Fergus re-ranking results (b) 63

4.1 Overview of the approach: feature histograms (*e.g.* bag-of-words) are randomly projected before being binarized (thresholded). Each image is represented, for each projection, by a set of items. The final representation is the distribution of the Jumping Emerging Patterns. See text for more details. 67
4.2 Toy example showing that after two random projections followed by top-K ($K = 2$) binarizations, relative relations between the BoW features are preserved (compare the discovered relation with the original BoW relation) and implicitely encoded within the representaion. 71
4.3 Toy example showing how patterns are obtained. 72
4.4 Influence of the parameters on the behavior of the algorithm. 75

4.5 Example images from the KTH TIPS 2a texture dataset [15] 77
4.6 Some samples from the Daimler pedestrian classification dataset [26]. . . 79
4.7 Pedestrian recognition on the non-occluded testset of Daimler pedestrian dataset. 80
4.8 Example images from the Oxford-Flowers 17 database. 80

5.1 Example of frequent itemsets and the notations. I: images, a: visual attributes, fr: number of occurrences of an itemset in the database, l: length of an itemset, $\mathcal{S}$: the support of the itemset (set of transactions containing the itemset). 89
5.2 Toy example of three Images that suppose to be a group of duplicates when setting $L_{min} = 3$. 90
5.3 Image retrieval experiments: performance of the proposed representation and of the baseline representation, for two vocabularies (100 and 1,000 visual words). 93
5.4 Duplicate detection: quantitative results. 95
5.5 Mean F-score as a function of L_{min} 96
5.6 Size statistics found on the *One million random web images database* 97
5.7 Some of the groups of duplicate/similar images found on the One million random web images database 98

List of Tables

2.1 Comparison of different binarization methods. 30
2.2 Comparison of pair difference features. 34
2.3 Comparison of pair difference features. 37
2.4 Image classification results on the 'aeroplane' class validation set. 37
2.5 Number of images in each subset. 38
2.6 Full results on PASCAL VOC 2007 dataset. 40

3.1 Comparison of different weighting schemes. 54
3.2 Computational time for different queries on a single core machine. 55
3.3 Comparison to other existing re-ranking approaches. 56
3.4 Fergus re-ranking results 57

4.1 Comparison of linear and non-linear classifiers (mAP). 76
4.2 mean accuracy(mA) on the KTH-TIPS2a dataset. 78
4.3 Comparison with state-of-the-art results on the Oxford Flower 17 dataset. 81
4.4 Comparison with state- of-the-art approaches on the PASCAL VOC 2007 dataset. 81
4.5 Comparison of computational time between the method of transposing data (Tr.) and the method of multiple random projections (Rp.) on the validation data. 83
4.6 Comparison of the re-ranking approach in Chapter 3 and the re-ranking approach based on multiple-random projections 83
4.7 Comparison to other existing re-ranking approaches on QUAERO's visual concepts image dataset. 84

Bibliography

[1] R. Agrawal, T. Imieliński, and A. Swami. Mining association rules between sets of items in large databases. In *ACM SIGMOD Record*, volume 22. ACM, 1993.

[2] A. D. Bagdanov. Color attributes for object detection. In *CVPR*, 2012.

[3] C. Barat, C. Ducottet, E. Fromont, A.-C. Legrand, and M. Sebban. Weighted symbols-based edit distance for string-structured image classification. In *Machine Learning and Knowledge Discovery in Databases*, pages 72–86. Springer, 2010.

[4] R. J. Bayardo Jr. Efficiently mining long patterns from databases. In *ACM Sigmod Record*, volume 27. ACM, 1998.

[5] P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegman. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. *PAMI*, 19(7):711–720, 1997.

[6] N. Ben-Haim, B. Babenko, and S. Belongie. Improving web-based image search via content based clustering. In *CVPR Workshop*, 2006.

[7] T. Berg and D. Forsyth. Animals on the web. In *CVPR*, 2006.

[8] T. L. Berg and A. C. Berg. Finding iconic images. In *CVPR Workshop*, 2009.

[9] B. Bringmann, S. Nijssen, and A. Zimmermann. Pattern-based classification: A unifying perspective. In *ECML/PKDD Workshop on From Local Patterns to Global Models (LeGo)*, 2009.

[10] B. Bringmann and A. Zimmermann. The chosen few: On identifying valuable patterns. In *ICDM*, 2007.

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

[12] Y. Cai, N. Cercone, and J. Han. An attribute-oriented approach for learning classification rules from relational databases. In *Data Engineering, 1990. Proceedings. Sixth International Conference on*. IEEE, 1990.

[13] 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*. Springer, 2005.

[14] Z. Cao, Q. Yin, X. Tang, and J. Sun. Face recognition with learning-based descriptor. In *CVPR*, 2010.

[15] B. Caputo, E. Hayman, and P. Mallikarjuna. Class-specific material categorisation. In *ICCV*, 2005.

[16] 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.

[17] E. 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*, 1998.

[18] K. Chatfield, V. S. Lempitsky, A. Vedaldi, and A. Zisserman. The devil is in the details: an evaluation of recent feature encoding methods. In *BMVC*, 2011.

[19] J. Chen, S. Shan, C. He, G. Zhao, M. Pietikainen, X. Chen, and W. Gao. WLD: A Robust Local Image Descriptor. *PAMI*, 32(9):1705–1720, 2010.

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

[21] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005.

[22] C. Dance, J. Willamowski, L. Fan, C. Bray, and G. Csurka. Visual categorization with bags of keypoints. In *ECCV Workshop*, 2004.

[23] C. Desai, D. Ramanan, C. Fowlkes, and U. C. Irvine. Discriminative models for multi-class object layout. In *ICCV*, 2009.

[24] G. Dong and J. Li. Efficient mining of emerging patterns: discovering trends and differences. In *KDD*, 1999.

[25] M. Douze, H. Jégou, H. Sandhawalia, L. Amsaleg, and C. Schmid. Evaluation of gist descriptors for web-scale image search. In *CIVR*, 2009.

[26] M. Enzweiler, A. Eigenstetter, B. Schiele, and D. M. Gavrila. Multi-cue pedestrian classification with partial occlusion handling. In *CVPR*, 2010.

[27] M. Everingham, L. V. Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results. http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html, 2007.

[28] M. Everingham, L. Van Gool, C. Williams, J. Winn, and A. Zisserman. The pascal visual object classes challenge 2007. In *2th PASCAL Challenge Workshop*, 2009.

[29] A. C. F. Schroff and A. Zisserman. Harvesting image databases from the web. In *ICCV*, 2007.

[30] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *PAMI*, 32(9):1627–1645, 2010.

[31] R. Fergus, L. Fei-Fei, P. Perona, and A. Zisserman. Learning object categories from google's image search. In *ICCV*, 2005.

[32] R. Fergus, P. Perona, and A. Zisserman. A visual category filter for google images. In *ECCV*, 2004.

[33] B. Fernando, É. Fromont, and T. Tuytelaars. Effective use of frequent itemset mining for image classification. In *ECCV*, 2012.

[34] D. Fogaras, B. Rácz, K. Csalogány, and T. Sarlós. Towards scaling fully personalized pagerank: Algorithms, lower bounds, and experiments. *Internet Mathematics*, 2(3):333–358, 2005.

[35] J. Foo, J. Zobel, and R. Sinha. Clustering near-duplicate images in large collections. In *MIR Workshop*, MIR. ACM, 2007.

[36] M. Fritz and B. Schiele. Decomposition, discovery and detection of visual categories using topic models. In *CVPR*, 2008.

[37] 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*. IEEE Computer Society, 1999.

[38] F. Geerts, B. Goethals, and J. Van den Bussche. A tight upper bound on the number of candidate patterns. In *ICDM*, 2001.

[39] A. Gilbert, J. Illingworth, and R. Bowden. Scale invariant action recognition using compound features mined from dense spatio-temporal corners. In *ECCV*, 2008.

[40] A. Gionis, P. Indyk, and R. Motwani. Similarity search in high dimensions via hashing. In *Proceedings of the 25th International Conference on Very Large Data Bases*, VLDB, 1999.

[41] D. Grangier and S. Bengio. A discriminative kernel-based model to rank images from text queries. *PAMI*, 30:1371–1384, 2008.

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

[43] W. H. Hsu, L. S. Kennedy, and S.-F. Chang. Reranking methods for visual search. *Multimedia, IEEE*, 14:14–22, 2007.

[44] S. U. Hussain and B. Triggs. Feature sets and dimensionality reduction for visual object detection. In *BMVC*, 2010.

[45] S. U. Hussain and B. Triggs. Visual recognition using local quantized patterns. In *ECCV*, 2012.

[46] H. Jégou, M. Douze, and C. Schmid. Product quantization for nearest neighbor search. *PAMI*, 33(1):117–128, 2011.

[47] H. Jégou, F. Perronnin, M. Douze, J. Sánchez, P. Pérez, and C. Schmid. Aggregating local image descriptors into compact codes. *PAMI*, 34(9):1704–1716, 2012.

[48] Y. Jing and S. Baluja. Visualrank: Applying pagerank to large-scale image search. *PAMI*, 30:1877–1890, 2008.

[49] G. L. L. W. Jinyan Li. Mining statistically important equivalence classes and delta-discriminative emerging patterns. In *KDD*, 2007.

[50] L. Kennedy and M. Naaman. Generating diverse and representative image search results for landmarks. In *WWW*. ACM, 2008.

[51] F. S. Khan, J. van de Weijer, and M. Vanrell. Top-down color attention for object recognition. In *ICCV*, 2009.

[52] A. Knobbe, B. Crémilleux, J. Fürnkranz, and M. Scholz. The lego approach to data mining. In *ECML/PKDD Workshop on From Local Patterns to Global Models (LeGo)*, 2008.

[53] J. Krapac, M. Allan, J. Verbeek, and F. Jurie. Improving web-image search results using query-relative classifiers. In *CVPR*, 2010.

[54] N. Kumar, A. C. Berg, P. N. Belhumeur, and S. K. Nayar. Attribute and simile classifiers for face verification. In *ICCV*, 2009.

[55] B. Larsen and C. Aone. Fast and effective text mining using linear-time document clustering. In *KDD*, 1999.

[56] S. Lazebnik, C. Schmid, and J. Ponce. A maximum entropy framework for part-based texture and object recognition. In *ICCV*, 2005.

[57] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *CVPR*, 2006.

[58] L. Lhote, F. Rioult, and A. Soulet. Average number of frequent (closed) patterns in bernouilli and markovian databases. In *ICDM*, 2005.

[59] H. Li, J. Li, L. Wong, M. Feng, and Y.-P. Tan. Relative risk and odds ratio: A data mining perspective. In *Proceedings of the twenty-fourth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*. ACM, 2005.

[60] W. Li and M. Fritz. Recognizing materials from virtual examples. In *ECCV*, 2012.

[61] H. Ling and S. Soatto. Proximity distribution kernels for geometric context in category recognition. In *ICCV*, 2007.

[62] D. Liu, G. Hua, P. Viola, and T. Chen. Integrated feature selection and higher-order spatial feature extraction for object categorization. In *CVPR*, 2008.

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

[64] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004.

[65] T. M. Mitchell. Version spaces: A candidate elimination approach to rule learning. In *Proceedings of the 5th international joint conference on Artificial intelligence-Volume 1*. Morgan Kaufmann Publishers Inc., 1977.

[66] N. Morioka and S. Satoh. Building compact local pairwise codebook with joint feature space clustering. In *ECCV*, 2010.

[67] N. Morioka and S. Satoh. Learning directional local pairwise bases with sparse coding. In *BMVC*, 2010.

[68] H. V. Nguyen, L. Bai, and L. Shen. Local gabor binary pattern whitened pca: A novel approach for face recognition from single image per person. In *Advances in Biometrics*, pages 269–278. Springer, 2009.

[69] M.-E. Nilsback and A. Zisserman. Automated flower classification over a large number of classes. In *ICVGIP*, 2008.

[70] D. Nistér and H. Stewénius. Scalable recognition with a vocabulary tree. In *CVPR*, 2006.

[71] P. K. Novak, N. Lavrac, and G. I. Webb. Supervised descriptive rule discovery: A unifying survey of contrast set, emerging pattern and subgroup mining. *Journal of Machine Learning Research*, 10:377–403, 2009.

[72] S. Nowozin, K. Tsuda, T. Uno, T. Kudo, and G. BakIr. Weighted substructure mining for image analysis. In *CVPR*, 2007.

[73] W. H. Organization. Global status report on road safety 2013: Supporting a decade of action. *Geneva, Switzerland: WHO*, 2013.

[74] F. Pan, G. Cong, A. K. H. Tung, Y. Yang, and M. J. Zaki. CARPENTER: finding closed patterns in long biological datasets. In *KDD*, 2003.

[75] N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal. Discovering frequent closed itemsets for association rules. In *ICDT*. 1999.

[76] N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal. Discovering frequent closed itemsets for association rules. In *ICDT*, 1999.

[77] J. Pei, J. Han, R. Mao, et al. Closet: An efficient algorithm for mining frequent closed itemsets. In *ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, volume 4, 2000.

[78] F. Perronnin, J. Sánchez, and T. Mensink. Improving the fisher kernel for large-scale image classification. In *ECCV*. Springer, 2010.

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

[80] T. Quack, V. Ferrari, B. Leibe, and L. J. V. Gool. Efficient mining of frequent and distinctive feature configurations. In *ICCV*, 2007.

[81] F. Rioult, J.-F. Boulicaut, B. Crémilleux, and J. Besson. Using transposition for pattern discovery from microarray data. In *DMKD*, 2003.

[82] G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. In *IPM*, 1988.

[83] G. Sharma, S. ul Hussain, and F. Jurie. Local higher-order statistics (LHS) for texture categorization and facial analysis. In *ECCV*, 2012.

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

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

[86] Y. Su and F. Jurie. Visual word disambiguation by semantic contexts. In *ICCV*, 2011.

[87] X. Tan and B. Triggs. Enhanced local texture feature sets for face recognition under difficult lighting conditions. In *Analysis and Modeling of Faces and Gestures*. 2007.

[88] X. Tan and B. Triggs. Enhanced local texture feature sets for face recognition under difficult lighting conditions. *TIP*, 19(6):1635–1650, 2010.

[89] F. Thollard and G. Quénot. Content-based re-ranking of text-based image search results. In *ECIR*, 2013.

[90] M. Topi, O. Timo, P. Matti, and S. Maricor. Robust texture classification by subsets of local binary patterns. In *ICPR*, 2000.

[91] M. A. Turk and A. P. Pentland. Face recognition using eigenfaces. In *CVPR*, 1991.

[92] T. Uno, T. Asai, Y. Uchida, and H. Arimura. Lcm: An efficient algorithm for enumerating frequent closed item sets. In *FIMI*, 2003.

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

[94] J. van de Weijer, T. Gevers, and A. D. Bagdanov. Boosting color saliency in image feature detection. *PAMI*, 28(1):150–156, 2006.

[95] A. Vedaldi and B. Fulkerson. VLFeat: An open and portable library of computer vision algorithms. http://www.vlfeat.org/, 2008.

[96] A. Vedaldi, V. Gulshan, M. Varma, and A. Zisserman. Multiple kernels for object detection. In *ICCV*, 2009.

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

[98] J. Vreeken, M. Van Leeuwen, and A. Siebes. Characterising the difference. In *KDD*. ACM, 2007.

[99] C. Wallraven, B. Caputo, and A. B. A. Graf. Recognition with local features: the kernel recipe. In *ICCV*, 2003.

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

[101] J. Wang, Y.-G. Jiang, and S.-F. Chang. Label diagnosis through self tuning for web image search. In *CVPR*, 2009.

[102] L. Wang, Y. Wang, T. Jiang, and W. Gao. Instantly telling what happens in a video sequence using simple features. In *CVPR*, 2011.

[103] X. Wang, T. X. Han, and S. Yan. An hog-lbp human detector with partial occlusion handling. In *ICCV*, 2009.

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

[105] N. Xie, H. Ling, W. Hu, and X. Zhang. Use bin-ratio information for category and scene classification. In *CVPR*, 2010.

[106] J. Yang, K. Yu, and T. Huang. Efficient highly over-complete sparse coding using a mixture model. In *ECCV*, 2010.

[107] B. Yao and L. Fei-Fei. Grouplet: A structured image representation for recognizing human and object interactions. In *CVPR*, 2010.

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

[109] M. J. Zaki and C.-J. Hsiao. Charm: An efficient algorithm for closed itemset mining. In *SDM*, 2002.

[110] J. Zhang, K. Huang, Y. Yu, and T. Tan. Boosted local structured hog-lbp for object localization. In *CVPR*, 2011.

[111] D. Zhou, J. Weston, A. Gretton, O. Bousquet, and B. Schölkopf. Ranking on data manifolds. In *NIPS*, 2004.

[112] L. Zhu, Y. Chen, A. L. Yuille, and W. T. Freeman. Latent hierarchical structural learning for object detection. In *CVPR*, 2010.

Représentation des images au moyen de motifs fréquents et émergents pour la classification et la recherche d'images.

Cette thèse a pour but d'améliorer les performances sur différentes tâches de vision par ordinateur en se focalisant sur l'étape de représentation des images. Notre idée clé est d'intégrer des relations entre les descripteurs de l'image à sa représentation originelle, ces relations apportant une information additionnelle par exemple pour discriminer des images. La recherche de telles relations n'est pas simple compte-tenu de la grande combinatoire entre descripteurs.

Nous proposons d'employer des techniques de fouille de données fondées sur la recherche de motifs pour mettre en évidence des relations pertinentes entre les descripteurs d'images. En effet, le fouille de données est appropriée pour l'analyse de grandes quantités de données et la découverte des motifs intéressants traduisant des dépendances, le regroupement de données, la détection d'anomalies. Un premier obstacle à l'emploi de techniques de fouille de données en vision par ordinateur porte sur le *recodage* des descripteurs des images. Ces dernières possèdent usuellement des valeurs réelles alors que les méthodes d'extraction de motifs sont appropriées aux données discrètes. Pour traiter ce problème, nous proposons des techniques fondées sur des seuillages locaux. Le nombre de motifs extraits étant élevés, ceux-ci ne peuvent pas être directement utilisés dans une tâche comme la classification supervisée. Aussi, nous présentons une méthode d'agrégation des motifs permettant d' obtenir une représentation compacte évitant le sur-apprentissage. Les résultats expérimentaux sur de nombreuses bases d'images montrent que notre approche est largement au niveau de l'état de l'art.

Nous montrons que les caractéristiques de la fouille de données sont aussi propices à d'autres tâches de vision par ordinateur. Ainsi, nous avons conçu une méthode de détection de doublons reposant sur l'utilisation de motifs fermés dans de grandes bases d'images. Nous avons testé notre méthode sur une base de 1 million d'images obtenues avec Google image : les doublons sont découverts en moins de 3 minutes. Enfin, nous avons développé une méthode de re-classer d'images fondée sur le nombre de motifs fréquents que chaque image supporte, cette méthode permet d'améliorer le classement initial.

Mot-clés: Vision par ordinateur; Exploration de données; Reconnaissance des formes (informatique); Traitement d'images

Discipline: Informatique et applications
Laboratoire: GREYC CNRS UMR 6072, Sciences 3, Campus 2, Bd Marechal Juin, Université de Caen, 14032 Caen