Unsupervised Detection of Emergent Patterns in Large Image Collections

Umair Mateen Khan

a thesis submitted for the degree of

Doctor of Philosophy

at the University of Otago, Dunedin,

New Zealand.

20 October 2017

Abstract

With the advent of modern image acquisition and sharing technologies, billions of images are added to the Internet every day. This huge repository contains useful information, but it is very hard to analyze. If labeled information is available for this data, then supervised learning techniques can be used to extract useful information. Visual pattern mining approaches provide a way to discover visual structures and patterns in an image collection without the need of any supervision.

The Internet contains images of various objects, scenes, patterns, and shapes. The majority of approaches for visual pattern discovery, on the other hand, find patterns that are related to object or scene categories. Emergent pattern mining techniques provide a way to extract generic, complex and hidden structures in images.

This thesis describes research, experiments, and analysis conducted to explore various approaches to mine emergent patterns from image collections in an unsupervised way. These approaches are based on itemset mining and graph theoretic strategies. The itemset mining strategy uses frequent itemset mining and rare itemset mining techniques to discover patterns. The mining is performed on a transactional dataset which is obtained from the BoW representation of images. The graph-based approach represents visual word co-occurrences obtained from images in a co-occurrence graph. Emergent patterns form dense clusters in this graph that are extracted using normalized cuts. The patterns that are discovered using itemset mining approaches are: *stripes and parallel lines*; *dots and checks*; *bright dots*; *single lines*; *intersections*; and *frames.* The graph based approach revealed various interesting patterns, including some patterns that are related to object categories.

Acknowledgements

First praise is to Allah, the Almighty, on whom ultimately we depend for sustenance and guidance.

Second, my sincere appreciation goes to my supervisors, Brendan McCane, Steven Mills, and Andrew Trotman for all the invaluable advice they have provided over past few years, and for all their help in proof-reading this thesis.

I also wish to thank the Department of Computer Science, its leadership and the staff for providing me with an academic base, which has enabled me to take up this study. I am also indebted to my colleagues at "Graphics and Vision" lab who were fundamental in order to have a good and fun time. I am glad of having them as colleagues, especially Nabeel Younus Khan, Nurazlin Zainal Azmi, Hamza Benanni, and Xiping Fu.

Special thanks, tribute and appreciation to all those their name do not appear here who have contributed to the successful completion of this study.

Finally, I'm forever indebted to my family who, understanding the importance of this work suffered my hectic working hours, to my parents, my wife, my children, and my siblings.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Every day millions of images are generated using various devices such as smart phones, medical imaging equipment, and space exploratory apparatus, etc. These images contain very useful information, indeed *"a picture is worth a thousand words"*. Therefore it is important to have a strategy that can extract information from this massive repository.

Supervised learning approaches aim to infer a function from a labeled training set. The training data is a mapping of input objects to label (Mohri *et al.*, 2012). A machine learning algorithm is then trained to find a match between input objects and output labels. Once training is complete, the label for an unseen object can be discovered. In a classification scenario, the information that maps each object to a particular class is provided as the training data. For a large collection of images, the unavailability of the labeled training data poses a serious limitation in using supervised methods. On the other hand, unsupervised learning approaches aim at finding hidden structures in data and do not need any labeled data for training purposes. In application that aims to discover information from a large collection of images, unsupervised learning approaches are more suitable.

Visual patterns are sets of visual primitives (e.g., interest points, features, or visual words) that co-occur multiple times in an image collection. Visual pattern extraction is a process of finding interesting information from images in a supervised or unsupervised way. Unsupervised approaches for pattern discovery can be divided into bottom-up and top-down strategies (Wang *et al.*, 2014). Usually local features are extracted from images which are quantized to obtain visual words. Images are then represented in the Bag-of-Words (BoW) format (Sivic and Zisserman, 2003).

Frequent itemset mining (FIM) (Agrawal *et al.*, 1993) is a bottom-up approach that has been used for extracting common visual patterns from a collection of images in an unsupervised way. The majority of research efforts conducted in this field focus on extracting patterns relating to objects or scene categories in images (Fernando *et al.*, 2012; Quack *et al.*, 2006, 2008; Yuan *et al.*, 2007a). For these applications, embedding spatial relationships among visual words in the pattern mining process is critical. The downside of this approach is that these methods are unable to catch global patterns, or patterns that are complex and appear at random locations with varying size or shape (Gao *et al.*, 2009). On the other hand, very little effort has been made to discover generic patterns, i.e. patterns that do not necessarily relate to a particular object or scene category but rather represent some structures which are not evident in the image collection. These generic patterns could be a combination of corners, blobs, or textures features that co-occur in many images. We call these generic patterns *emergent patterns*, because they arise from the data without any supervision or interpretation. Throughout the thesis the terms global patterns, emergent patterns, and generic patterns are used interchangeably.

The work presented in this thesis is exploratory and focuses on finding emergent patterns in a large collection of images. In this thesis, I am looking for answers to following questions: What information do emergent patterns possess? What do these patterns look like? What are different ways of extracting these patterns? I also want to find out various applications for which emergent patterns could be used. Emergent patterns are generic and therefore using co-occurrences that are in a local neighborhood can limit the types of extracted patterns. Instead both local (within a close neighborhood) and global (anywhere in the image) co-occurrences can be used for the mining process.

In this thesis, I first use FIM for finding emergent patterns from a large heterogeneous image collection using the FP-growth (Han *et al.*, 2000) algorithm. The mining process discovers significant semantic patterns which are broadly categorized into six classes. I also experiment with rare itemset mining technique which have been explored by very few researchers. The algorithm I used for this purpose is called RP-Tree (Tsang *et al.*, 2011).

In the second phase of this work, a new algorithm for finding emergent patterns using a graph theoretic approach is presented (Khan *et al.*, 2014). In this work, I present an approach to determine important co-occurrences from the entire data set. A graph is created by using the most important co-occurrences which results in emergent clusters. The results are first verified using a simple image data set. Experiments on a complex image dataset containing various object categories show that this process reveals interesting patterns including, but not limited to, object classes. In another set of experiments these important co-occurrences are used to represent images. To measure its efficacy tests are conducted in an image retrieval scenario. This approach is called *Bag-of-Co-occurring-Words* (BoCoW) and it encodes the co-occurrence information in a representation similar to the standard *Bag-of-Words* (BoW) model.

1.1 Motivation

Today the Internet provides a gigantic repository containing billions of images. This poses a challenge: How can we find useful information in such a large collection of images? The heterogeneity of the image contents and scale of data makes it hard for supervised methods to work well, and encourages the use of unsupervised methods for finding information. FIM algorithms are used to discover visual patterns in images. The majority of applications (Fernando *et al.*, 2012; Quack *et al.*, 2006, 2008; Yuan *et al.*, 2007a) that use FIM for visual pattern extraction find patterns that could relate to object or scene categories, and do not experiment with large scale image data sets. However, there is very little work done in finding generic patterns using FIM based approaches on large scale data sets. Rare itemset mining (RIM) is another area of interest because it has been largely ignored in the computer vision community.

The main motivation of this work is to find out:

What do the emergent patterns in images look like? How can we extract emergent patterns, and what are different scenarios in which emergent patterns can be used?

1.2 Challenges and Contributions

Very little work has been done in finding emergent patterns. Amongst the challenges that I faced were:

- Evaluating emergent patterns is hard as these pattern are generic and in most cases patterns cannot even be related to everyday objects. Usually patterns (e.g., a checked texture) appear at multiple positions and on different objects which makes evaluation even harder.

- Another problem is the unavailability of ground truth data sets of generic objects that could be used in the evaluation process.

As a building block of emergent patterns, I have chosen SIFT feature descriptor (Lowe, 2004) which is one of the most used descriptors. The bottom-up approaches that are used in this thesis are: frequent itemset mining, rare itemset mining, and graph based mining. To solve the problem of evaluating emergent patterns, I created a data set of ground truth images. The main contributions of this thesis are:

- I have applied the frequent itemset mining (Agrawal *et al.*, 1993) technique on a large scale image data set for finding emergent patterns (Khan *et al.*, 2012a). The patterns that emerged are dots and checks, parallel lines, bright lights, etc.

- I have applied a rare itemset mining technique on a large scale image collection for finding emergent patterns (Khan *et al.*, 2012a).

- I have presented a novel approach to find emergent clusters of visual words using a graph-based approach (Khan *et al.*, 2014). The emergent clusters are obtained by applying normalized cuts (Shi and Malik, 2000) algorithms on the graph.

- I have introduced a criterion that finds significant co-occurrences that contribute to finding emergent clusters. Rather than finding frequent or rare patterns, this criterion determines co-occurrences that appeared more than a random chance would allow.

- I have presented an approach that encodes significant co-occurrences into Bag-of-co-occurring-words (BoCoW). This technique is used to analyze the performance of significant co-occurrences in an image retrieval scenario.

- I have developed a ground truth data set which contains 6000 images from six objects. This data set is used to evaluate the performance of the graph-based approach for finding emergent clusters.

- I have presented a technique for compressing SIFT features (Lowe, 2004) to reduce the amount of memory needed by features. This is critical for an application that deals with a large number of images. The performance of this approach is compared to another approach presented in the literature (Khan *et al.*, 2012b).

1.3 Limit of Scope

The work presented here aims to find emergent patterns in large scale image collections without supervision. Emergent patterns are complex or generic patterns that are often hidden. Different factors can affect the types of emergent patterns we get. However, it is not possible to address all these challenges during the duration of a single Ph.D. Some of the limitations of the work are:

- The goal of this research is to find out different ways of discovering emergent patterns. In this Ph.D. I have explored the bottom-up method for pattern extraction. There are other approaches based on top-down strategies (Wang *et al.*, 2014) but are the not focus of this work.

- Emergent patterns are the combination of image features, and a particular feature captures image properties in a certain way. It could be interesting to see how these patterns change by changing the type of feature or using multiple features together, but this is not the focus of this work. This work is just restricted to SIFT descriptors.

- The purpose of association rules is to find relationships among items and then use them to determine a higher level semantics. In this thesis I use association rules for finding important itemsets (based on the strength of relationship) but association rules are not used for getting higher level semantics. This is because the complexity of the patterns I get makes analysis difficult and is therefore out of the scope of this work.

- The extracted emergent patterns are considered to have a flat structure. Instead they could be organized in a hierarchy to discover relationships among them, but this is out of the scope of this work.

- When illustrating FIM and RIM techniques patterns are chosen randomly for visualization. Instead, more intelligent methods could be tried which are set as future work.

1.4 Thesis Layout

This thesis describes the various approaches; experiment and results carried out to evaluate algorithms on finding emergent patterns in a large collection of images. The thesis consists of eight chapters which are detailed as follows:

Chapter 2 presents important concepts such as image features, clustering, and the Bag-of-Words (BoW) model in detail. These concepts are necessary to understand the rest of the thesis.

Chapter 3 reviews techniques for finding visual patterns in images. Visual pattern extraction using frequent itemset mining is then discussed in detail.

Chapter 4 presents an approach for finding emergent semantic patterns in large scale images. The underlying techniques of frequent itemset mining and rare itemset mining are discussed in detail, along with experiments and results.

Chapter 5 presents a novel graph-based approach for finding emergent patterns in image collections. The separation of emergent clusters using normalized cuts is also discussed in this chapter.

Chapter 6 explains how important co-occurrences can be used to represent image collections. The performance of this method is evaluated in an image retrieval scenario.

Chapter 7 presents an approach to reduce the amount of memory needed by the feature descriptors which is a problem when dealing with large scale image data sets. A SIFT feature compression approach is presented. The approach is compared with another approach from the literature.

Chapter 8 concludes the thesis and presents the final remarks and suggestion for possible future works.

1.5 Publications

Some of the techniques mentioned in this thesis have previously been described in several refereed publications, which are listed below.

- Khan, U.M.; McCane, B.; Trotman, A., "*Emergent Semantic Patterns in Large Scale Image Dataset: A Datamining Approach,*" Digital Image Computing Techniques and Applications (DICTA), 2012 International Conference on , vol., no., pp.1,8, 3-5 Dec. 2012
- Khan, U.M.; McCane, B.: Trotman, A.; "*A Feature Compression Scheme for Large Scale Image Retrieval Systems,*" Image and Vision Computing New Zealand

(IVCNZ), 2012 Proceedings of the 27th Conference on , vol., no., pp.492,496, 26-28 Nov. 2012

- Khan, U.M.; Mills, S.; McCane, B.; Trotman, A., "*Emergent Properties from Feature Co-occurrence in Image Collections,*" Pattern Recognition (ICPR), 2014 22nd International Conference on , vol., no., pp.2347,2352, 24-28 Aug. 2014

Chapter 2

Bag-of-Words (BoW) and Scale Invariant Feature Transform (SIFT)

This chapter introduces some basic concepts and techniques which are key to understand this work. These techniques are very common and are adapted to various kinds of computer vision applications ranging from scene and object categorization, image clustering and classification. In the first part, the Bag-of-Words (BoW) model including its constituent stages and its applications are discussed. In the second part, the SIFT feature descriptor is examined.

2.1 Introduction

As introduced in Chapter 3, visual patterns are combinations of visual primitives that co-occur multiple times in images. The visual primitives can be interest points, regions, corners, blobs, and, local features etc. The techniques to extract these are: Harris-Laplace (Mikolajczyk and Schmid, 2001), difference of Gaussian (DoG) (Lowe, 2004) or maximally stable extremal regions (MSERs) (Matas *et al.*, 2004), and scale invariant feature transform (SIFT) (Lowe, 2004), PCA-SIFT (Zickler and Efros, 2007), speeded up robust features (SURF) (Bay *et al.*, 2006) or histogram of oriented gradients (HoG) (Dalal and Triggs, 2005). SIFT is one of the most widely used descriptors that has proven to be the best in many comparative studies (Mikolajczyk *et al.*, 2005; Quelhas *et al.*, 2007a). SIFT feature descriptor and BoW method are adapted in many research works.

Sivic and Zisserman (2004) use SIFT and BoW for mining spatial configurations of viewpoint features for movie summarization. Quack *et al.* (2007) use both techniques to find frequent spatial configurations of visual primitives using an Apriori-based mining algorithm. The frequent configurations separate foreground and background objects in images. Chum and Matas (2010) use these techniques to discover sets of features that co-occur with high probability i.e., *co-ocsets*. Rather than using FIM, their approach is based on min-Hash for discovering dependencies in features. Geometric preserving visual phrases (GVP) presented by Zhang *et al.* (2011) uses SIFT and BoW for an image retrieval application, while Fernando *et al.* (2012) adapt these techniques for image classification problem using FIM technique.

2.2 The Bag-of-Words (BoW) Model

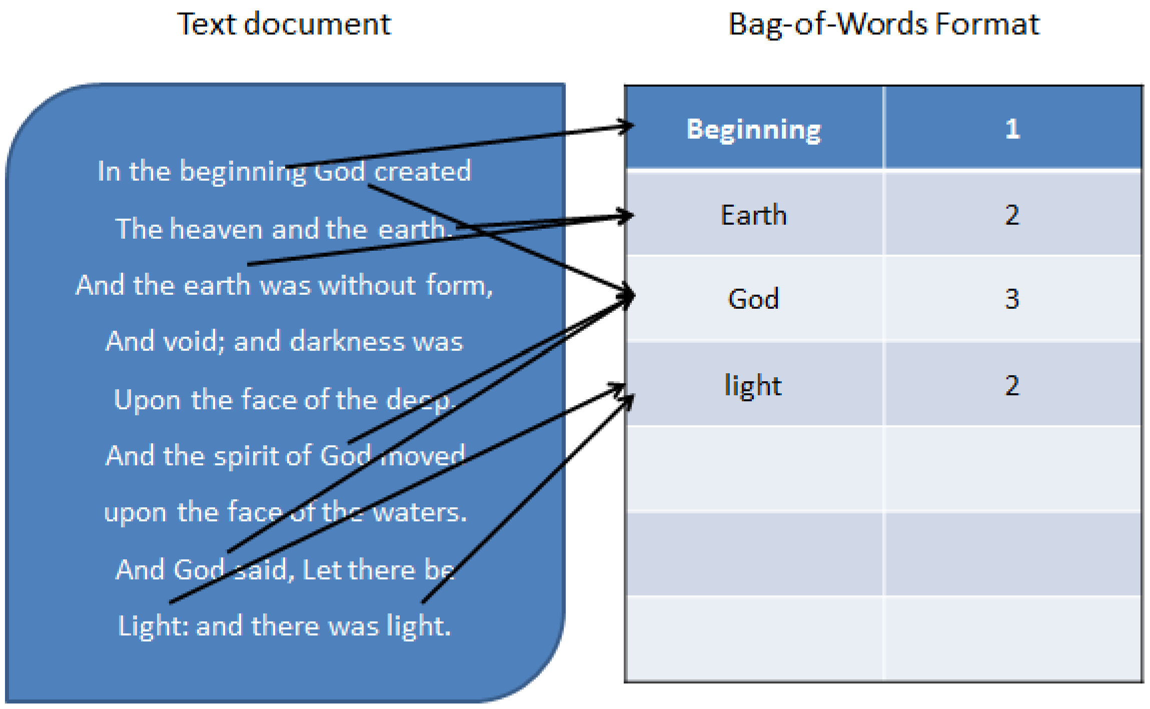
The term Bag-of-words (BoW) has its origin in the text document retrieval domain. A text document contains some distribution of words, and thus it can be summarized by the frequency count of these words (called a Bag-of-Words) as shown in Figure 2.1. This methodology provides some cues for applications that deal with searching or retrieving images. For example, an image is like a document that contains local feature descriptors, which we can think of as words. However, one obvious problem with this analogy is that text words are discrete "tokens" and local feature descriptors on the other hand, are typically high dimensional, real-valued feature points. So the next question is how to obtain a discrete representation or a "visual word". To solve this problem a method based on the process of vector quantization by clustering the local feature descriptors is suggested. A feature then can be coded according to the nearest discretized region of feature space it belongs.

To obtain these visual words (visual vocabulary) from images, a standard pipeline is adopted (Figure 2.2). This involves:*1*) local feature detection and description, *2*) quantization (clustering) of the feature space into a predefined number of clusters to form a visual vocabulary, and *3*) for each feature in the image finding the closest visual word from the vocabulary and representing in the BoW format i.e., a histogram of visual words frequencies.

Definition of BoW

The BoW model can be defined as follows (Tsai, 2012). Let D be the set of training image dataset containing n images, and I be the features space, $I = \{i_1, i_2, ..., i_n\}$, where i_k is the list of extracted features in an image. An unsupervised learning algorithm, such as k-means, is used to cluster I into a fixed number of visual words W. Where

Figure 2.1: Bag-of-Words (BoW) format for a text document.


Text document
Bag-of-Words Format
In the beginning God created
The heaven and the earth.
And the earth was without form,
And void; and darkness was
Upon the face of the deep.
And the spirit of God moved
upon the face of the waters.
And God said, Let there be
Light: and there was light.
Beginning 1
Earth 2
God 3
light 2

W is represented as $W = \{w_1, w_2, ..., w_k\}$, where k is the total number of clusters. The data can be represented in a $k \times n$ co-occurrence table of counts $N_{pq} = c(w_p, i_q)$,where $c(w_p, i_q)$ denotes how many times the word w_p occurred in an image i_q.

2.2.1 Local Feature Detection and Description

The first step in creating a BoW is to obtain information from image by extracting features. A feature is an interesting part or region of the image and provides an abstraction of image information. There are several image processing techniques that are used in the literature to detect these interest points (Mikolajczyk *et al.*, 2005; Tuytelaars and Mikolajczyk, 2008). Some properties of a good feature and its description method are as follows.

Rotation invariance: The extraction algorithm should be capable of detecting the same features regardless of the changes in the orientation of objects in images.

Scale invariance: The detected feature should be same regardless of changes in the scale (i.e., change in size) of the images.

Perspective invariance: The features descriptor remains invariant to viewpoint changes and retains same information. This kind of invariance is also called affine invariance.

Illumination Invariance: The feature descriptor should not change because of changes in lighting conditions or illumination changes.

Noise Invariance: The local feature should be invariant to the various kind of noises in image, e.g. motion blur, Gaussian noise, etc. should not affect the feature detection process.

Some of these region detectors that are interesting, or related to the our work are discussed here.

Interest Point Detection

1. Harris-Laplace regions: Harris-Laplace is a scale invariant corner detector Mikolajczyk and Schmid (2001). To detect Harris-Laplace regions a multi-scale variation of Harris corner detector (Harris and Stephens, 1988) is used. A region is selected in scale-space by the Laplacian-of-Gaussian operator.

2. DoG regions: Difference of Gaussian (DoG) regions (Lowe, 2004) are blob-like structures. These regions are detected at local maxima of the difference of Gaussian filter. The region detector is faster and more compact than other detectors (Tsai, 2012).

3. Hessian-Laplace regions: These regions (Mikolajczyk *et al.*, 2005) are detected in space at the local maxima of the determinant of Hessian (or Hessian (Binmore and Davies, 2001)) and at the scale of the local maxima of the Laplacian of Gaussian (LoG).

4. Salient regions: These regions (Kadir and Brady, 2001) are also detected in a scale space but at local maxima of the entropy of the pixel intensity histograms. Multiple circular regions of different sizes are extracted at each position in the image. Then the entropy of pixel intensity histograms is computed for each circular region.

5. Maximally stable extermal regions (MSERs): These regions (Matas *et al.*, 2004) are connected components of pixels obtained after thresholding the image. A watershed-like segmentation algorithm is run on image intensity values which

produces multiple segments in the image. To obtain the regions, only those segments whose boundaries remain stable over a wide range of its thresholds are considered. The position of the region is obtained by computing the average of x and y pixel locations. The get the size of the region, the geometric mean of the eigenvalues of the second order moment matrix is computed for each pixel location.

Local Feature Description

Usually, features provide regions or interest points in the image and descriptors are used to describe them efficiently. A variety of feature descriptors are presented in the literature, and some of them are SIFT (Lowe, 2004), PCA-SIFT (Zickler and Efros, 2007), SURF (Bay *et al.*, 2006), and HoG (Dalal and Triggs, 2005). SIFT is one of the most widely used descriptors. SIFT combines scale invariant region detector and a gradient distribution descriptor together. SIFT descriptor represents gradient locations and orientations using a 3D histogram into a 128-dimensional feature vector. A number of comparative studies (Mikolajczyk *et al.*, 2005; Mikolajczyk and Schmid, 2005; Quelhas *et al.*, 2007a; Zhang *et al.*, 2007) suggest that SIFT performs better than other descriptors. The SIFT descriptor is discussed in more detail in the next section of this chapter.

2.2.2 Quantization and Visual Words Generation

After the local features are extracted and described using a descriptor (e.g., SIFT), the next step is to quantize local descriptor vectors to compute visual words. The standard procedure to obtain a visual vocabulary involves: *1*) obtaining a large sample of local features from a collection of images representing a corpus; and *2*) quantizing the feature space using a clustering algorithm. Typically the k-means algorithm (Lloyd, 1982) is used for this purpose, where k is the total number of clusters to be made which is supplied by the user. The k-means algorithm aim to partition n observations into k clusters, and each observation is associated to a cluster with the nearest mean. This mean value serves as a prototype of a cluster and is called centroid or cluster center. These centroids become the visual word, so there are k words in the vocabulary. Finding the solution to k-means is an NP-hard problem so there are approximate methods that are usually adopted e.g., hierarchical k-means (Nister and Stewenius, 2006).

2.2.3 Image Representation using Vocabulary

The sample features that are used in quantization are usually discarded after obtaining the visual vocabulary. To represents a new image in the BoW format first all image features are obtained. Then for each feature its nearest visual word is determined. A distance function, which is usually based on Euclidean distance, is used to compute the similarity between a visual word and a feature descriptor. At the end of this process, we know how many times a particular word occurs in an image, and the feature(s) associated with each visual word. This information is represented as a histogram of visual words for each image, which is the BoW. The number of bins in this histogram is the number of visual words (k), and the frequency of each bin represents the number of features associated with the words in an image. Apart from just using raw frequency values to represent a BoW there are weighting schemes that are discussed in the literature. Normalized term frequency (NTF), term frequency inverse document frequency (TF-IDF) (Jiang *et al.*, 2007) are the most common weighting schemes.

2.3 Scale Invariant Feature Transform (SIFT)

SIFT features are claimed to be invariant to image scale, rotation, and performs well under 3D viewpoint change and illumination changes (Lowe, 2004). There are four major computational stages of SIFT the descriptor. To reduce the feature extraction time, a cascade filtering mechanism is adopted. This allows us to apply the most expensive operations only to locations in an image which pass some initial criteria. The SIFT has following main stages:

Scale-space extrema detection: In this stage extremas (i.e., maximas or minimas) are detected at all scales of an image that determines the candidate keypoints. This step provides scales invariance to the SIFT feature, and it can be efficiently implemented using Difference-of-Gaussian (DoG).

Keypoint localization: The initial keypoints detected in the previous stage are refined here, and only the stable keypoints are selected. The stability criterion is based on the contrast and the location of the keypoint (i.e., to determine whether it is located at the corner or an edge). A detailed model is fit at each keypoint to find the exact location and scale of the keypoint.

Orientation assignment: The stable keypoints are assigned one or more orientations

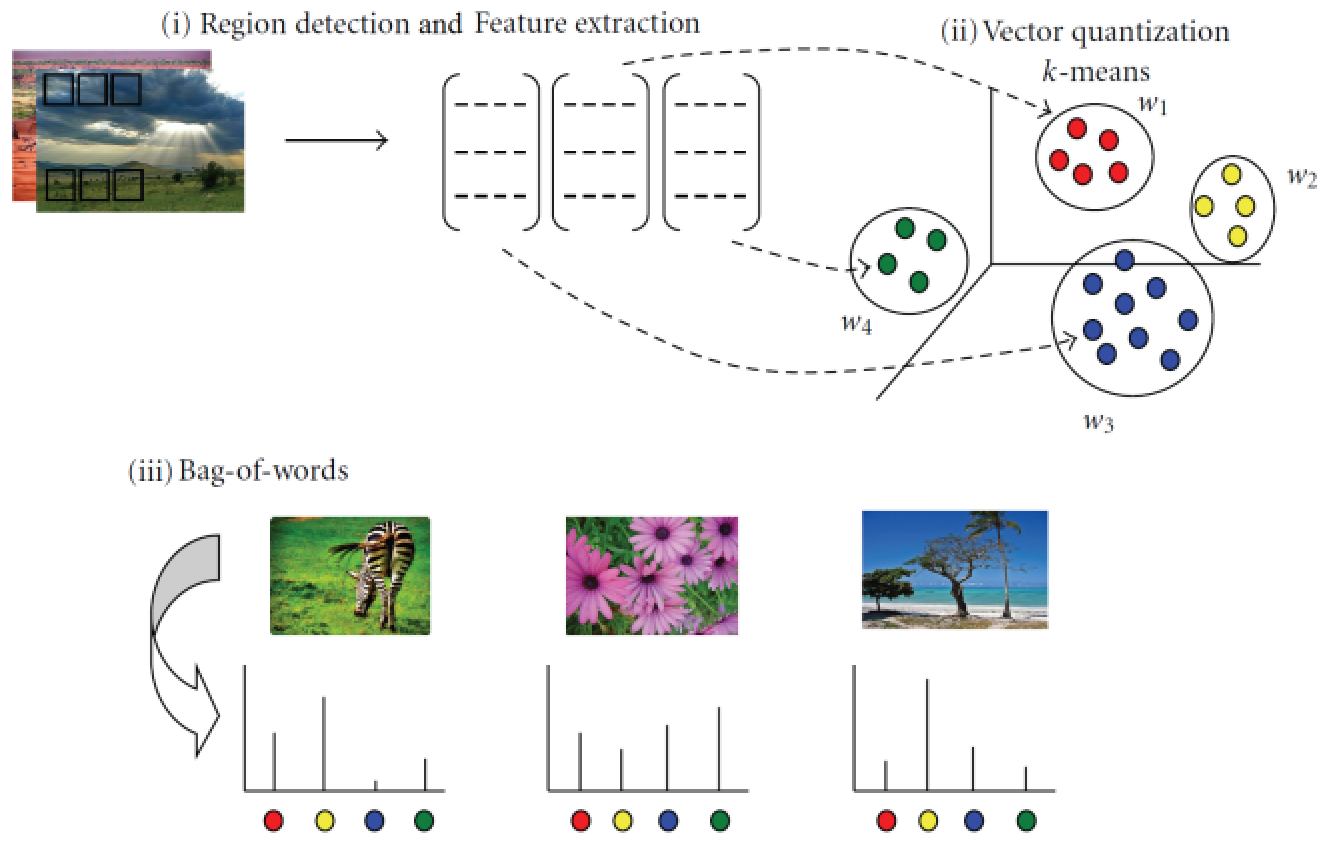


Figure 2.2: Steps for constructing visual bag-of-words (BoW) representation from images. (Adapted from Tsai (2012) with permission)

based on the local image gradients. This operation adds rotation invariance to the descriptor.

Keypoint descriptor: This stage represents the region around a keypoint in the form of a 128 dimensions vector. It is because of this stage that the SIFT descriptor gets its partial invariance from illumination and viewpoint change. The process starts by computing a 16×16 window (i.e., sixteen 4×4 regions) around a keypoint. For each region, the gradient magnitude and orientations are computed, and the orientations are represented in an 8-binned histogram. The gradient magnitude in the regions is weighted by a Gaussian weighting function to give more weight to regions closer to the keypoint. The final step is to represent the orientation histogram values obtained from the 4×4 regions in a vector. A normalization process is usually followed which converts these values to a unit length and helps achieve invariance to illumination.

2.4 Summary

This chapter described two basic but important concepts that are used in many computer vision applications, and which are very important to visual patterns discovery approaches. Multiple stages of the BoW model such as: keypoint detection and feature description, visual vocabulary creation, and representing an image in BoW are discussed. We also explained multiple stages of the SIFT algorithm and discussed how they contribute to making SIFT descriptor invariant to various transformations.

The next chapter describes a data mining technique for extracting emergent patterns. This method is built upon the concept of local features, BoW, and frequent itemset mining techniques.

Chapter 3

Background Theory

This chapter discusses various techniques for finding visual pattern in images. The literature is grouped into two main approaches i.e., bottom-up and top-down. The bottom-up pattern extraction methods are described in detail. These algorithms are further categorized into frequent and rare itemset mining techniques, with the FP-Growth algorithm and RP-Tree algorithms are discussed in detail. This chapter also covers some methodologies from the top-down methods.

3.1 Introduction

Advances in image acquisition devices and storage technology allows us to generate billions of digital images every day. Some of the sources of these images are digital cameras, smartphones, scientific equipment and medical imaging devices, etc. This huge repository contains useful information, but it is very hard to analyze. If labeled information is available for this data, then some form of supervision can be provided to the learning algorithms to extract useful information. Even in this case, variations within objects or scene categories can pose serious challenges. In most real scenarios, when very large collection of data are collected from the Internet, obtaining labeled information for training purposes becomes hard. As a result, there is a need for using unsupervised learning techniques to extract this information as they do not require any labeled information for training. Unsupervised learning approaches can be used to discover visual structures and patterns in an image collection. The goal of this thesis is to find out what these patterns are, and what information these patterns encode.

Visual patterns are set of visual primitives (e.g., pixels and features) that co-occur multiple times in images or video data (Wang *et al.*, 2014). These visual patterns

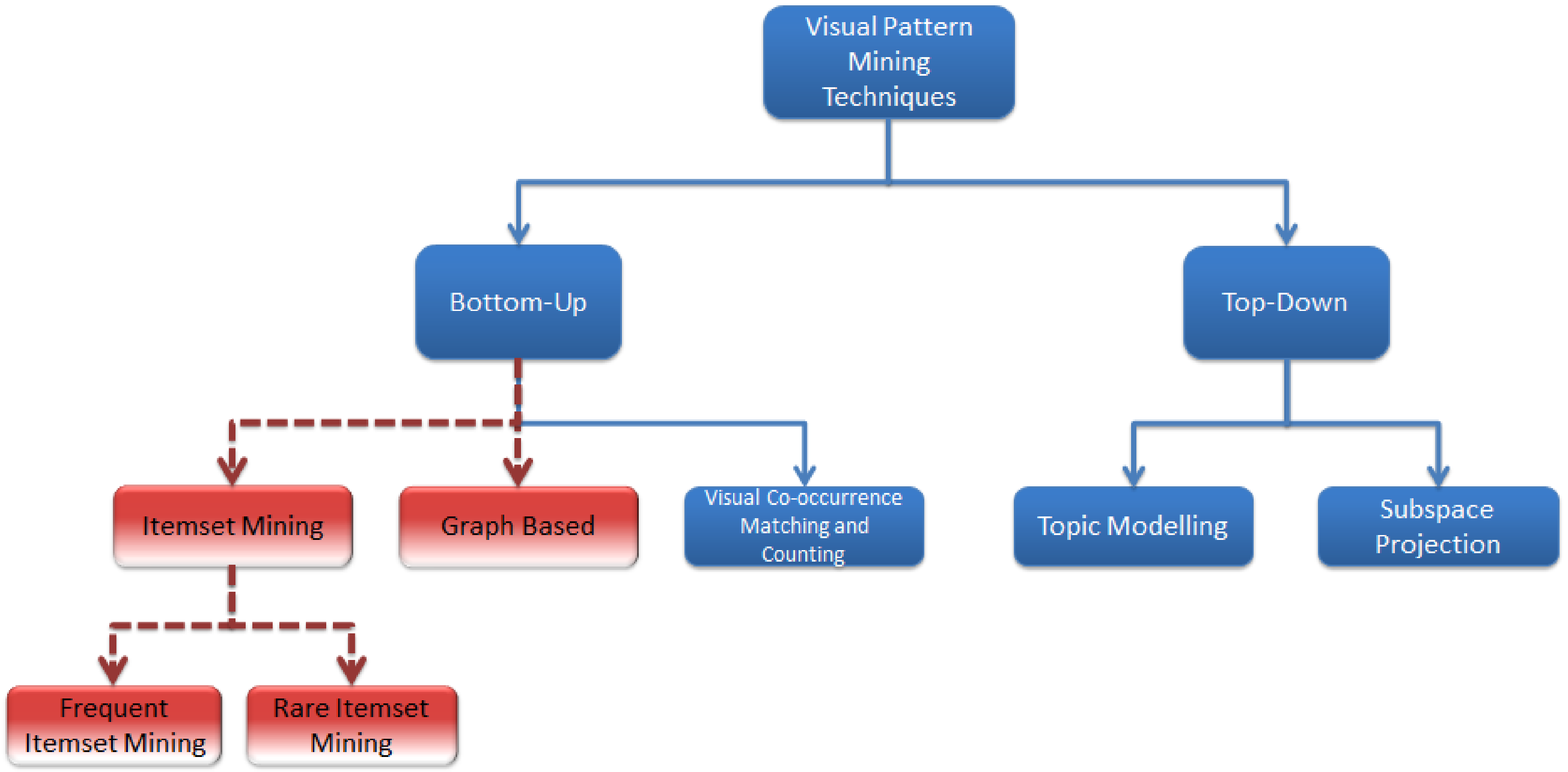


Figure 3.1: A summary of visual pattern discovery approaches (Wang *et al.*, 2014). The red segments describe the techniques explored in this thesis.

provides implicit knowledge, structural relationships within an image, or other patterns that are not explicitly stored in the data set (Bhatt and Kankanhalli, 2011). There are two main ways to find visual patterns that are described in the literature i.e., bottom-up and top-down (Wang *et al.*, 2014). Figure 3.1 depicts the different approaches in a hierarchical fashion. In this figure the highlighted approaches (connected through broken arrowed line) are the focus of this review, and hence they are discussed in detail.

The *bottom-up* mining process starts by extracting visual primitives which are then quantized to obtain visual words using techniques such as K-Means clustering. Images are usually represented in Bag-of-Words (BoW) format (Sivic and Zisserman, 2003). After that visual patterns or commonly co-occurring visual word configurations are discovered in the entire collection. Figure 3.2 shows the different layers of the bottom-up process. The bottom level depicts three kind of visual words (i.e., plus, star, and diamond shaped) obtained after the clustering process. The middle level shows each frequent configuration of visual words in a specific color. The top level shows some visual patterns discovered in the image collection. Frequent itemset mining (FIM) techniques (Agrawal *et al.*, 1993) are a common strategy under this category.

The *top-down* method Figure 3.3 on the other hand, first builds a model of images and visual patterns in it, and then visual patterns are inferred from this model. Topic modeling methods such as probabilistic Latent Semantic Analysis (pLSA) approach (Hofmann, 2001) and Latent Dirichlet Allocation (LDA) approach (Blei *et al.*, 2003) are common strategies in this category. LDA is a generative model which aims to find hidden topics in a text document. In this model each document is a distribution over topics and each topic is a distribution over a fixed vocabulary of words. In the case of images each image can be considered as a document and the topics are the visual patterns. Figure 3.3 shows a graphical representation (plate structure) of a LDA model made over documents.

Visual pattern mining has been used for various applications such as image retrieval (Quack *et al.*, 2008; Zhang *et al.*, 2011; Zheng *et al.*, 2009), scene and object categorization (Cao and Fei-Fei, 2007; Fernando *et al.*, 2012; Yuan *et al.*, 2007b), video analysis (Gilbert *et al.*, 2008; Quack *et al.*, 2006; Sivic and Zisserman, 2004) etc. Most of these applications use FIM to find patterns that are related to object or scene categories. They take advantage of spatial information about each visual primitive. In other words, only those co-occurrences, which exist within a local neighborhood of a visual primitive, are used and long-range global relationships are ignored. These global co-occurrences can result in finding generic and complex patterns that appear

Figure 3.2: Bottom up approach for finding visual patterns in images. The bottom level depicts three visual words (plus, star, and diamond shaped) obtained after the clustering process. The middle level shows each frequent configuration of visual words in specific color. The top level shows a few visual patterns discovered in the image collection. The figure is best viewed in color.

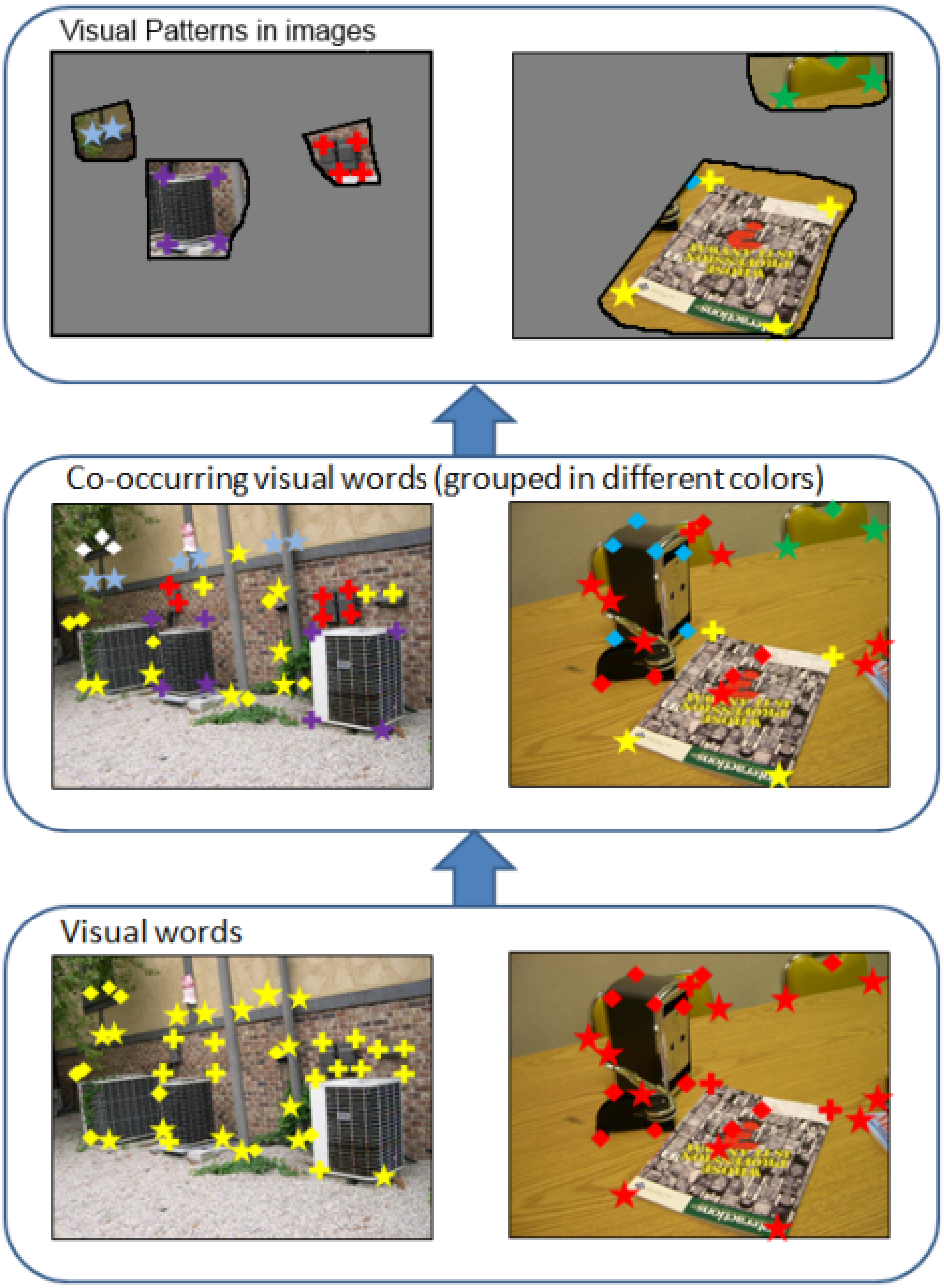
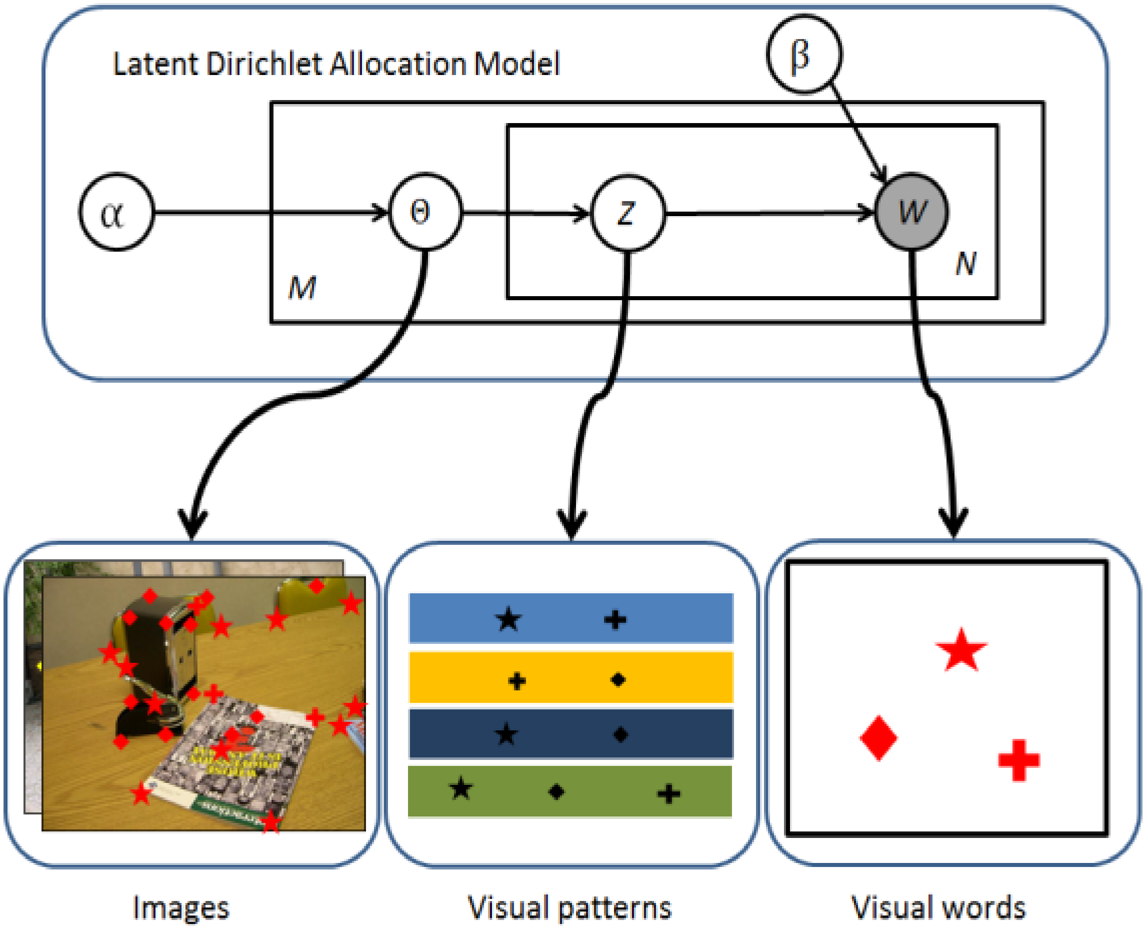

Visual Patterns in images
Co-occurring visual words (grouped in different colors)
Visual words

Figure 3.3: Top down approach for finding visual patterns in images. (Adopted from Wang *et al.* (2014))

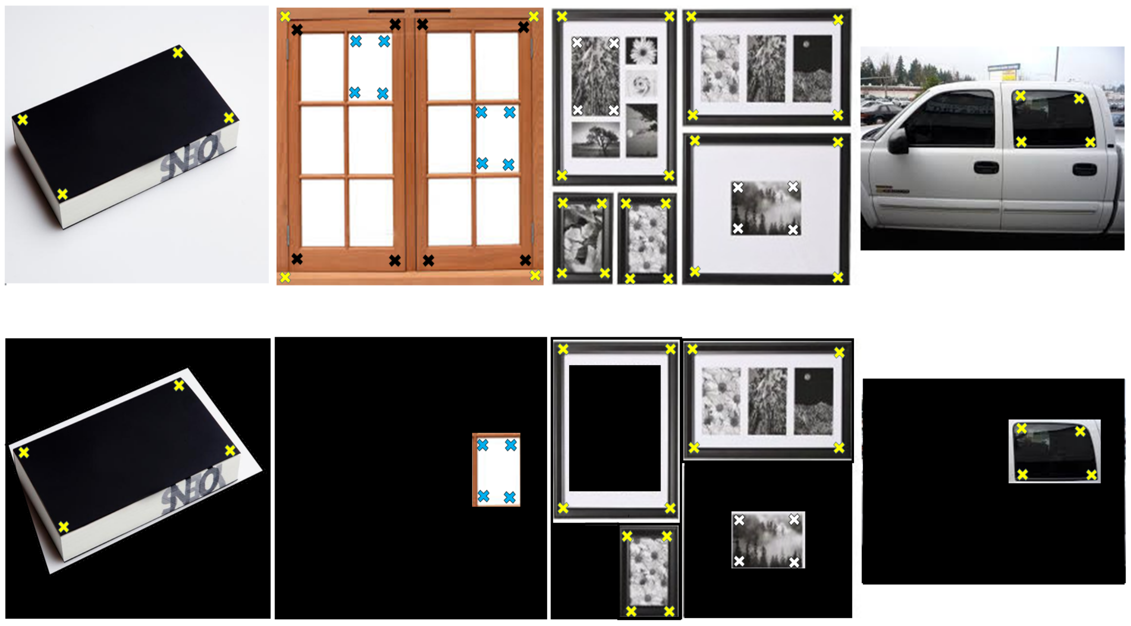


at random locations in images and do not relate to a particular object or a scene (Gao *et al.*, 2009). For example, think of a pattern that represents an object with four corners, e.g., a book, a window, or perhaps a frame of a picture, as shown in Figure 3.4. The pattern has multiple instances appeared at different locations and have multiple sizes. Any pattern discovery algorithm that only uses visual primitives within a local neighborhood might be unable to extract this pattern. So we consider it very important to use global co-occurrences and see what kind of information is obtained. These global co-occurrences not only contain long-range relationships, but also include local relationships. The patterns that utilize this information are called *emergent patterns.*

> *Emergent patterns are generic, complex and hidden structures in images that arise in an unsupervised way.*

To extract these patterns, we record co-occurrence information of every visual primitive with all other primitives in an image. As local co-occurrences are also been recorded, we expect to mine some emergent patterns representing objects or scene categories. The next section describes more detail about bottom-up and top-down techniques.

Figure 3.4: Example of a generic pattern representing four corners. The pattern appears on different objects and at multiple scales. The bottom row shows few pattern that are segmented out for better visualization.



3.2 Bottom-Up Approaches

Bottom up approaches can be divided into three main categories i.e., itemset mining, visual co-occurrence matching and counting, and graph based approaches. These approaches differ in way different method are used for extracting visual patterns. This section briefly discusses each category along with its applications.

3.2.1 Itemset Mining

A transactional dataset is composed of multiple records (transactions), where a record contains one or more elements, and is represented as a row of in the dataset. A market basket is an example of a transactional dataset where each element of a transaction

Table 3.1: A toy example of a transaction database. Each transaction contains a lits of all items in it. This layout of data is called horizontal layout it can be depicted as (TID:Itemsets).

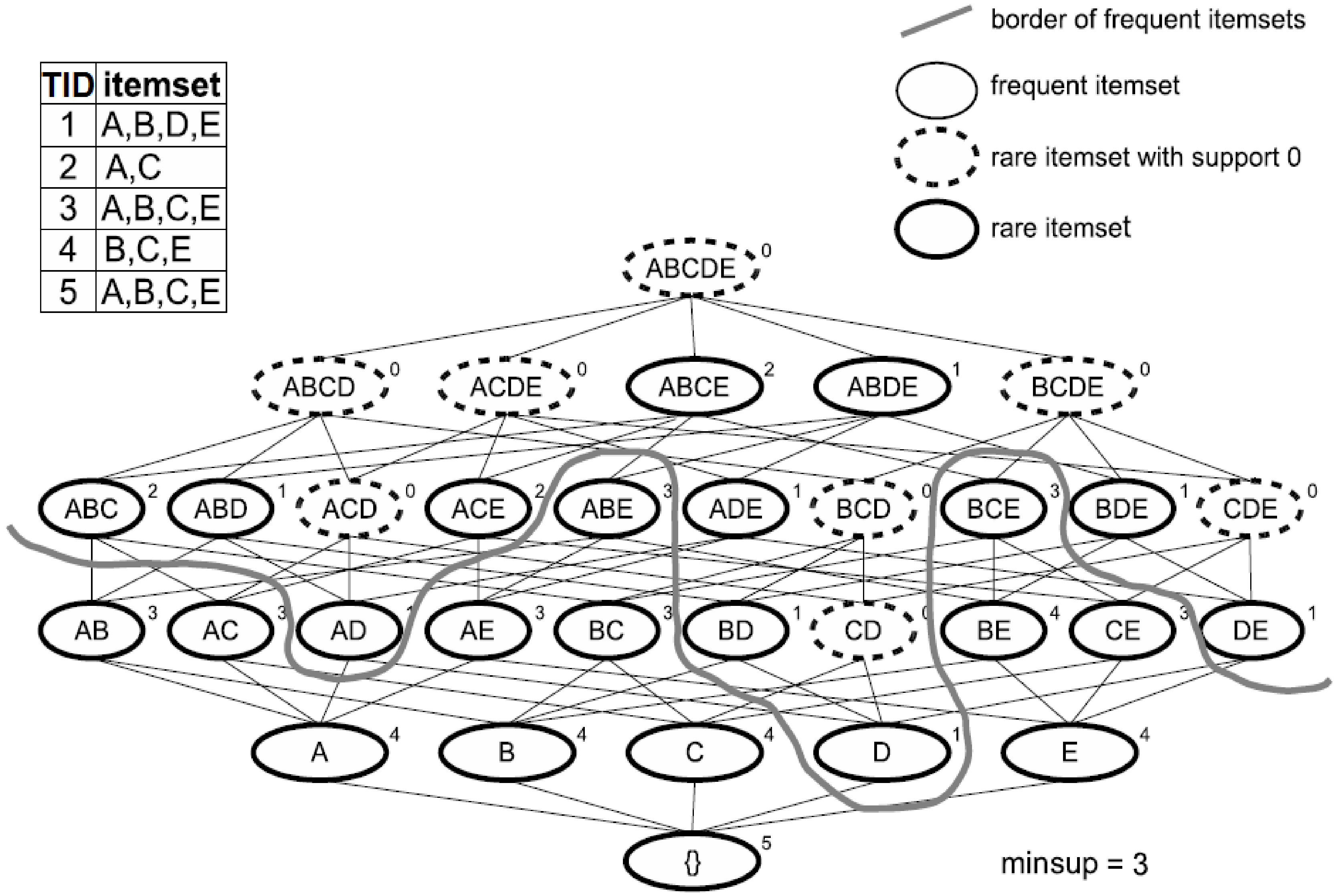
TID	Itemsets
1	f,a,c,d,g,i,m,p
2	a,b,c,f,l,m,o
3	b,f,h,j,o
4	b,c,k,s,p
5	a,f,c,e,l,p,m,n

is a grocery item purchased by a customer (Han and Kamber, 2006). Usually every transaction is comprised of a unique transaction identity (TID) code, and a set of items in the transaction called an *itemset.* This kind of data format is called horizontal data format (i.e., TID:Itemsets) as shown in Table 3.1. Usually the items in a transaction are represented as binary (1 or 0) flags, describing whether a transaction contains an item or not.

Let $I = \{i_1, i_2, i_3, ..., i_n\}$ be the set of all possible items in the dataset. A transaction, T contains a subset of items, that is $T \subseteq I$, and D is the collection of all transactions. An itemset is referred as k-itemset, if it contains k items in it. Itemset mining refers to a class of algorithms that discover interesting itemsets in a market basket dataset. The Frequent itemset mining technique introduced by Agrawal *et al.* (1993) finds itemsets that are frequent (common) in the entire dataset. Rare itemset mining on the other hand discovers itemsets that have very low frequency in the entire dataset. Images can be represented in the market basket metaphor by using a Bag-of-Words (BoW) representation (Sivic and Zisserman, 2003). Visual primitives from an image collection are quantized using a clustering algorithm to form visual words. Then each image is represented in a BoW format as a histogram of feature counts over the cluster centers (visual words). Itemset mining techniques can then be applied to a transactional dataset built on image collection.

Frequent Itemset Mining

As mentioned earlier, the concept of frequent itemset mining was first described by Agrawal *et al.* (1993). Following the definition of itemset mining described in the previous section, the interestingness criterion for a frequent itemset can be defined as: A k-itemset,



TID	itemset
1	A,B,D,E
2	A,C
3	A,B,C,E
4	B,C,E
5	A,B,C,E

Figure 3.5: The power set lattice of dataset D, shown on the table at the left. (Adopted from Troiano *et al.* (2009) with permission.)

is frequent if it occurs in a minimum (user-defined) number of transactions in the collection. This threshold is known as minimum support or *minsup*. FIM targets patterns that appear with high frequency in images.

The three very important algorithms (Han *et al.*, 2007) which are briefly discussed in this section are *Apriori*, *FP-Growth*, and the *Eclat* algorithm. It is important to note that these algorithms mainly differ in terms of efficiently finding frequent itemsets but the patterns extracted remain similar. This section briefly describes these approaches.

- **The Apriori** algorithm is based on the *downward closure* property (Agrawal *et al.*, 1994a), which means that a *k-itemset is frequent, iff all of its sub-itemsets are also frequent.* Apriori uses a level-wise approach for generating frequent itemsets and usually a power set lattice is built on the transactional data as shown in Figure 3.5. This permits itemsets at a higher level to be built on the itemsets that exists one level lower. For example, k-itemsets are used to generate $(k + 1)$-itemsets, which are then pruned using the downward closure property. Apriori terminates when there are no new $(k + 1)$-itemsets remaining after pruning. Hence, the algorithm can be divided into candidate generation and pruning stages. Some of the disadvantages of Apriori are the generation of a huge number of candidate itemsets and a high number of iterations over the transactional data set when checking these candidates.

- **The FP-Growth** algorithm solves some of the problems faced by Apriori and is presented in (Han *et al.*, 2000). It does not require the most time-consuming phase of candidate generation and hence is faster than Apriori (Goethals and Zaki, 2003). The whole mining process takes two iterations over the data set. In the first pass, all the frequent items are obtained and sorted in descending order according to their appearance frequency. These ordered items are used to build a frequent pattern tree (FP-Tree). Figure 3.6 depicts a toy example of a FP-Tree built on the data in Table 3.1. The nodes of this tree are the items and the counter at each node signifies the number of transactions containing that item. The header table stores pointers to the first instances of each item. The top to bottom order of the nodes is from the most to least frequent item. Using this order allows many overlapping paths that result in higher compression. The dotted arrowed lines (maintained as a singly linked list) allows us to locate the same item across the tree. The solid lines (read top to bottom) describe the order in which items appeared in the transaction. More details about this algorithm

Table 3.2: A toy example of a vertical data format. For every item, a TID-set is created. The items are ordered in descending order according to appearance frequency in transactions.

Item	TID-set
f	1,2,3,5
c	1,2,4,5
a	1,2,5
b	2,3,4
m	1,2,5
p	1,4,5

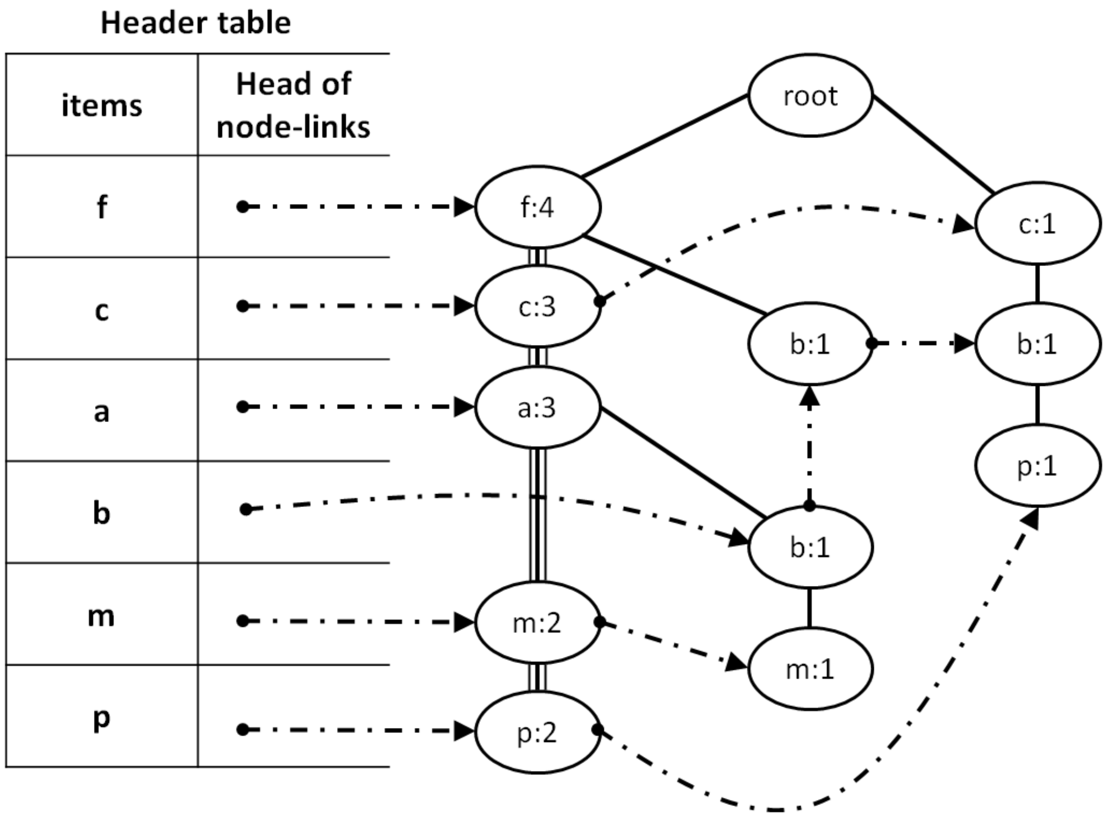
Table 3.3: All the itemsets obtained from the transactional data in Table 3.1.

Serial #	Frequent Itemsets
1	f,c,a,m,p
2	f,c,a,b,m
3	b,f
4	c,b,p
5	f,c,a,m,p

can be found in Chapter 4.

- **Equivalence Class Transformation (Eclat)** algorithm for mining frequent itemsets is proposed by Zaki (2000). Eclat performs mining on a vertical data format. This format is represented as (item:TID-set) i.e., for each item there is a set of the transactions that contain this item. The vertical data format is shown in Table 3.2 and is similar to inverted file index. In the first scan, the TID-set of each single item is created. The (k+1)-itemset can be obtained by taking an intersection of the two TID-sets of k-itemsets. This will give a TID-set for the (k+1)-itemset. The process is repeated until all the frequent itemsets are discovered. One obvious advantage of this algorithm is that once the vertical data format is built there is no need to scan the actual data again as it already contains all the information required for the mining process.

Figure 3.6: FP-Tree constructed from the items in Table 3.1. The highlighted (multi-lined) path means that the itemset (*f-c-a-m-p*) occurred twice in the dataset. Table 3.3 displays all the itemsets obtained using a $minsup = 2$.



Rare Itemset Mining

Rare itemset mining (RIM) finds patterns that appear in a very small number of images. A k-itemset is rare if its occurrence is unusual in the entire collection. To find out whether an itemset is rare or not the RIM mining process uses two threshold values, i.e., minimum-rare-support (*minRareSup*), and minimum-frequent-support (*minFreqSup*). The *minRareSup* act as a noise filter and all items that have the frequency below than this threshold are rejected. An itemset is rare if it has support higher than the *minRareSup* and less than the *minFreqSup*.

The majority of the rare itemset mining techniques are inspired by two algorithms. The first class of approaches use Apriori level-wise approach and suffer from same problems as Apriori, i.e., computationally expensive candidate generation and pruning steps. Some of these are *Rarity*, *AfRIM*, *ARIMA*, and *Apriori-Inverse* algorithms (Tsang *et al.*, 2011). The second class of approaches are those inspired by the FP-Growth algorithm and do not have a candidate generation phase. One example is RP-Tree (Tsang *et al.*, 2011) algorithm.

- **Apriori Based Techniques** As discussed before these techniques use level-wise exploration of the search space as in Apriori. In Apriori, the mining process uses a level-wise approach, that is the itemsets in the higher level are built on the itemsets that are in one lower level. For example, k-itemsets are used to generate $(k + 1)$-itemsets. The Apriori algorithm can be used for mining rare-itemsets by setting a very low minimum support threshold. This process may cause a combinatorial explosion of itemsets as there are a huge number of itemsets that meet the minimum threshold criterion.

  Troiano *et al.* (2009) presented a fast algorithm for mining rare itemsets called *Rarity*. They discovered that rare itemsets are at the top of the search space (see Figure 3.5), which means the algorithm first passes through all the lower layers generating frequent itemsets. They avoid all the passes through lower layers by first identifying the longest transaction in the data set, and then using it for mining rare items by performing a downward search. Adda *et al.* (2007) use very similar idea of using top-down search method in their presented algorithm called AfRIM.

  In another work Szathmary *et al.* (2007) presented A Rare Itemset Miner Algorithm (ARIMA) for mining rare itemsets. They split the mining task into two stages. The first stage identifies *minimal rare itemsets* which act as minimal generation seeds for the entire rare itemset family. The second stage then processes minimum rare itemsets to obtain rare itemsets. They presented two algorithms for the first stage and one algorithm for the second stage. The first algorithm in stage one is a naïve algorithm that is based on Apriori style enumeration. The second algorithm on the other hand is an optimized algorithm called minimal rare generators (MRG). ARIMA is used in the second stage for mining rare items.

  Finally, Apriori-inverse is proposed by Koh and Rountree (2005). It mines perfectly rare itemsets (all items have support less than some minimum frequent support). Apriori-inverse inverts the downward closure property, so the support of rare itemsets must be below a maximum support and higher than an absolute minimum threshold. Because of these thresholds, typically there are very few perfectly rare-itemsets. They further suggest some modifications to find imperfectly rare-itemsets.

- **FP-Growth Based Techniques** Tsang *et al.* (2011) presented an algorithm called *RP-Tree* which is built using an FP-Tree. Similar to FP-Growth, this approach in its first scan computes the support of items. However, for the second scan it only uses transactions that have at least one rare item (threshold below maximum frequent threshold and above a minimum rare threshold) in them. This approach is discussed in detail in Chapter 4.

Applications

One of the earliest works in image mining is presented by Ordonez and Omiecinski (1999) in which they find associations between blobs within the context of images. The mining is performed on blobs obtained from synthetic images of different geometrical shapes and rules are obtained between these blobs regardless of shape, orientation, and positions. During the same year, Megalooikonomou *et al.* (1999) presented an approach to discover association rules for relating different structures of the brain to functions. Although these initial approaches were far from mature and indeed required much work towards perfection, these works opened a new research direction for the next decade. Antonie *et al.* (2001) used the Apriori algorithm and association rule mining for detecting breast images that are normal, or abnormal. The abnormal images contain both benign and malign cases. The breast portion in the images is cropped in a preprocessing step. The antecedent of a rule are features (set of items) while the consequent of the rule is the category of the image. They achieve an average success rate of 69.11% using this approach. The work presented by Rushing *et al.* (2001) aims to detect texture patterns in images using association rules. The method they adopted converts a window of neighboring pixels to a transaction on which mining is performed. The approach is capable of detecting both natural and man-made textures and is also used for texture based segmentation. Some of the described approaches use global co-occurrences of image features that are very similar to what we are doing, but these works are very application oriented and do not mine generic patterns. The work of Rushing *et al.* (2001) is for finding generic patterns, but the method, they used for defining co-occurrence, is local and based on neighborhood approach.

Quack *et al.* (2006) used this method for video analysis and discovered, frequently co-occurring scenes and objects in a video. An image contains multiple transactions that are the spatial neighborhood of visual words obtained using a motion segmentation method. Apriori based frequent itemset mining is also been used for detecting features that occur on instances of a given object class in images (Quack *et al.*, 2007). The transactions represent spatial configurations of local features which are discovered automatically, and that frequently occur on the object of a given category. The mined data obtained from these transactions is used to identify features that occur on unseen objects from one of the categories with high probability.

Yuan *et al.* (2007b) used a modification of FP-Growth for finding semantically meaningful visual patterns in images. Instead of using itemset frequency as the interestingness criterion a likelihood-based criterion is proposed. The discovered patterns are used to refine the visual pattern related to objects.

Chum and Matas (2010) presented a method to find dependencies in sparse high dimensional data. They discover co-occurring sets (co-ocsets) using the minimum hash (Broder, 1997) algorithm. Co-ocsets are sets of features that have high probability of co-occurring together. They show that the general assumption about the independence of visual words is often violated and the co-ocsets are fairly common, which can degrade the performance of an image retrieval system if a standard term frequency inverse document frequency (TF-IDF) (Baeza-Yates and Ribeiro-Neto, 1999) based weighting scheme is used. They show that retrieval performance can be improved if these co-occurring words are efficiently modeled. Some of the co-ocsets extracted are: bricks, railing, text, faces, and water etc., (Figure 3.7, 3.8).

Figure 3.7: The different co-ocsets ('light text' in blue, 'bricks' in red, and 'railing' in green) detected in the method presented by Chum and Matas (2010). (Image adopted from Chum and Matas (2010) with permission).

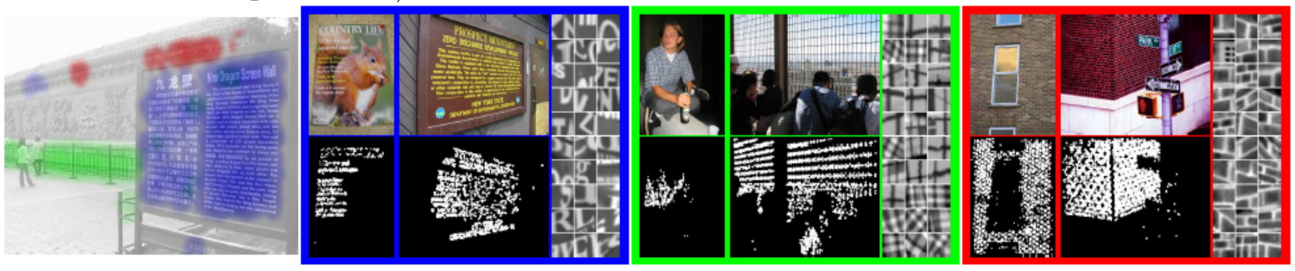
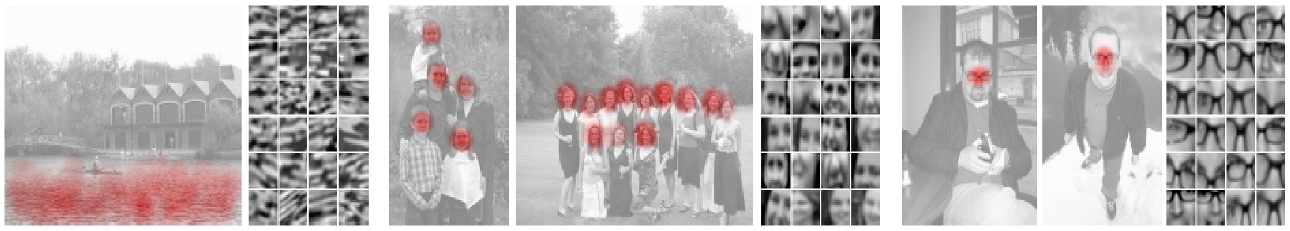


Figure 3.8: Examples of different co-ocsets with a sample of patches associated with core features. The color show the spatial distribution of co-ocset features. (Image adopted from Chum and Matas (2010) with permission).



3.2.2 Graph Mining

Graphs provide an efficient way of encoding relationships among data elements. In images, graphs are usually used to encode co-occurrence relationships among visual primitives. The visual primitives form the vertices of the graph, and the relationships among them define the edges among the vertices. The goal of graph mining techniques is to discover structures (sub-graphs, dense clusters) using graph theoretic approaches. Graph mining approaches are divided into two categories: transaction graph mining and single graph mining (Jiang and Coenen, 2009). In the transaction graph mining approach the data is in the form of many small graphs or transactions. The goal of the mining task is to find recurring or frequent sub-graphs. In the second type, the data is represented in the form of one big graph on which mining is done.

Applications

One of the earliest approaches to discover all frequent structures in a graph is presented by Inokuchi *et al.* (2000), which is called Apriori-based graph mining (AGM). Later Kuramochi and Karypis (2001) presented a method called frequent subgraph discovery (FSD) which is an extension of AGM and in which they presented the idea of using adjacent representation of the graph and an edge growing strategy. Both of these approaches are inspired by Apriori level-wise strategy (Agrawal *et al.*, 1994b). Using Apriori based approaches for frequent subgraph discovery faces the following challenges. Firstly, the candidate generation procedure, which generates candidate $(k + 1)$-subgraphs from k-subgraphs, is much more complicated and costly than in the case of itemset generation. Secondly, as the subgraph isomorphism test is an NP-complete problem so pruning false positives is very costly (Yan and Han, 2002). These problems are addressed by Yan and Han (2002) in their approach called *gSpan* (Graph-based substructure pattern mining). Their approach does not require any candidate generation. They arrange graphs in a lexicographic order and then use a depth-first search-based mining algorithm to mine the subgraphs. Their approach outperforms FSG by an order of magnitude.

Subgraph mining has also been used for finding common visual patterns between images for finding correspondences. In an approach presented by Leordeanu and Hebert (2005) common patterns are formed between two images by applying subgraph mining techniques on a feature correspondence graph. They formulate the problem of finding visual correspondence between images as a graph matching problem by defining an objective that includes terms based on both appearance similarity and geometric compatibility between pairs of correspondences. Zhao and Yuan (2011) use the problem of graph mining for finding thematic patterns in a video. Thematic patterns are sets of visual words that are spatiotemporally collocated. They formulate this problem as a cohesive sub-graph selection problem. They also performed accurate localization of the occurrences of all thematic patterns.

Graph pattern mining has also been used for discovering objects using shape features. Lee and Grauman (2009) presented an approach to model the shape of common objects in an unsupervised way. They extract edge fragments and represent them using local features that are used for matching. Spectral graph clustering is applied for common shape discovery. To separate the foreground edges from clutter within-cluster match patterns are computed. In another work Payet and Todorovic (2010) focus on the problem of finding objects categories in images by mining repetitive spatial configurations of contours across images. For this purpose, a graph is built on all pairs of matching geometric contours. All contour pairs, which deform similarly from one image to another are considered as collaborating (straight graph edges), or conflicting (zigzag graph edges). These edges help to cluster the graph into shapes that represent objects.

Gao *et al.* (2009) consider extracting structural semantics (often appear as repeated patterns) as a key in understanding both natural and man-made objects. They define semantics of a pattern as a specific set of relationships that connect visual words carrying special information globally i.e., regardless of their spatial proximity. The target is to find pair-wise associations of visual words having consistent geometric relationships sufficiently often. The problem is formulated as a minimal cost bipartite graph matching, where the cost depends on the spatial consistency of the candidate pairings. They further present a multiple associations (multi-model) approach to connect consistent associations.

3.2.3 Visual Co-occurrence Matching and Counting

In itemset mining and graph based approaches a transactional dataset is used to extract co-occurrence information. In these approaches the discovery of visual patterns depends the quality of transactional data, and visual words. This problems can be solved by visual co-occurrence matching and counting approaches because they do not need to build a transactional data for mining (Wang *et al.*, 2014). The main idea behind visual co-occurrence matching approaches is to identify high order feature co-occurrences in one image and then find it in other image. One solution is to use an offset space i.e., the relative location difference of visual primitives between two images. Calculating the offset space allows co-occurring visual primitives to assemble near the same places which facilitate visual pattern discovery.

Applications

In an approach (Zhang and Chen, 2009) they identify the higher order spatial features by evaluating the inner product of features from two images. Their approach can serve as a kernel for any kernel-based learning algorithm. They show that the performance increases in object categorization task when high-order features are used. In (Zhang *et al.*, 2011) they presented a technique called geometric preserving visual phrases (GVP). It encodes the neighborhood of a word as a visual phrase (set of neighboring words). Performance is measured on two data sets, and results are compared with bag-of-words followed by RANSAC (Fischler and Bolles, 1981) based verification steps. Their approach outperformed RANSAC based method and needed less memory and computation time.

Apart from finding discriminating sets of visual primitives, this approaches can also be used in many other ways. In (Yuan and Wu, 2007) they presented a method to find common visual patterns in images. They randomly partitioned each image several times, and a pool of sub-images are obtained. For each sub-image, a set of matched images is obtained. Similar patches are aggregated to obtain common visual patterns in images. Their approach directly operates on features, and there is no need to calculate visual words. In another interesting approach (Yuan *et al.*, 2007a) they focus the problem of finding recurring patterns in a single image. They first find optimal visual word matches. Then a greedy randomized adaptive search procedure is used to find common object patterns. A joint optimization procedure is adopted to find recurring patterns automatically.

3.2.4 Summary of Bottom-up Strategies

The bottom-up approaches typically start from visual primitives or visual words and repeatedly merge them until no other visual patterns are remaining. The bottom-up approaches has several advantages:

- As these methods are data-driven, they are easily adaptable in various domain areas.
- These methods allow contextual information to be used in the mining process, e.g., spatial co-occurrence and geometric relationships between visual primitives.
- They are often easier to implement.

Typically bottom-up methods incorporate spatial cues into the mining process and usually ignore global co-occurrences of visual words (Wang *et al.*, 2014).

3.3 Top-Down Approaches

The previous section discussed various strategies for the bottom-up approach for visual pattern extraction. A bottom-up process starts from low-level visual primitives which are combined to make higher level visual patterns. A top-down process, on the other hand, starts from building a model of visual patterns and then pattern discovery results are inferred from this model. These methods are based on unsupervised topic discovery methods usually adopted in natural language processing domain and strategies for subspace projection.

3.3.1 Topic Model Based Approaches

Topic modeling provides a way to find the main themes or patterns which are spread across large numbers of unstructured collection of documents. Topic modeling algorithms are capable of running on various kind of data e.g., text documents, images, etc. The literature for topic modeling approaches can be divided into two strategies: the classical approaches which do not use spatial and temporal information, and advanced approaches which benefit from the spatial and temporal information. The classical approach is closer to what we are trying to do, because when finding visual patterns only the global relationships among visual words are used.

In an early approaches by Sivic *et al.* (2005), topic modeling is used for determining the object categorization in unlabeled images. They use a topic model called probabilistic latent semantic analysis (pLSA). The model is built using BoW representations of images and they later add doublets (pairs of co-occurring words) to include spatial information. Experiments are conducted by setting multiple values of the number of topics to be discovered, which depend on the number of object categories in the collection. In the task of topic discovery, their method successfully learns topics related to object categories. Later, they test their approach for object and background classification task, where the classifiers are trained on the discovered topics. They use a separate training dataset that contains examples from object categories and background images. The new images can be successfully classified into object or background categories. In another work Russell *et al.* (2006) use topic modeling to find objects categories and also to segment the object area from the image. In their approach, they first make multiple image segmentations using normalized cuts, in a hope that some of them contain object specific information. Then they learn topics on theses segmentations using LDA. The segmentations are sorted based on the similarity of visual words in them. They show that the discovered topics are closer to object categories. Sivic *et al.* (2008) present an approach to obtain a hierarchy of objects using a hierarchical latent Dirichlet allocation (hLDA). They show that object classification and localization performance improved when compared with the state of the art (Russell *et al.*, 2006) which uses non-hierarchical LDA.

Spatial latent Dirichlet allocation (sLDA) is used by Wang and Grimson (2008) to cluster neighboring words and words that co-occur together into the same topic. The word-document information is not known a priori and rather becomes a random hidden variable. sLDA has a generative process that partitions visual words which appear close in the same document. They show that the sLDA achieves better performance than LDA. This approach aims to find the type and location of object in an image. In another approach, Philbin *et al.* (2011) targets the slightly different problem of finding building facades in a large collection of unordered images. Geometric latent Dirichlet allocation (gLDA) includes an affine homographic geometric relation in the generative process. The method has proved better performance from LDA in the task of finding particular objects and building facades.

These approaches use image datasets for building the model. Topic modeling is extended to video by Liu and Chen (2007) by including a temporal model. In their approach they perform appearance modeling using topic models, and motion modeling using probabilistic data association (PDA) filter. Both models are tightly integrated which helps to remove uncertainty appearing in one by using information from the other. They have shown promising results in video object discovery, which they claim are not achievable if just one of the models is used.

3.3.2 Subspace Projection Approaches

The previous section described a statistical viewpoint which uses pLSA and LDA based models to mine visual patterns in multimedia data. This section discusses another approach called subspace space projection, which is used to approximate the semantic structure of visual patterns. Non-negative matrix factorization (NMF) is one such method used in the literature. NMF is a group of algorithms in linear algebra and multivariate analysis. Given a matrix V, the method provides a way to find two vectors W and H such that their product approximates V. All these matrices have no negative values that makes it easy to inspect the resulting matrices.

Let matrix V be the product of W and H.

$$V \approx WH \tag{3.1}$$

The dimensions of the factor matrices may be very low as compared to V. If V is a $m \times n$ matrix, W is $m \times p$, and V is $p \times n$. Here, p is a set of features which can be very small from both m and n. We can think of each element in the matrix V being built from some hidden features and NMF discover these features.

Tang and Lewis (2008) present an approach that uses NMF for object class discovery and image auto-annotation. For the task of object class detection they use the parts-based representation characteristics of NMF. And for the second task of image auto-annotation they use NMF as an alternative sub-space technique i.e., SVD. They also mentioned the problem of finding the optimal number of dimensions for the subspace using NMF. They produced competitive results to LDA based method by Russell *et al.* (2006). Sun and Van Hamme (2011) used regularized NMF to model recurring visual patterns in images for the object classification task. The results show that their method always perform better than unsupervised NMF methods for smaller size of codebooks, i.e., under 5000.

3.3.3 Summary of Top-down Strategies

Top-down strategies for visual pattern discovery starts by building a model from images or documents. Where documents are considered as a mixture of topics, each topic is considered as a probabilistic distribution over words. The main goal of these approaches are to model and then infer the composition of visual primitives. There are several advantages of these approaches.

- While modeling visual data, any variations can be dealt with by using probabilistic reasoning.
- Multiple patterns can be discovered simultaneously because the generative process is designed for multiple patterns.
- These methods can also include geometrical and spatial information in the modeling process.

One major difficulty in using some of these approaches is that the process of model parameter learning and inferring the posterior probability is very challenging.

3.4 Discussion

This chapter discusses various approaches to discover visual patterns in a set of images. Emergent patterns are a special kind of visual pattern that represents generic, complex and hidden structures in images. The majority of approaches discussed here do not extract emergent patterns, as the extracted patterns are not generic and usually represent information related to specific object and scene categories. Only a handful of approaches exists to extract emergent patterns.

The method presented by Chum and Matas (2010) is particularly interesting in this regard, since they extract co-occurring sets (co-ocsets) from a large collection of images. Co-ocsets are sets of features that have high probability of co-occurring together. They extracted these patterns because they wanted to show that the general assumption about the independence of visual words is often violated, and this can degrade the performance of a retrieval system. Some of the extracted patterns can be seen in Figures 3.7 and 3.8. The approach they use is min-Hash algorithm which is based on Locality Sensitive Hashing method for sets (Indyk and Motwani, 1998).

The approach presented by Gao *et al.* (2009) is also relevant. Their method finds consistent associations of visual words using a minimal cost bipartite graph matching technique. The patterns are discovered regardless of their spatial proximity. This is one of the properties that we require for emergent patterns. Some of the consistent associations that they discover appear on windows, face elements (e.g., connecting eyes, lips, nose and hairline etc.), and grocery images.

It is also important to differentiate emergent patterns from *emerging patterns* (Dong and Li, 1999). Emerging patterns are used to find out the differences between two data sets. These are the itemsets whose *support* increases significantly from one data set to another. The itemset, which has a growth rate (the ratio of the two *support* values) larger than a threshold, is an emerging pattern. For example in time stamped dataset they can capture emerging trends. On the other hand, it reveals useful contrast between classes when applied to data sets with multiple classes (poisonous *vs* edible, male *vs* female or cured *vs* not cured) (Dong and Li, 1999).

3.5 Summary

Visual pattern mining is an important topic because of its applications in different areas of computer vision and data mining community. This chapter reviews various approaches for discovering these patterns from images without using any supervision. There are two main strategies: bottom-up and top-down approaches. The bottom-up approaches focus on mining compositions of visual primitives co-occurrences that are found in many images. Different algorithms for generating visual patterns in a bottom-up way are also discussed. Top-down methods, on the other hand, start by building a model from images, which is then used to infer the patterns in a document. We also introduce emergent patterns as, patterns which represent generic objects and that do not only use local co-occurrences.

Chapter 4

Emergent Patterns in Images: A Data Mining Approach

Note: Some portions of this chapter are based on my own published work (Khan *et al.*, 2012a).

This chapter describes a technique to extract emergent patterns from a large collection of images in an unsupervised way. The data mining strategy known as itemset mining is used to extract re-occurring sets of features. Initially, features are extracted from images and then clustered to obtain visual words. These are used to create Bag-of-Words (BoW) representations for each image. Before running the mining algorithm, the images need to be in a market basket transaction format which can be created from the BoW. The mining task finds patterns which are either frequently found in many images or found in a very small number of images. In our experiments the mining process results in several interesting emergent patterns which are assigned semantic names to build semantic relationships among images containing them.

The main contribution of this chapter is to discover emergent patterns that are raised from a large unstructured image collection using a frequent itemset mining technique. The other contribution is to experiment with rare itemset mining which to our knowledge has never been experimented before on such a large collection.

4.1 Introduction

Given a set of images, we want to explore emergent patterns extracted in an unsupervised way. We want to see what kind of information these emergent patterns possess. Do they contain any semantics, or object-specific information? We are interested in emergent behavior caused by repeated feature co-occurrences across many images. If the same set of features co-occur in many images, it could be the case that these images contain similar structure. We hypothesize that investigating a very large image databases will result in the emergence of interesting patterns in a bottom-up sense. We also investigate whether these emergent patterns can be associated with the semantic meaning of images or not. To find emergent patterns we need an approach that can find co-occurring sets of features efficiently over a large number of images. Association rule mining is one such technique and has been extensively used for finding interesting patterns within items in a market basket data set.

4.2 Literature Review

Several significant works have looked at the applications of data-mining techniques for image retrieval and object recognition applications, and itemset mining is one such method. Itemset mining techniques as discussed in Section 4.3, discover both global and local knowledge from large a collection of data. These techniques can be applied to any field that produces or deals with a large amount of data. Some of the applications of itemset mining include census data analysis (Brin *et al.*, 1997), healthcare (Stilou *et al.*, 2001), and social network analysis (Lauw *et al.*, 2005). Similarly, in the case of images, the mining process deals with discovering hidden relationships in various visual primitives such as pixels, shapes, textures, or higher level features such as SIFT (Lowe, 2004) and SURF (Bay *et al.*, 2006). For these techniques, data is usually represented with a market basket transactions metaphor (Agrawal *et al.*, 1993) where each transaction is a set of items.

Association mining in general collections of data is a highly researched area, but is rather less studied in the context of image mining (Pan *et al.*, 2008). Martinet and Satoh (2007) mined relationships among objects from different modalities (video, audio, and text) of multimedia data. These objects (from images e.g., visual term (visterms) and blobs; from audio e.g., energy, pitch and tone; from text e.g., words and syntagms (words in a syntactic relationships), etc.) are called perceptual objects and are defined in a spatiotemporal window. The association rules are used to define these relationships in a more compact and semantic way. Another approach to mine frequently occurring objects (actors), and scenes in video is given by Quack *et al.* (2006). For each visual word, a transaction is created, and all neighboring visual words around this central word are considered as the items in this transaction. Once these transactions are formed, they are used for mining co-occurring objects or actors in the video. Association rule mining has also been used for clustering web images (Malik, 2006). In this work, the association rules are generated using both visual and text features obtained from web pages. The rules are used to make hypergraphs, which are clustered using a hypergraph partitioning algorithm. Pan *et al.* (2008) perform association rule mining on regions of interest (ROI) in CT images of brains. The ROIs are first extracted using a region extraction and clustering algorithm which uses domain knowledge for making these clusters. The association rules are then generated on discovered frequent itemsets considering ROIs of brain images as its items.

Quack *et al.* (2007) use association rule mining for the classification of objects. The association rules build on low-level features occurring within a bounding box which contains either a background image or an object image from one of several classes. Visual words inside this box are represented as transactions and then association rules are mined. The transaction database contains combined sets of transactions obtained from both objects and background bounding boxes. The learned rules are then used to tell the presence of a particular object class or of background in the unseen images. A similar approach is presented by Kleban *et al.* (2008) where they detect logos of different categories in the image. They locate dense configurations of frequent local features which are related to each logo class. Association rules are extracted on a spatial pyramid of each base feature. A base feature is represented as a group of all neighboring features that lie inside a grid of a fixed radius value. Each base feature is then represented as a transaction and all surrounding features as items of the transaction. Some work is also done on human action classification by Gilbert *et al.* (2008). The key idea lies in the concept of compound features which are groups of corner descriptors used to encode local features in space and time. These features are learned using data mining techniques by looking at their co-occurrences. The classifier is actually a group of these computed features and is capable of both recognizing and localizing a real-time activity.

4.3 Itemset Mining and Association Rule Mining in Images

The previous section discusses techniques that find co-occurring items in a market basket data set. It also describes applications of this technique in the computer vision domain. A majority of the work discussed in the literature focuses on using data mining techniques for a specific application. These applications include object recognition, object classification, clustering, scene recognition and content-based image retrieval. In this section, we show that semantic level features of images can emerge in a bottom-up process from a reasonably large collection of images. Although these patterns do not relate to objects, they are interesting in themselves. The methods used here are similar to the previous approaches, but the purpose is quite different.

Initially, local image features are extracted using a feature extraction technique, i.e., SIFT (Lowe, 2004), which are then clustered to generate BoW representation for each image. The BoW representation is used to generate market basket transactions. This transactional dataset is used to mine emergent patterns.

4.3.1 Frequent Itemsets Mining

Association rule mining introduced by Agrawal *et al.* (1993) is a method used for mining interesting relationships in a market basket transactions dataset. Each transaction of this dataset is analogous to a list of items that are purchased together by a customer in a grocery store.

Let $I = \{i_1, i_2, i_3, ..., i_n\}$ be the set of all possible items in the data collection (all visual words in our case). Let T be a transaction (a single image) carrying a subset of these items such that $T \subseteq I$. In our case, a transaction T contains all visual words from a single image, and D is the collection of all transactions. An association rule is an implication, $X \Rightarrow Y$, where $X \subset I$ and $Y \subset I$ but $X \cap Y = \varnothing$ (Agrawal *et al.*, 1993). Association rules have two parameters called *support* and *confidence*. The support of an itemset, *support(X)*, is the number of transactions containing the item(s). And the *support* of an association rule $X \Rightarrow Y$, that contains items from two itemsets (i.e., X and Y), is the ratio of transactions that contain $X \cup Y$ compared to the total number of transactions. The *confidence*, on the other hand, is the ratio of the transactions that contain $X \cup Y$ to the total transactions that contain X. An association rule only exists if it has a *support* greater than a minimum threshold, *minsup* and a *confidence* greater than a minimum confidence value, *minconf*. The values of both *minsup* and *minconf* are specified by the user.

The *support* of an itemset can be defined as:

$$support(X) = \frac{\text{Total\_transactions\_containing\_X}}{\text{total\_transactions}} \tag{4.1}$$

The *support* of a rule on the other hand can be defined as:

$$support(X \Rightarrow Y) = \frac{\text{Total\_transactions\_containing\_both\_X\_and\_Y}}{\text{total\_transactions}} \tag{4.2}$$

The *confidence* of a rule can be calculated as:

$$confidence(X \Rightarrow Y) = \frac{\text{Total\_transactions\_containing\_both\_X\_and\_Y}}{\text{transactions\_containing\_X}} \tag{4.3}$$

Association rule mining is a two-staged process. The first is to find all itemsets that have a frequency higher than the *minsup* value. These are named as frequent itemsets (Kotsiantis and Kanellopoulos, 2006). The second process is to generate association rules satisfying a *minconf* threshold from these frequent itemsets. The association rule generation process is usually as follows: Let $L_k = \{i_1, i_2, i_3, ..., i_k\}$ be a frequent itemset. The first rule generated is $\{i_1, i_2, i_3, ..., i_{k-1}\} \Rightarrow \{i_k\}$ and its interestingness is determined by using the confidence value. For generating the other rules, the last item in the antecedent is deleted and inserted in the consequent, and its interestingness is determined in a similar way. The process is repeated until the antecedent becomes empty. This process is very straight forward, so most research work focuses on the first problem, i.e., finding frequent itemsets (Kotsiantis and Kanellopoulos, 2006).

The first process can further be divided into two sub-processes: candidate frequent itemset generation and frequent itemset generation. In the first sub-process, we generate only those itemsets that are expected to be frequent and named them as candidate itemsets. In the second sub-process the candidate itemsets that are not frequent are pruned. The number of items in an itemset defines its length. An itemset that has k items in it is usually written as a k-itemset.

In general the itemset mining algorithm is comprised of the following three steps. These steps are iterated until there are no more frequent itemsets.

- A set of candidate k-itemset is generated by 1-extensions of the frequent $(k-1)$-itemsets that are generated previously.
- The support value for this candidate k-itemset is determined by passing over the dataset.
- The itemsets, that have support higher than the minimum support, are called frequent k-itemset while the others are rejected.

The three most important algorithms for frequent itemset mining are Apriori, FP-Growth, and Eclat (Han *et al.*, 2007). These algorithms mainly differ in terms of how efficient they are at finding frequent itemsets, but they generate similar itemsets. We used FP-Growth (Han *et al.*, 2000) to perform mining on image transactions. Unlike Apriori (Agrawal *et al.*, 1993) which first generates candidate itemsets (the most time-consuming process), FP-Growth discovers frequent itemsets without the generation of candidate itemsets, and hence is more efficient than Apriori. After generating the itemsets, the association rules are generated, and only those rules that meet the *minconf* criteria are taken into account.

FP-Growth

Several research works have used FP-Growth in various domains that show the efficacy of this approach. For example, in (Quack *et al.*, 2008) they used it for clustering a large number of images while in (Rajendran and Madheswaran, 2010) they adopted FP-Growth for the classification of brain tumors. Also, in (Yuan *et al.*, 2007b) they used this method for finding semantically meaningful visual patterns. The algorithm has two major steps:

- Build a compact data structure called a frequent pattern tree (FP-Tree). This process requires iterating over the entire transactional dataset twice.
- Extract frequent itemsets directly from the tree, which is done by traversing the tree.

Figure 4.1 shows the process for constructing a FP-Tree for the transactions listed in the table. The nodes of this tree are the items and the counter at each node signifies the number of transactions containing it. The top to bottom order of the nodes is also fixed and describes the most to least frequent item. This allows more path overlap, which results in higher compression. The tree is constructed as each transaction is read. The dotted arrowed lines (maintained as a singly linked list) locate the same item across the tree. The FP-Tree construction required just two passes over the transactions. These are the steps that are performed during the first pass.

- The data is scanned to compute the support for each item.
- All items that have support below the *minsup* threshold are rejected.
- All the remaining frequent items are sorted in decreasing order according to their support.

The decreasing order (most to least frequent item) is used while constructing the tree because it allows common prefixes to share the path down the tree. During the second pass the following steps are performed.

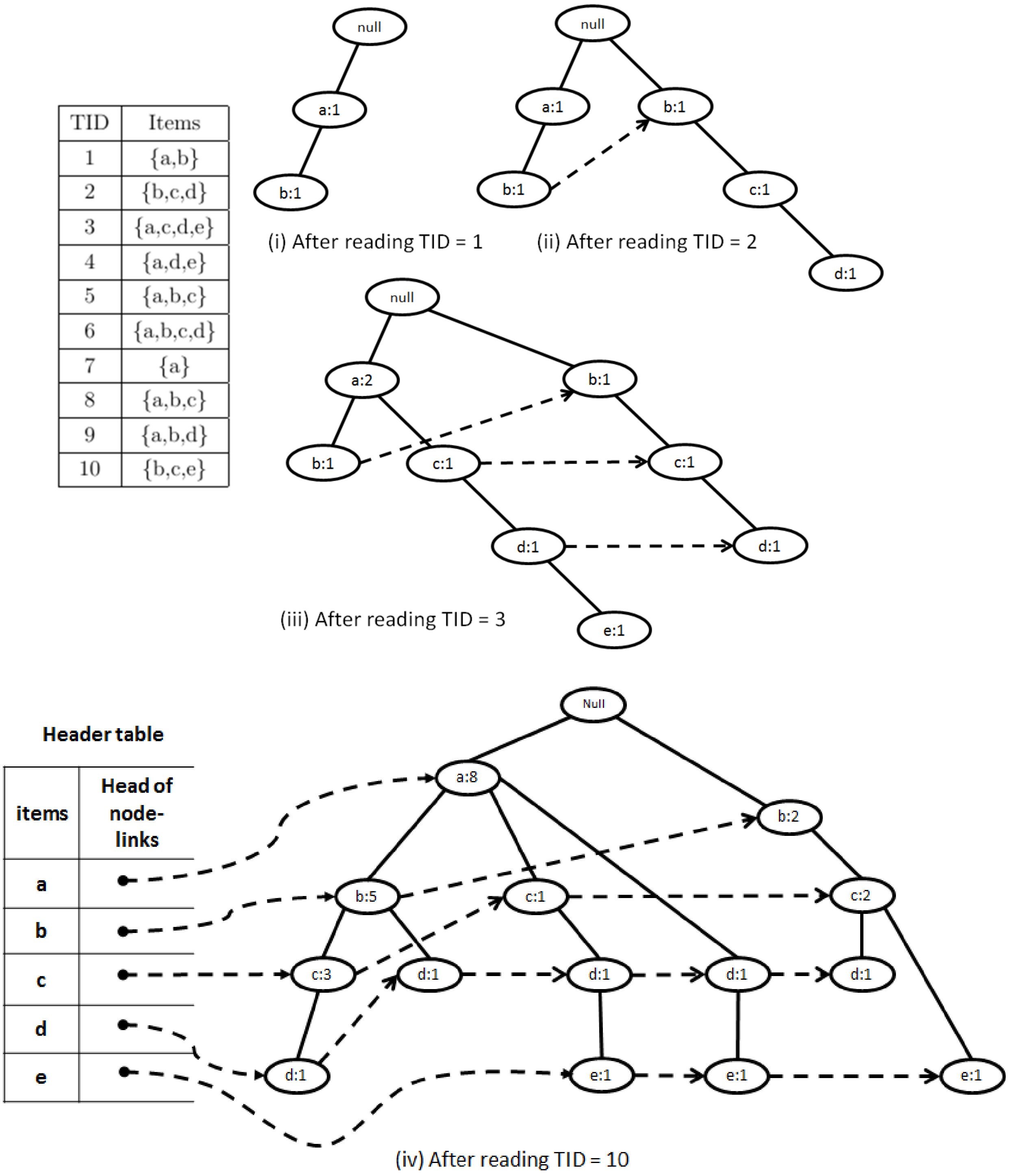
- The items in each transaction are ordered in their decreasing frequency before mapping to a path in the tree.
- The path overlaps if the sorted transaction share the same prefix. This compresses the tree and makes it easier to fit into the memory. The frequency counter within each node is also incremented during this step.
- The nodes that contain similar items are connected through pointers which are maintained in a singly linked list. A header table is maintained to store the head of the node links as shown in Figure 4.1(b).

Because of the compression strategy adopted during the tree construction phase the resulting tree generally has a smaller size than the uncompressed data. In the best case, all the transactions have the same set of items in them; hence the generated path would be a single path from the top to bottom. In the worst case, every transaction has a different set of items and the minimum size of the tree would be the same as the data size. Once the tree is constructed, the frequent itemsets are extracted by simply traversing the tree as shown in Figure 4.2.

Unlike the tree construction phase, frequent itemset generation is carried out in a bottom-up order i.e., from the leaves towards the root. To generate an itemset a divide and conquer approach is adopted as shown in Figure 4.2.

- First we find all frequent itemsets ending in *e*, then *de*, etc....then *d*, then *cd*, etc.
- The linked list is used to extract prefix path sub-trees which end in an item(set).
- Each prefix path sub-tree is processed recursively to extract the frequent itemsets as shown in Figure 4.3. All the itemsets starting from one prefix are merged.

The final list of generated itemsets is shown in Table 4.1. The main disadvantages of FP-Growth is that the tree may not fit in the memory and it's very expensive to build. As the tree is built once for the entire data set in an offline phase, we do not consider it a problem.



TID	Items
1	{a,b}
2	{b,c,d}
3	{a,c,d,e}
4	{a,d,e}
5	{a,b,c}
6	{a,b,c,d}
7	{a}
8	{a,b,c}
9	{a,b,d}
10	{b,c,e}

Figure 4.1: The process of FP-Tree construction on the data in the top table. The first column of this table contains the identity of the transaction (TID), while the second column holds the list of items in it. The header table stores the heads of linked lists, created to maintain linkages between similar items in the tree. (Image adapted from Tan *et al.* (2005))

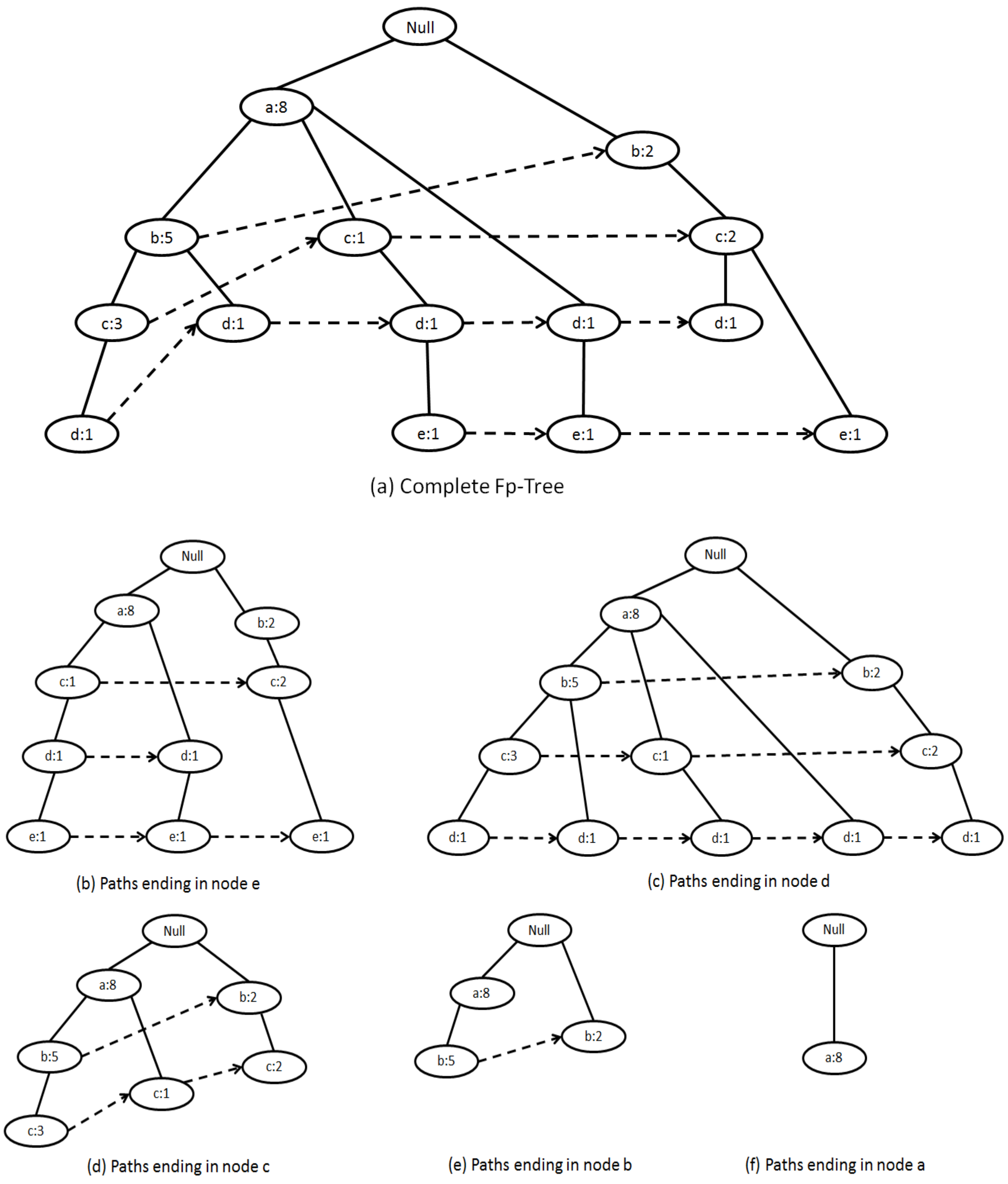

(a) Complete Fp-Tree
(b) Paths ending in node e
(c) Paths ending in node d
(d) Paths ending in node c
(e) Paths ending in node b
(f) Paths ending in node a

Figure 4.2: The process of obtaining frequent itemsets directly from the FP-Tree. (Image adapted from Tan *et al.* (2005))

Figure 4.3: Frequent itemsets generation using FP-Growth.

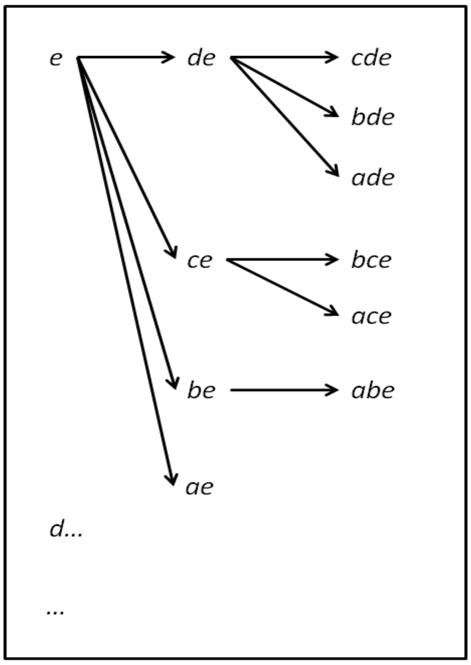


Table 4.1: The list of frequent itemsets obtained using FP-Growth.

Suffix	Frequent Itemsets
e	{e},{d,e},{a,d,e},{c,e},{a,e}
d	{d},{c,d},{b,c,d},{a,c,d},{b,d},{a,b,d},{a,d}
c	{c},{b,c},{a,b,c},{a,c}
b	{b},{a,b}
a	{a}

4.3.2 Rare Itemset Mining

Mining frequent patterns from the data is crucial, and it gives a global insight into the data. But in some scenarios a global information can be easily predicted by domain experts and hence does not necessarily give useful knowledge. For example, if we look at the records of a patient history for a fatal disease, then common symptoms can be easily mined, and most of them would likely already be known to the domain experts. In this case a more interesting finding would be to see which symptoms occur rarely or infrequently but with a high *confidence* value. A frequent itemset miner in this case would completely ignore rare itemsets because they occur in very few transactions and have very low *support* value. The obvious way to find such rare co-occurrences is to reduce the *minsup* to a very low value and then use the same frequent itemset mining algorithm. Setting a very low *minsup* will cause the algorithm to run for a very long time and will produce a substantially large number of itemsets, a phenomena known as the rare itemset problem (Tsang *et al.*, 2011) or just the rare problem. Usually, algorithms that are designed to mine rare itemsets use two threshold values rather than one, and we used the RP-Tree (Tsang *et al.*, 2011) algorithm for his purpose. We hope to find patterns or objects that have appeared in just a few images.

RP-Tree

This algorithm extracts rare patterns by building a prefix tree only for those transactions that contain at least one rare item. The algorithm is a modification of FP-Growth. Two thresholds are used for mining rare itemsets; the first called the minimum rare support (*minRareSup*), is the minimum support for an item to be a rare item and works as a noise filter. All those items that have support less than this threshold are rejected. An itemset that has support less than a minumum frequent support (*minFreqSup*) threshold but above or equal to minRareSup threshold is considered rare.

Itemsets are categorized into three different types (Tsang *et al.*, 2011): first class of itemsets consist of all items which are rare, which means that the *support* value of each item is in the range defined by *minRareSup* and *minFreqSup* thresholds. The second type of rare itemsets are those which consists of both rare and frequent items. The third type of rare itemsets consists of items that are frequent, but the support of the complete itemset is below the minimum support threshold. The itemsets of the first and the second types are considered as *rare-item itemsets* because they contain rare items in them. The itemsets, that belong to the third type are considered *non-rare-item* *itemsets* because the individual items are frequent, but in a particular combination (itemset) they are rare. The first two types of itemsets are considered more interesting than the itemset of third type (Tsang *et al.*, 2011). These types are defined as:

Consider the itemset X. It is called a *rare itemset* iff

$$support\,(X) < minFreqSup, support\,(X) \geq minRareSup \tag{4.4}$$

X is called *rare-item itemset* iff

$$\exists x \in X, support\,(x) < minFreqSup, support\,(X) < minRareSup \tag{4.5}$$

X is called a *non-rare-item itemset* iff

$$\forall x \in X, support\,(x) \geq minFreqSup, support\,(X) < minRareSup \tag{4.6}$$

4.3.3 Experimental Setup

We want to investigate emergent patterns that arise due to feature co-occurrence in a large collection of images using itemset mining techniques. For our experiments, we used the first half of the MIRFLICKR-1Million (Mark J. Huiskes and Lew, 2010) data set to fit the FP-Tree on a single machine. The dataset contains images of various resolutions of natural and everyday scenes, and it has been used in ImageCLEF (Nowak *et al.*, 2011; Thomee and Popescu, 2012) for photo annotation, concept discovery and image retrieval tasks. Some of the images from this collection are shown in Figure 4.4. The purpose of using this dataset is to see what kind of patterns arise when the majority of the images in the data set are very different from each other. For feature extraction, SIFT (Lowe, 2004) is used, and more than 0.1 billion local features are extracted. To obtain visual words approximate K-means clustering is performed on all the features for multiple values of K. These values are 5,000, 15,000, 35,000, 50,000 and 75,000 respectively. Once these clusters (visual words) are obtained all the images are represented in the market basket format.

Two different threshold values for the *minsup* are experimented when mining frequent itemsets from the dataset of 0.5 million images. The thresholds are at least 0.025%, and 0.05% of the total transactions, that means an item(set) is considered frequent if at least 125, and 250 transactions contain this item(set). For the rare itemset mining we experimented with three threshold values for both *minFreqSup* and *minRareSup* thresholds. The values for the *minFreqSup* are 0.04%, 0.05%, and 0.06%, i.e., at least 200, 250, and 300 images. For the *minRareSup* threshold the experimented values are 0.002%, 0.004%, and 0.006%, that is at least 10, 20, and 30 images. The main criterion for choosing these thresholds was so that the number of patterns generated was easily manageable on a single machine. Also, the mining process was terminated when the size of the file containing itemsets reached 10 Gbs.

Usually, the mining process generates a large number of itemsets and visualizing all of them becomes an impossible task. The association rules are generated from these itemsets to only keep the most interesting itemsets. All the association rules which have a *confidence* ≥ 0.9 are considered interesting. For an association rule of the form $X \Rightarrow Y$, the *confidence* threshold makes sure that there are at least 90% of the cases where X appears, Y also appears. The high confidence threshold signifies the strong relationship between item(s) in X and Y. This criterion reduces the total number of itemsets to a large extent but still in some cases there could be thousands of itemsets left. So 200 itemsets are chosen randomly for visualization. An issue with that is the resulting itemsets are of varying lengths i.e., they contain different numbers of items. To make sure that 200 randomly chosen itemsets contain itemsets from each length, we first count the number of itemsets of each length. We then compute the percentage of itemsets of a particular length to the total number of generated itemsets. While sampling itemsets of a particular length, the same percentage is used for choosing the number of samples itemsets.

To decide the semantic meaningfulness of itemsets, they are analyzed manually by visualizing them. This involves obtaining all the items (visual words) in the itemset and then creating a list of images containing these words. For each visual word the related features and their locations in the images are obtained and then marked on the images. All the images are examined to determine any semantics associated with them.

4.3.4 Frequent Itemset Mining Results

The mining process generated a large number of itemsets. Table 4.2 shows the total number of frequent itemsets generated for a different number of visual words. We can also see how the number of visual words affects the resulting itemsets. Smaller numbers of visual words mean that similar items repeat in many images. This results in a huge increase in the total number of co-occurrences that meet the *minsup* threshold, and hence a lot more itemsets. For some clusters such as K=5,000, and K=15,000, the pattern mining process was aborted as the size of the file containing itemsets exceeded 10 Gb.

The total numbers of itemsets for the remaining three clusters (i.e., 35,000, 50,000,

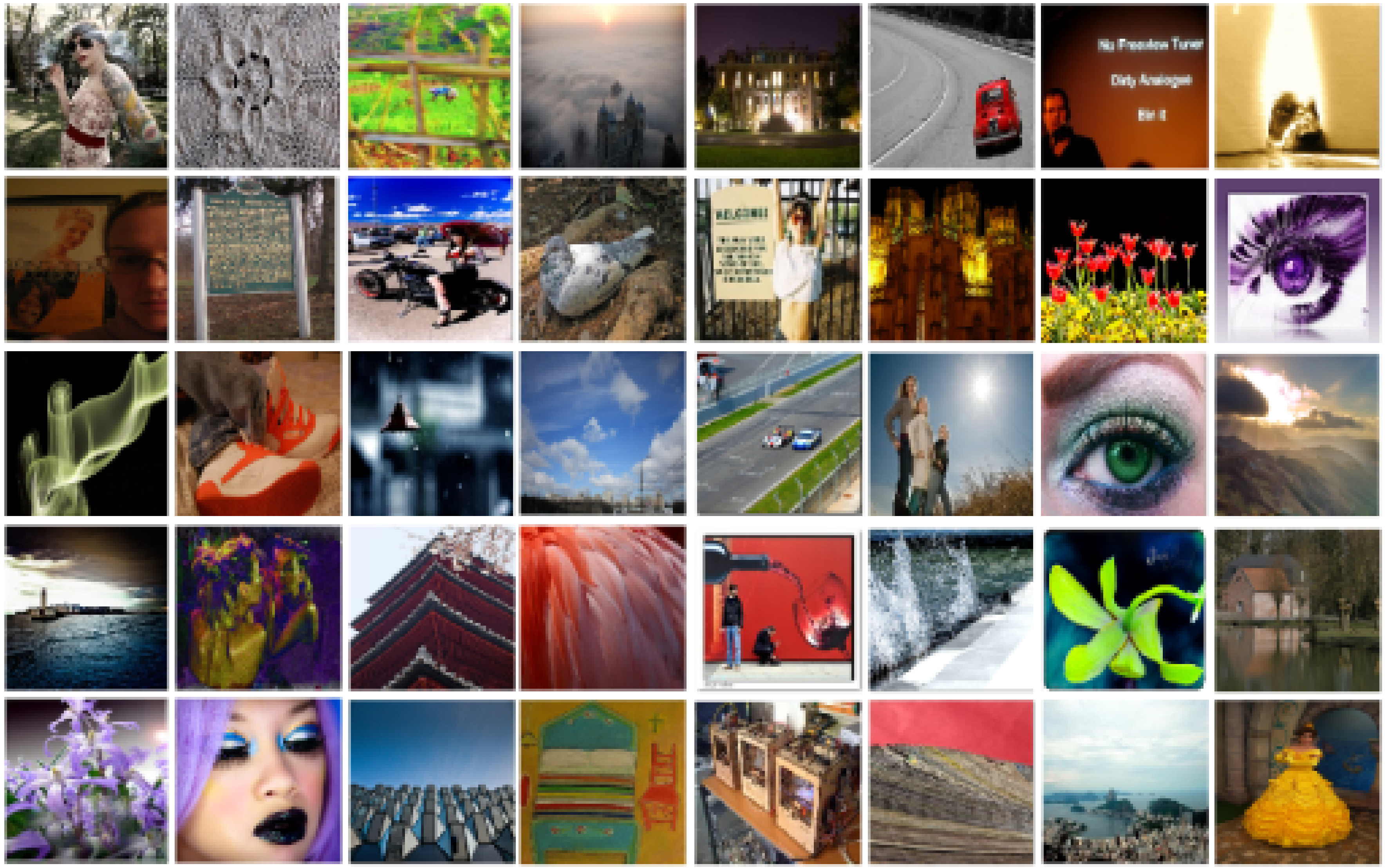


Figure 4.4: Some images from MIRFLICKR-1M (Mark J. Huiskes and Lew, 2010) 1 million images collection.

and 75,000) in Table 4.2 are still very high for visualization. So, the association rules are extracted using the *confidence* threshold 90%. This not only reduced the numbers of itemsets but also kept the most interesting itemsets that had strong relationships among its items. The resulting data is shown in Table 4.3. The table does not show the itemsets for the 75,000 clusters case as none of the itemsets met the confidence threshold criterion. Only the itemsets from 35,000, and 50,000 visual words cases are used for visualization.

Observing images containing these itemsets identifies semantically meaningful patterns. In total 6 semantic categories were formed. These are *stripes or parallel lines*; *dots and checks*; *bright dots*; *single lines*; *intersections*; and *frames* as shown in Figure 4.5 to Figure 4.10. All images in a category contain the same semantic concept, for example, all images in Figure 4.6 are from the *dots and checks* pattern category and this semantic pattern occurs at various places in each image. The images shown in figures for each emergent pattern are randomly chosen.

It is also interesting to note that apart from 35,000 visual words case (i.e., K=35,000, and minsup=0.025%), not all of these patterns are discovered with other settings of visual words. While, for the same number of clusters but using the minsup of 0.05% only the *stripes and parallel lines*, *dots and checks*, and *frames* patterns are detected. Similarly for 50,000 clusters case using both thresholds the *bright dots*, and *intersection* patterns are not discovered. One possible reason for obtaining a lower number of semantic patterns could be the high threshold on *confidence*, which is set to 90%. Once we lowered the threshold to 60%, the other patterns are detected, but at the expense of more itemsets.

We also discovered patterns that had different semantics than the already discovered six categories. The *Text1* and *Text2* patterns are examples of such a case as shown in Figures 4.11 and 4.12. Although images in both patterns contain text, the features refer to different properties in the images. For example, the majority of the features in images containing *Text1* pattern are on the letters: *o,d,c,* or *R*, which are blob-like features. Looking at the items in the *text1* pattern revealed that it had many similar items to the *Bright dots* pattern. Similarly, looking at the features in *Text2* pattern revealed that these features represent image portions that are between lines. This pattern also share items with *Stripes and parallel lines* category. It is because of these reasons that we categorized them into *Bright dots*, and *Stripes and parallel lines* categories rather than making new category.

Table 4.2: Number of frequent itemsets generated for different numbers of visual words before association rules are extracted. The mining process was aborted for some visual words when the generated file size reached 10 Gb. The data for the 5,000 visual words is not shown in the table for this reason.

Visual Words	*minsup*: 0.025%	*minsup*: 0.05%
15,000	aborted	1,085,926
35,000	988,354	32,852
50,000	427,398	14,754
75,000	86,203	3,129

Table 4.3: The numbers of itemsets resulting after generating association rules and generated from frequent itemsets for two values of cluster centers and two minsup thresholds. For the 75,000 visual words case none of the rules met the confidence threshold, so it is not shown.

Visual Words	*minsup*: 0.025%	*minsup*: 0.05%
35,000	71,645	752
50,000	2,598	556

Table 4.4: Number of rare itemsets generated for 35,000 and 50,000 clusters centers against 3 different values of *minRareSup* and *minFreqSup* thresholds. The mining process was aborted when the generated file size reached 10 GBs.

		minRareSup		
Clusters	*minFreqSup*	0.002% (10)	0.004% (20)	0.006% (30)
35,000	0.04% (200)	312	8	4
	0.05% (250)	aborted	34,212	1,358
	0.06% (300)	aborted	111,101,663	1,771,316
50,000	0.04% (200)	aborted	25,503,958	395,675
	0.05% (250)	aborted	95,198,773	1,315,304
	0.06% (300)	aborted	aborted	3,416,042

4.3.5 Rare Itemset Mining Results

In frequent itemset case, we discovered patterns that are common in many images. Rare itemset mining discovered rare patterns that are found in very few number of images. Rare patterns are formed by rare or less frequent words appearing together with high *confidence* value. As in these experiments, we have used a *confidence* value of at least 0.9, this means that words in a rare itemset appeared together more than 90% of their occurrences. Initially, rare itemset mining generated a large number of itemsets as shown in Table 4.4. To reduce the itemsets, association rules were generated from rare itemsets using the same *confidence* threshold as mentioned above.

Table 4.5 show that almost for all values of *minFreqSup* and *minRareSup*, there are still a large number of itemsets left. It is very hard to visualize them, so 200 itemset are randomly sampled as we did for frequent itemset case. After visualizing, we only found one semantic pattern i.e. *dots and checks* as shown in Figure 4.13. Apart from this pattern, there are a huge number of rare itemsets that don't show any repetitive behavior or don't have relationships within an image and hence are not given any semantic category. Also, we did not look at all the itemsets discovered as we found it too time-consuming and we did not have an efficient way to do that.

Table 4.5: Number of association rules generated for rare itemsets for 35,000 and 50,000 cluster centers. Two *minRareSup* thresholds and three different *minFreqSup* were used. The empty cell shows that no rules were found because of few number of itemsets.

		minRareSup	
Clusters	*minFreqSup*	0.004% (20)	0.006% (30)
35,000	0.04% (200)	0	0
	0.05% (250)	1,247	215
	0.06% (300)	14,120	9,616
50,000	0.04% (200)	9,515	2,243
	0.05% (250)	11,758	8,253
	0.06% (300)	78,417	26,121

4.4 Discussion and Future Work

The chapter describes itemset mining based approaches to discover emergent patterns from a large collection of images. The patterns are formed because a set of visual words commonly co-occurred together in the dataset. As every visual word in the pattern has associated features, it can be said that a pattern is formed because of repeated co-occurrence of these features. SIFT features detect image patches that have a corner, edge, and blob-like structures in them. These structures are the building block for any pattern formed using SIFT. Now looking at the patterns we got, it can be seen that the patterns certainly contain these building blocks which are repeated many times in an image. It is possible that changing the feature detection algorithm might reveal patterns that are totally different, without even changing the dataset. It is also possible that a different dataset might generate patterns similar to what we have now. We plan to do these experiments in our future work.

- What happens when we change features (e.g., MSER) that does not detect blobs or corners and see what kind of patterns are extracted
- Do we get any other pattern at all or these six are the only patterns.
- What happens if we extract patterns from Toymix?

In our experiments the itemsets generated have a different number of items in them, which ranged from 2 to 21 items. We found that itemsets having a much lower number of items in it are more semantically meaningful in both frequent and rare cases. Clustering has a direct impact on these patterns. An itemset is a combination of different items or visual words. During the clustering process it is possible that similar image features are clustered into different clusters, and hence belong to different visual words. The phenomenon is known as synonymy and is pretty common in text documents literature. If we look at images that belong to these patterns, we can see that similar features are repeated many times within each image, even though the itemset has different visual words in it. These patterns are generated because of similar features repeating within an image, and we get patterns like stripes and parallel lines, dots and checks, and intersections, etc. The other patterns that we discovered contain visual words that belong to different features, and it is difficult to assign semantic names to them.

We also find it very difficult to validate the generated patterns because of the type of the patterns and unavailability of the ground truth data for these generic patterns. In future, we also plan to see the kind of patterns we get when visual words in itemsets are forced to be of a different shape from each other. It is possible that the patterns emerged do not make sense at all due to an absence of repetitiveness.

4.5 Summary

In this chapter, we discussed a method for mining interesting patterns from a large collection of images in an unsupervised scenario. Emerging from these images were six semantic categories: *stripes and parallel lines*; *dots and checks*; *bright dots*; *single lines*; *intersections*; and *frames.* In the rare itemset case, however only one semantic category *dots and checks* emerged. Validating these patterns is a challenge because of the type of the patterns and unavailability of the ground truth data for these generic patterns.

Figure 4.5: Stripes and parallel lines. The red dots show the location of features that are in this pattern.

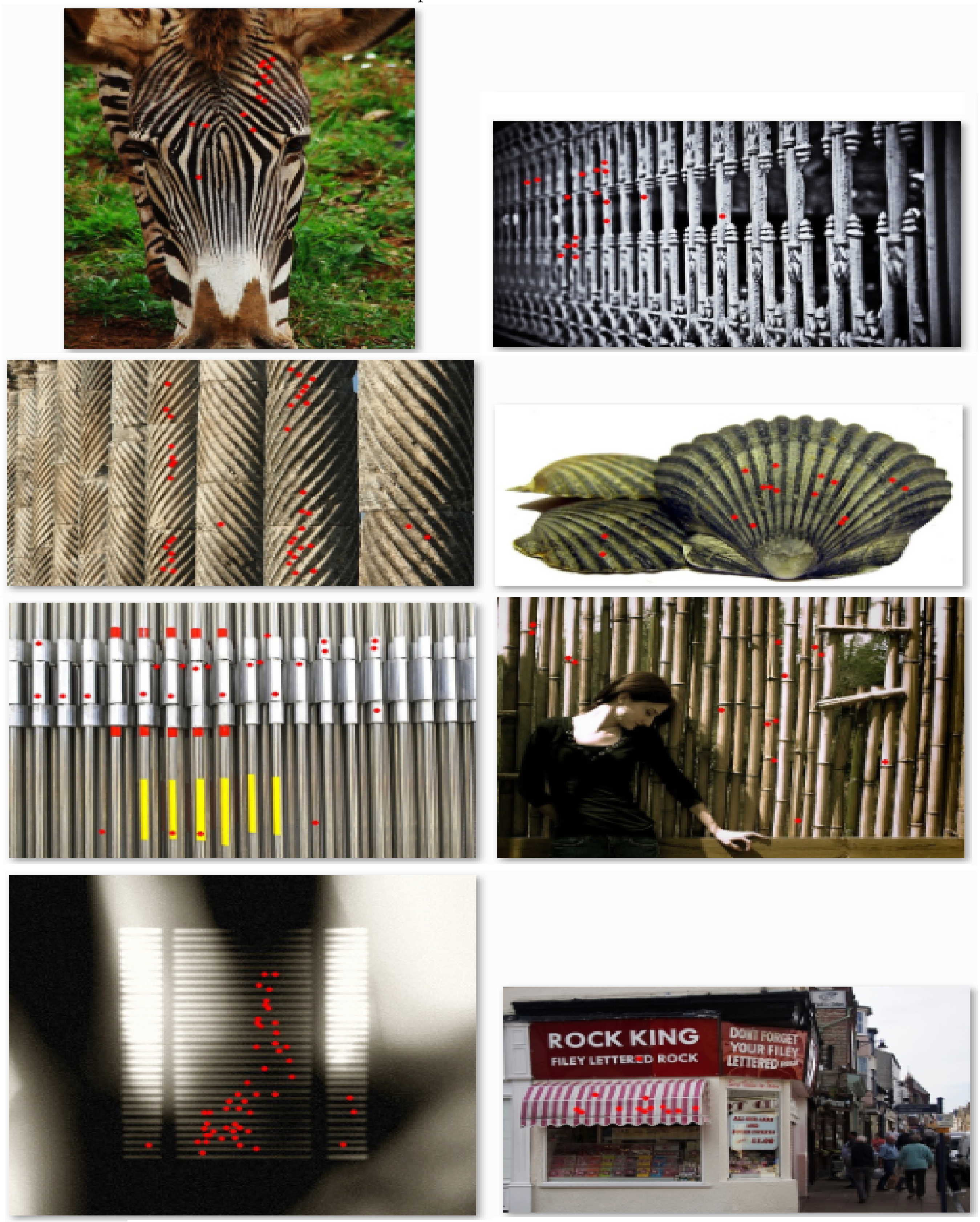


Figure 4.6: Dots and checks

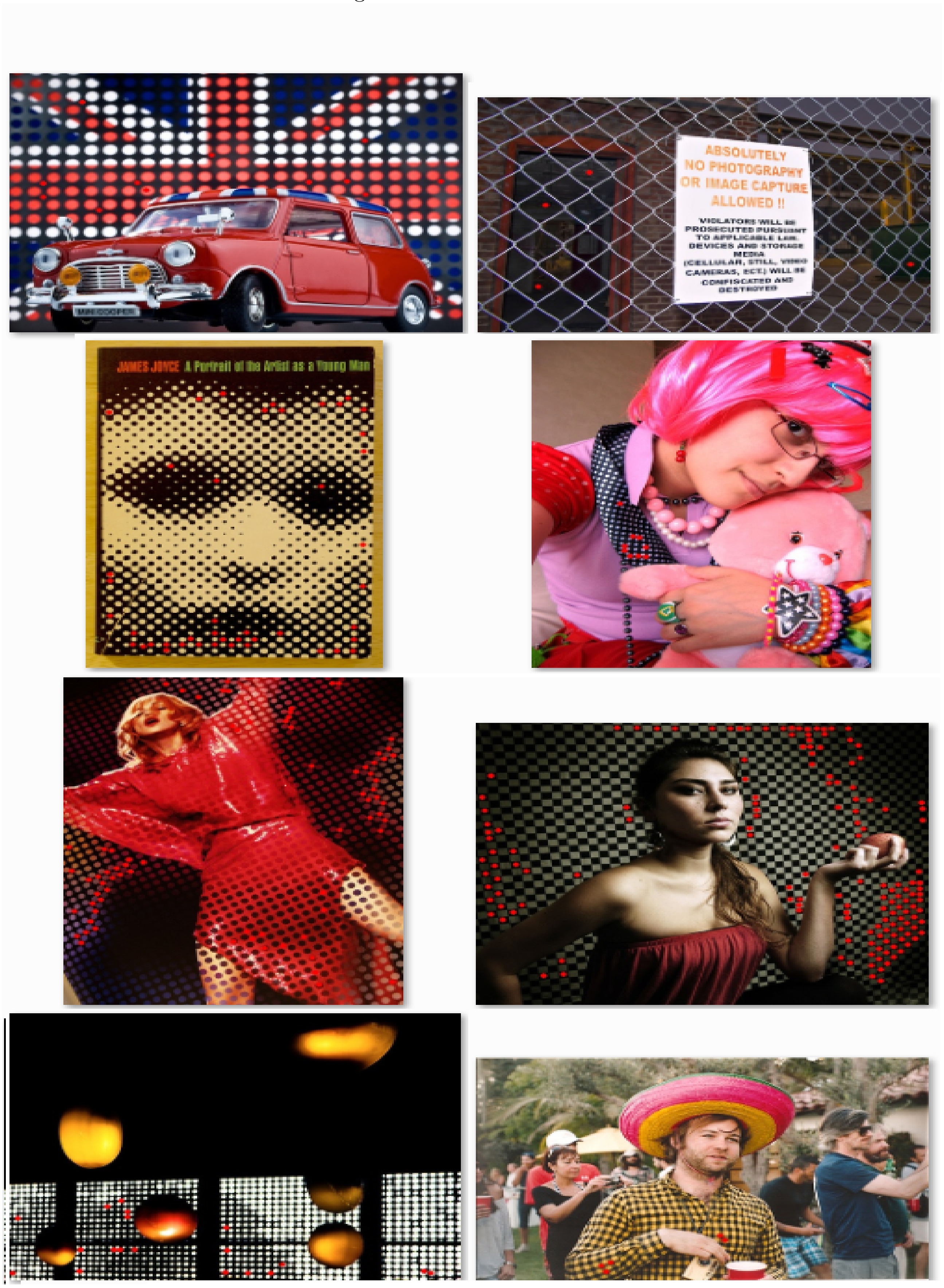


Figure 4.7: Single lines

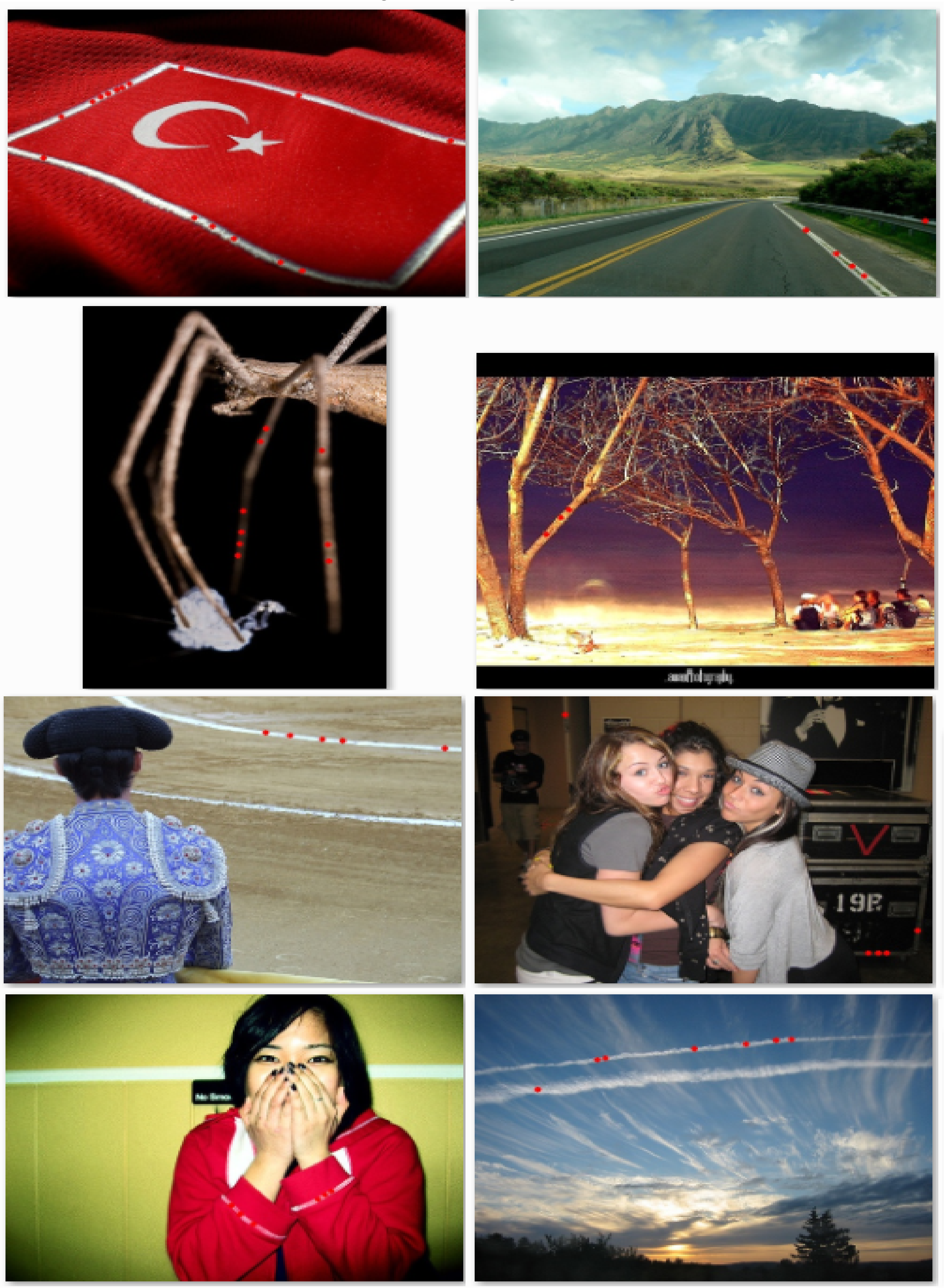


Figure 4.8: Bright dots

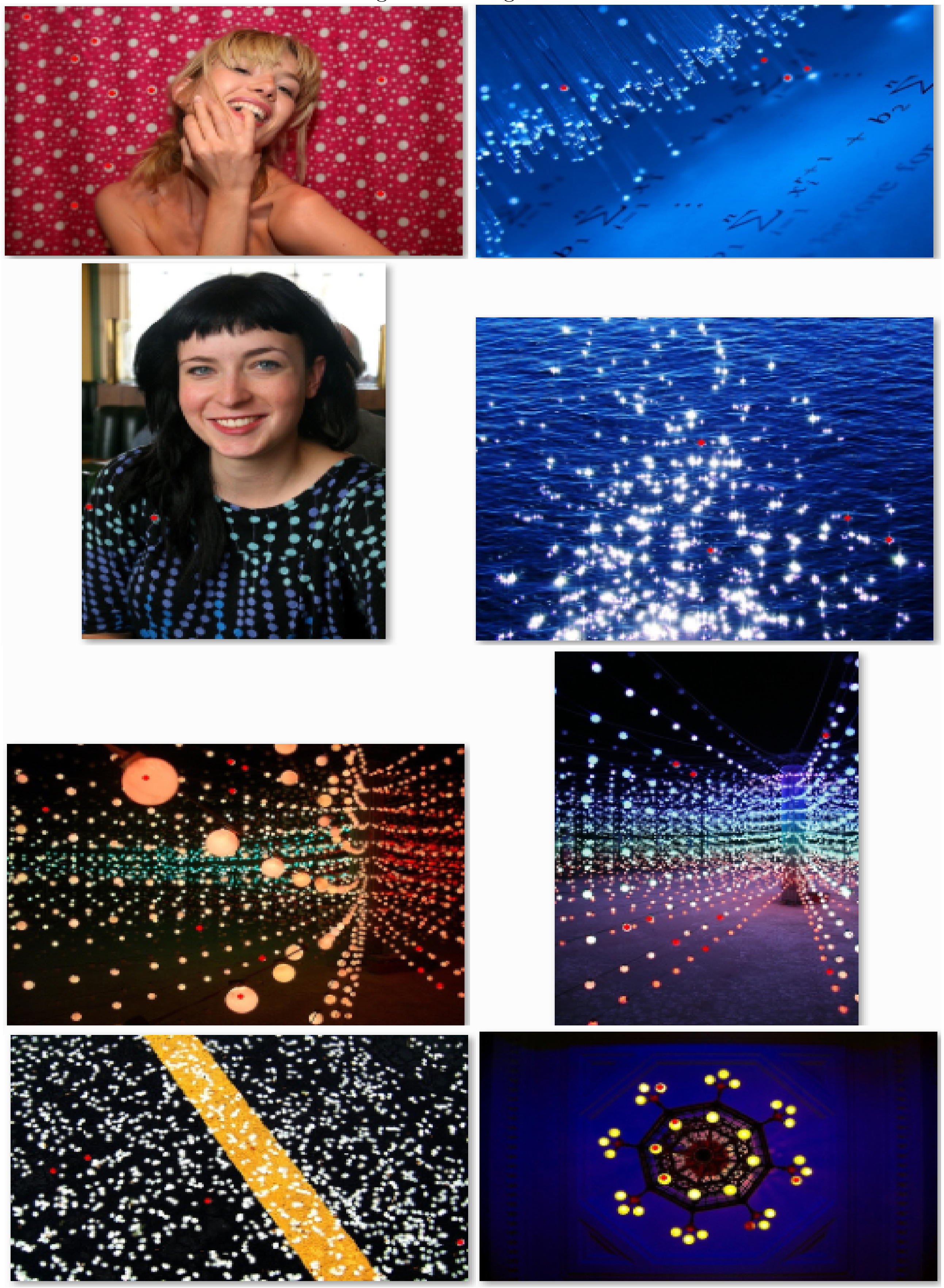
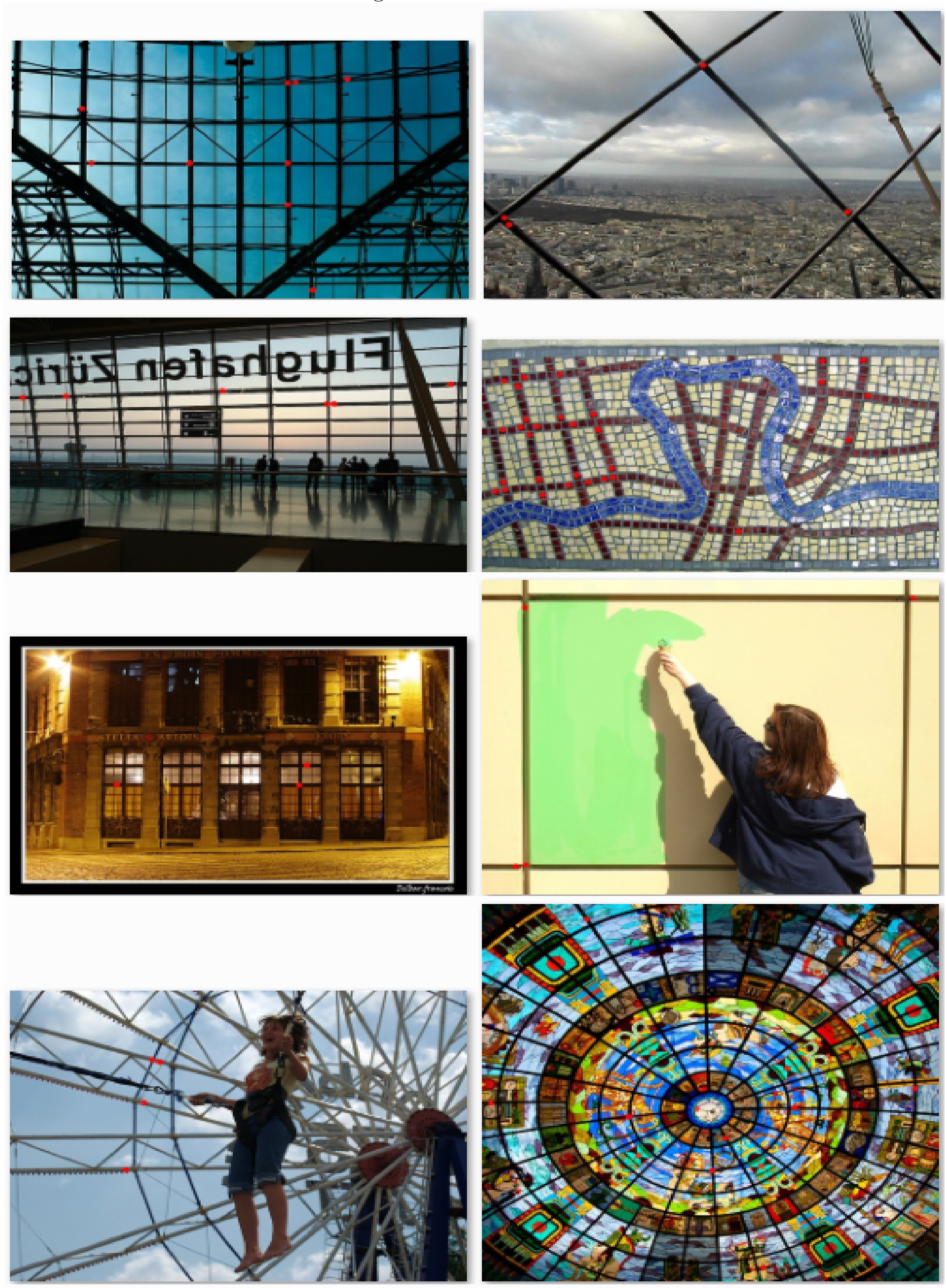


Figure 4.9: Intersections



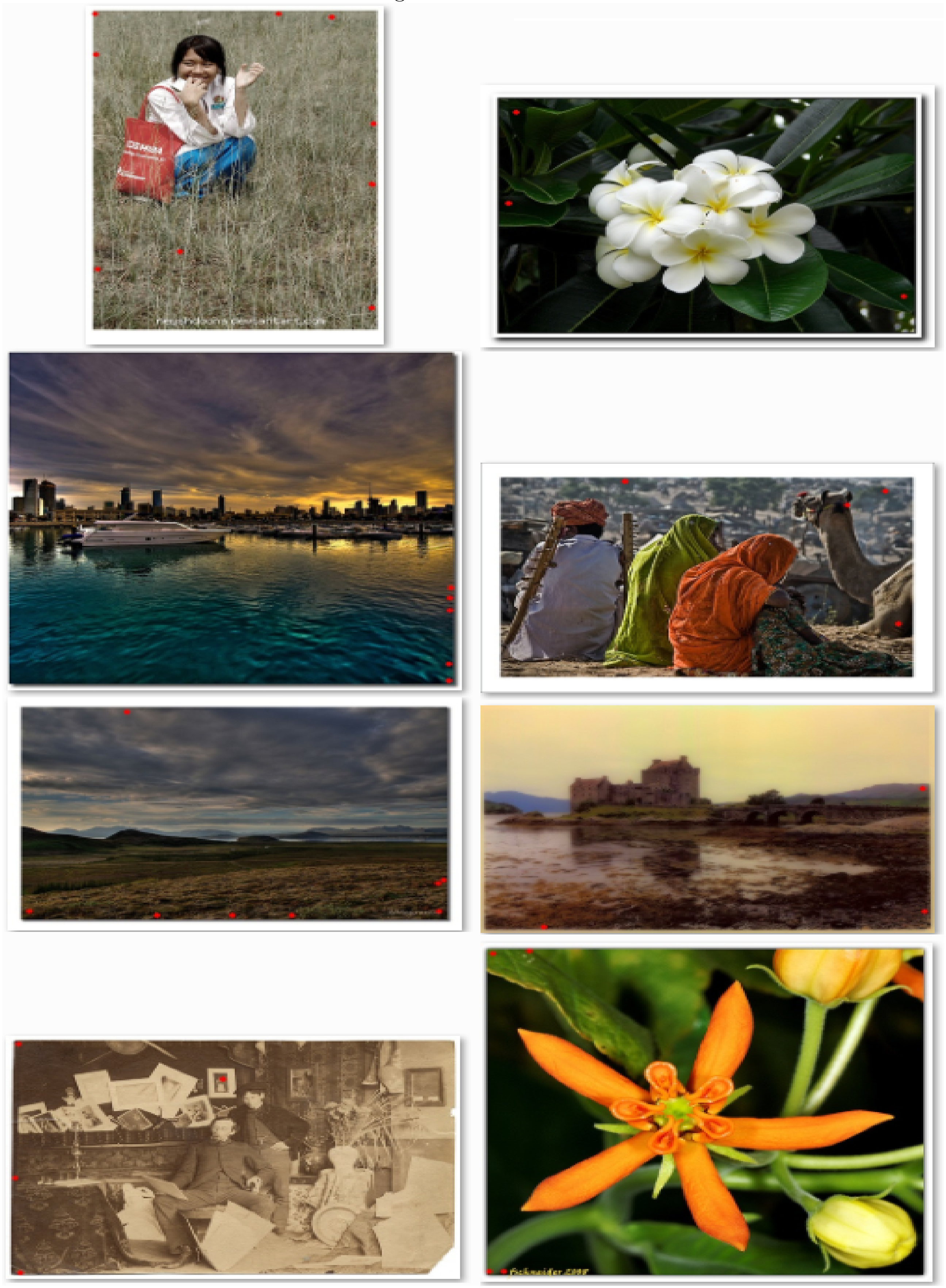


Figure 4.10: Frames

Figure 4.11: Text1

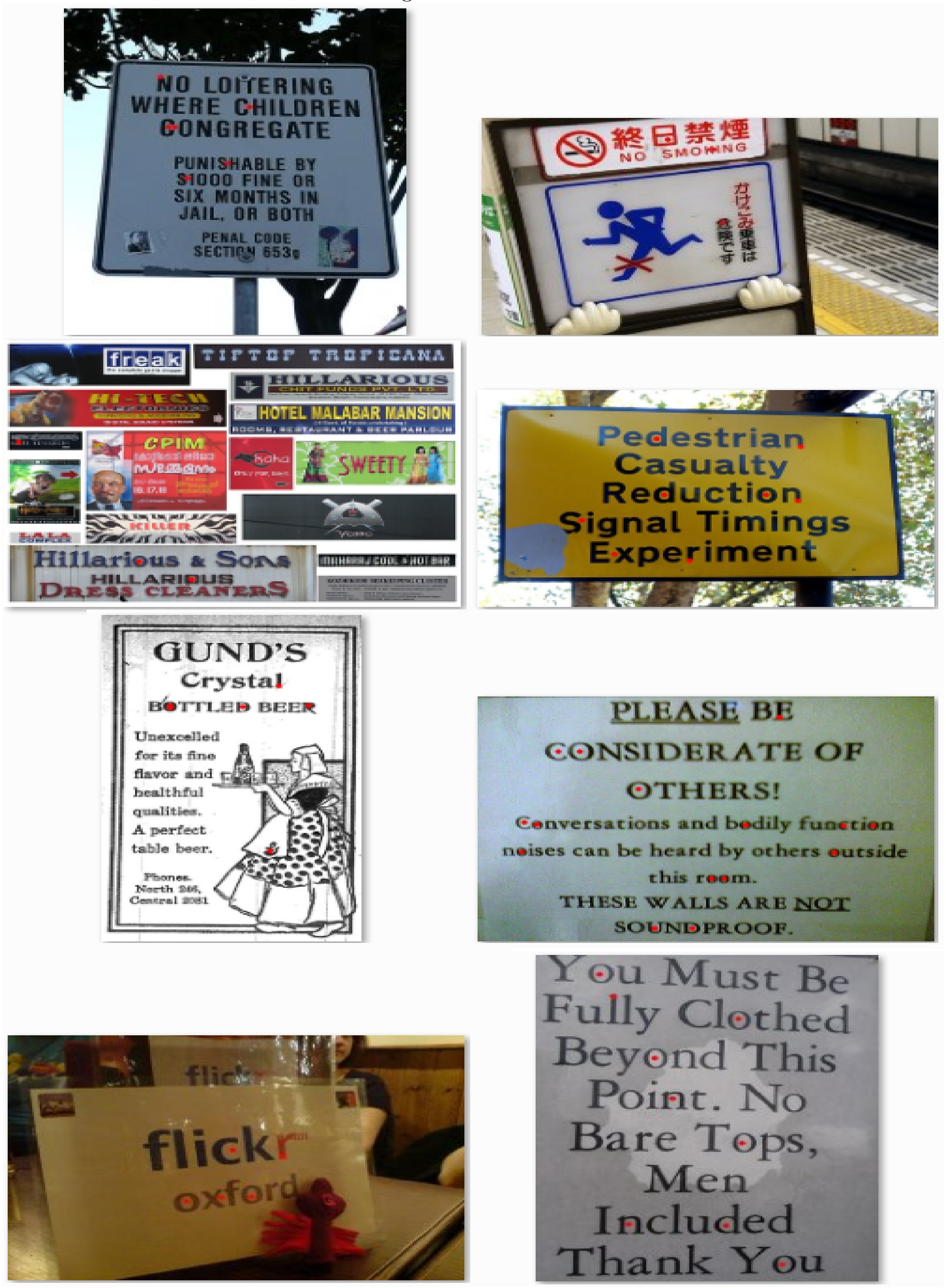
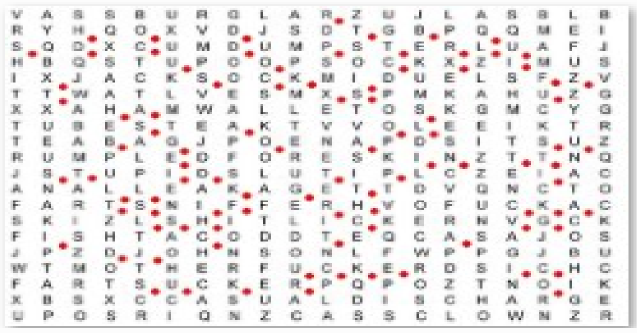
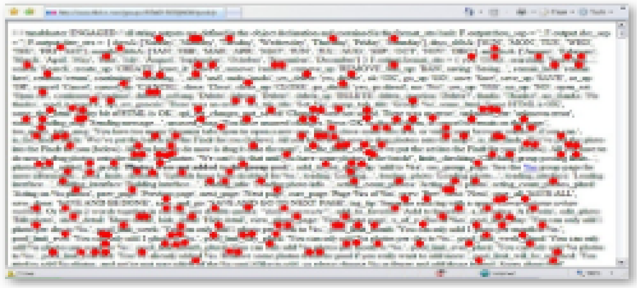
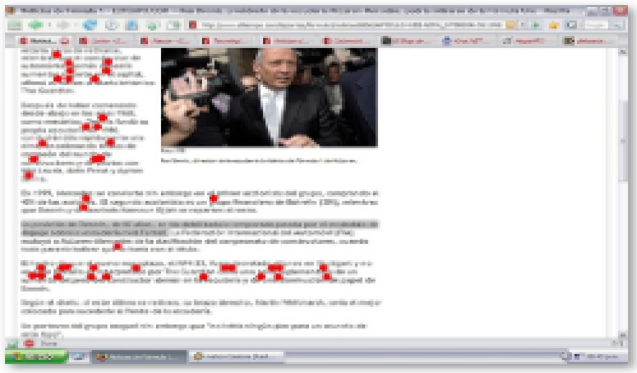
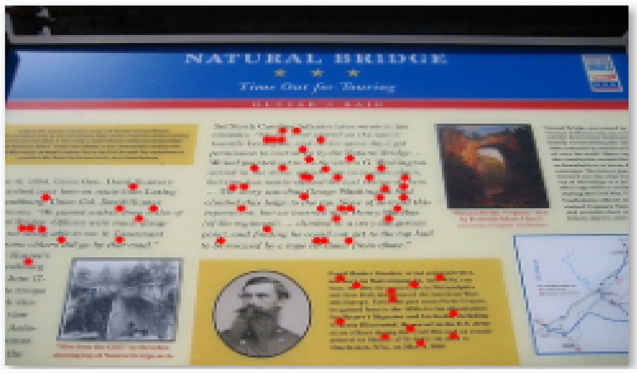
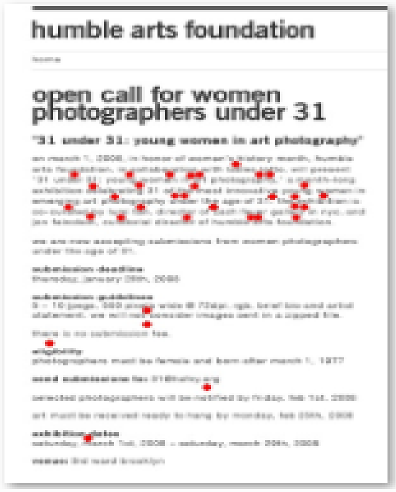
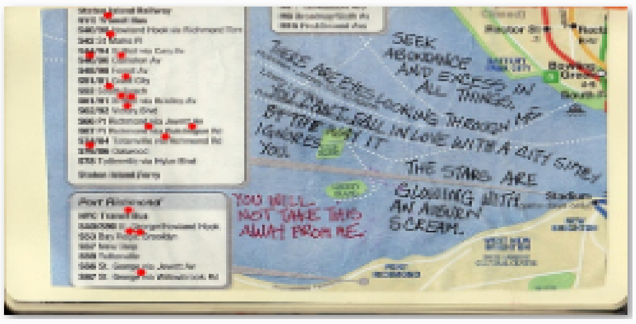

NO LOITERING WHERE CHILDREN CONGREGATE
PUNISHABLE BY $1000 FINE OR SIX MONTHS IN JAIL, OR BOTH
PENAL CODE SECTION 653g
終日禁煙
NO SMOKING
かけこみ乗車は危険です
freak
TIPTOP TROPICANA
HILLARIOUS
HI-TECH
HOTEL MALABAR MANSION
CPIM
SWEETY
KILLER
Hillarious & Sons
HILLARIOUS DRESS CLEANERS
Pedestrian Casualty Reduction Signal Timings Experiment
GUND'S Crystal BOTTLED BEER
Unexcelled for its fine flavor and healthful qualities. A perfect table beer.
Phones. North 246, Central 2081
PLEASE BE CONSIDERATE OF OTHERS!
Conversations and bodily function noises can be heard by others outside this room.
THESE WALLS ARE NOT SOUNDPROOF.
flickr
oxford
You Must Be Fully Clothed Beyond This Point. No Bare Tops, Men Included Thank You

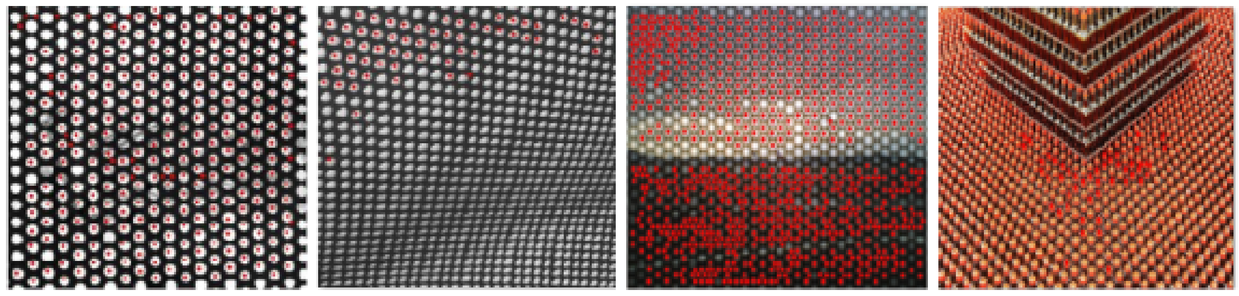
Figure 4.12: Text2



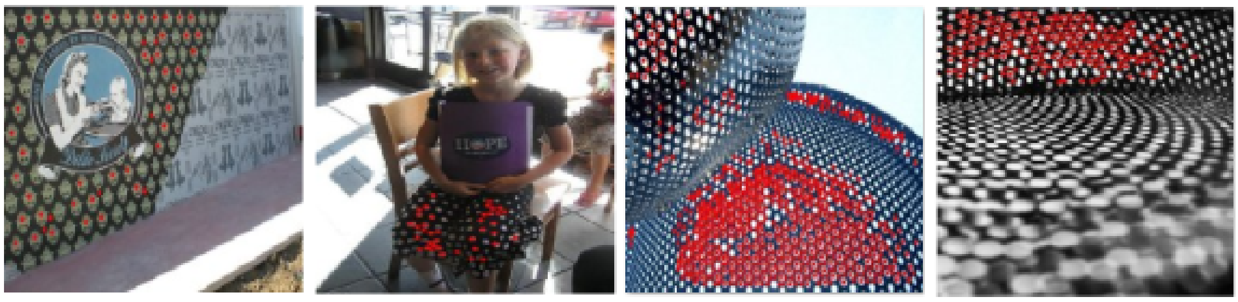




Give 'em a warm welcome . . .
Creamettes' Macaroni Cheese Bake
Creamettes
TENDER
DELICIOUS


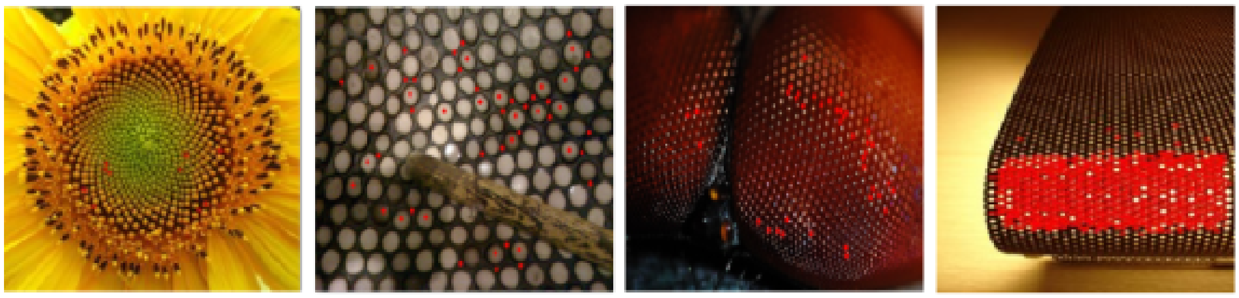

SEEK ABUNDANCE AND EXCESS IN ALL THINGS.
THERE ARE EYES LOOKING THROUGH ME
YOU CAN'T FALL IN LOVE WITH A CITY SIMPLY
BY THE WAY IT IGNORES YOU.
THE STARS ARE GLOWING WITH AN ABSENT SCREAM.
YOU WILL NOT TAKE THIS AWAY FROM ME.
Port Richmond



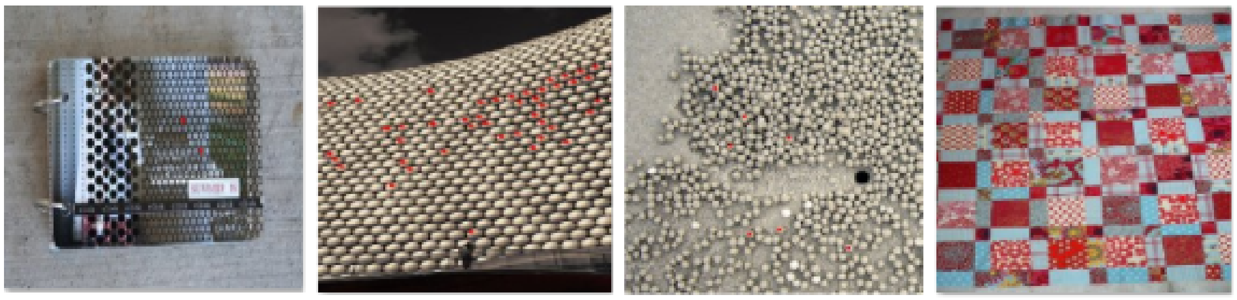
(a) Images containing 4 rare items



(b) Images containing 3 rare items



(c) Images containing 2 rare items



(d) Images containing 1 rare item

Figure 4.13: Dots and checks: The only semantic pattern observed by rare itemsets mining

Chapter 5

A Graph Based Approach for Finding Emergent Patterns

Note: Some portions of this chapter are based on my own published work (Khan *et al.*, 2014).

The previous chapter explored emergent patterns in large sets of images using a data mining technique. Six kinds of patterns emerged after running the frequent itemset mining algorithm on these images. The biggest challenge faced in the previous chapter is the evaluation of emergent patterns. The patterns are often found in multiple places and at different scales, and usually they are not associated with a particular object or a scene. Because of these reasons, evaluating these patterns becomes very challenging.

This chapter presents a novel graph-based approach to explore emergent patterns. The visual word co-occurrences obtained from each image are represented in an undirected weighted graph. A statistical analysis is performed on the co-occurrences to obtain the edge weights. These weights provide a measure of importance for edges. To obtain emergent clusters or dense subgraphs the normalized cuts algorithm is applied. One of the problems faced in the previous chapter is the evaluation of extracted patterns. This chapter addresses this problem by following a two-stage process. First, the performance is tested on a simple image dataset whose ground truth information is known a priori. After the desired results are achieved, the method is tested on a complex image dataset. We show that in simple datasets the emergent clusters can identify object classes, while experiments on a complex dataset result in various interesting patterns.

The main contribution of this chapter is the statistical analysis based technique that assigns an importance score to each edge. Other contributions include: a method to represent visual word co-occurrence information in a graph; a strategy to find emergent clusters; and a two staged process to verify the performance of the presented approach.

5.1 Introduction

Chapter 3 discusses strategies for discovering visual patterns in images using a bottom-up approach, and Chapter 4 uses itemset mining based techniques (i.e., Frequent Itemset Mining and Rare Itemset Mining) to discover emergent patterns. This chapter also explores a bottom-up strategy that is based on a graph theoretic approach. A graph can be used to represent relationships among visual primitives. To do so, vertices of the graph usually represent visual primitives (e.g., features, or visual words), and the edges represent relationships between them. A graph-based approach then aims to identify or separate out different subgraphs formed by strong relationships among vertices. Throughout this chapter, the terms vertices or visual words, and edges or co-occurrence are used interchangeably.

This chapter presents a novel approach that uses graphs for finding emergent patterns. As described earlier emergent patterns usually arise due to sets of co-occurring visual words that appear together many times. If visual word co-occurrences are represented in a graph then frequently co-occurring vertices form clusters or subgraphs having strong relationships among them. Several strategies, such as graph partitioning or edge cut techniques can be then applied to obtain these clusters. A common problem that can affect the semantics of the obtained clusters is related to visual words. Visual words have different properties e.g. they can be synonyms (many visual words describing the same part of objects), or exhibit polysemy (visual words having more than one distinct meanings) (Quelhas *et al.*, 2007b; Tirilly *et al.*, 2008). Such visual words can add unnecessary co-occurrences (edges) among vertices from different clusters, and hence add noise. Because of this noise, identifying these emergent clusters is a challenging task. The approach presented in this chapter facilitates the cluster identification process by assigning an importance score to each edge as its weight. Using only a subset of edges that are more important than others, helps in reducing this noise and identifying emergent clusters. In the previous chapter we discussed the difficulty of evaluating emergent patterns. In this chapter we follow a principled way of evaluating the proposed method. Initially, the performance is tested by experimenting with images that are very simple, with ground truth information available for comparison. In later stages, the experiments are repeated on a complex dataset.

5.2 Literature Review

Grauman and Darrell (2006) present an approach to learn object categories from unlabeled images. They first find feature correspondences between images which are used to calculate affinities among them. To partition this data spectral clustering is applied and the resulting partitions are used to train a classifier for different objects categories. An accuracy of 94% is achieved on four object categories from the Caltech-4 data set.

Kim *et al.* (2008) create a graph directly from image features. Link analysis techniques such as PageRank (Brin and Page, 1998) and vertex similarity algorithm (Paul *et al.*, 2004) are used to obtain object category information from this graph. They achieved the classification accuracy of 95.42% on six objects categories from Caltech-101 dataset (Fei-Fei *et al.*, 2004a). In another experiment that is performed on three objects categories from the TUD/ETHZ datasets[1], a classification accuracy of 95.47% is achieved.

In an approach Zhao and Yuan (2011) present a method to find thematic patterns in video using a cohesive subgraph mining method. A thematic video pattern is defined as a subset of visual words that are spatiotemporally collocated. In their approach, the overall mutual information scores among spatiotemporal visual words are maximized. The method is capable of finding various thematic patterns despite changes in scale, viewpoint, color, and illumination, or partial occlusions.

The majority of graph theoretic approaches including the techniques discussed here, focus on finding visual patterns that are related to object and scene categories. Our aim is to find emergent patterns. As discussed before emergent patterns may or may not relate to objects and scenes categories, but rather represent some complex and generic structure in the images.

The most relevant graph-based approach that finds patterns similar to this work, is presented by Gao *et al.* (2009). Their method finds consistent associations of image features using a minimal cost bipartite graph matching technique. The cost defines the spatial consistency of the feature pairings. The patterns are discovered regardless of their spatial proximity. The obtained higher order patterns are invariant to translation, scale, and rotation because of the way spatial relationships between visual words are encoded. Some of the consistent associations that they discover appear on windows, face elements (e.g., eyes, lips, nose and hairline etc.), and grocery images.

[1]The TUD dataset is available at http://www.pascal-network.org/challenges/VOC/ and ETHZ Giraffes at http://www.vision.ee.ethz.ch/datasets.

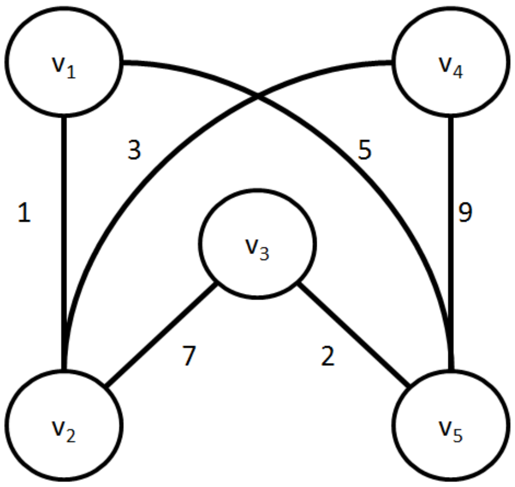


Figure 5.1: A graph containing five vertices $V = \{v_1, v_2, v_3, v_4, v_5\}$ and six edges. The numbers on the edges show their weight.

Section 5.3 of this chapter explains some basic concepts related to graph theory. Section 5.4 discusses our overall methodology, covering each step in detail. Creating a co-occurrence graph is explained in Section 5.4.1 and a strategy for performing statistical analysis on these co-occurrences is discussed in Section 5.4.2. Experiments and results are presented in Section 5.5.

5.3 Graph Theory

A graph, G, can be represented by a pair of sets, (V, E), where V is the set of vertices of the graph and E is the set of edges among these vertices. A sample graph is shown in Figure 5.1. An edge $e_{ij} = (v_i, v_j)$ of the graph (represented by solid line) links two vertices (circles), $v_i, v_j \in V$. In an undirected graph, the relationship between two vertices is direction-less. This means for an edge (v_i, v_j) the order of vertices is not important i.e., (v_i, v_j) and (v_j, v_i) represent the same relationship. However, in a directed graph every relationship has a direction (represented by a solid arrowed line) and a relationship exists only in the direction of the arrow. Hence, the order of vertices is very important i.e., two edges (v_i, v_j) and (v_j, v_i) do not represent the same relationship. A graph may be weighted by associating a weight, w_{ij}, with each edge e_{ij}. Figure 5.1 shows an undirected weighted graph with five vertices.

Graphs can be represented using an adjacency matrix or adjacency list. An adjacency matrix as shown in Figure 5.2 is a common way of representing graphs. Rows and columns of the matrix represent vertices of the graph. Elements of the matrix indicate the presence or absence of an edge between two vertices. In a weighted adjacency matrix, the elements of the matrix describe the weight of an edge. An adjacency matrix for a weighted graph, $G = (V, E)$ with n vertices is a $n \times n$ matrix A, and can be defined as:

$$A_{ij} = \begin{cases} w_{ij}, \text{if } (v_i, v_j) \in E \\ 0, \text{otherwise.} \end{cases} \tag{5.1}$$

An adjacency list is a memory-efficient way of representing a graph and is suitable for sparse graphs. In this representation, a linked list is maintained to keep a record of all the vertices that are connected to a vertex. Additional information, e.g., edge weights can be stored at each node.

The adjacency matrix representation has some advantages: It is simple and easy to understand, It provides very fast lookup to find out whether there is an edge between two vertices or not. Also, it is very easy to implement. However, for a sparse graph (a graph with very few edges) memory wastage is a big problem. Also for a graph with billions of vertices it is often difficult to fit it into main memory. An adjacency list representation has less memory wastage and is easier to fit in memory. The biggest disadvantage of this approach is longer lookup time because of the traversal of the linked list. Figure 5.3 displays an adjacency list for the same graph shown in Figure 5.1.

5.4 Methodology

This section explains major steps involved in our approach.

5.4.1 Co-occurrence Graph Generation

Emergent patterns usually are the result of visual words that appear commonly in the entire image collection. To identify emergent patterns using a graph based approach, all the visual words co-occurrences appearing in each image are represented in a graph. For each image we start from its Bag-of-Words vector, which contains information about visual words appearing in it along with their occurrence frequencies. Two words co-occur together if they are in the same image irrespective of their location in the image. Each word is paired with all of the following words to determine all the co-

	V1	V2	V3	V4	V5
V1	0	1	0	0	5
V2	1	0	7	3	0
V3	0	7	0	0	2
V4	0	3	0	0	9
V5	5	0	2	9	0

Figure 5.2: A graph adjacency matrix for the graph shown in Figure 5.1. The zeros in diagonal positions represent that the graph does not allow vertices to have edge to themselves.

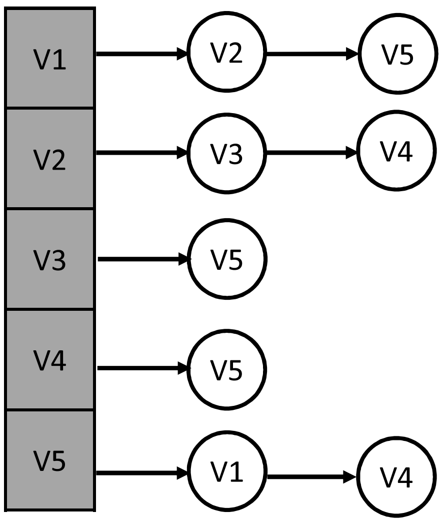


Figure 5.3: A graph adjacency list for the same graph shown in Figure 5.1.

occurrences in an image as shown in Figure 5.4. A co-occurrence defines an edge of the undirected weighted graph $G = (V, E)$ and is described as a pair of words. All the visual words of the vocabulary form the vertices V of this graph. As this is an undirected graph, two edges $(4, 5)$, and $(5, 4)$ are considered the same.

This process is repeated for all images in the dataset, and all observed co-occurrences are represented as graph's edges. The weight w_{ij} of an edge is incremented if a co-occurrence is observed in multiple images. The weight w_{ij}, of an edge, $e_{ij} \equiv (v_i, v_j) \in E$ is equal to the number of images containing both v_i and v_j.

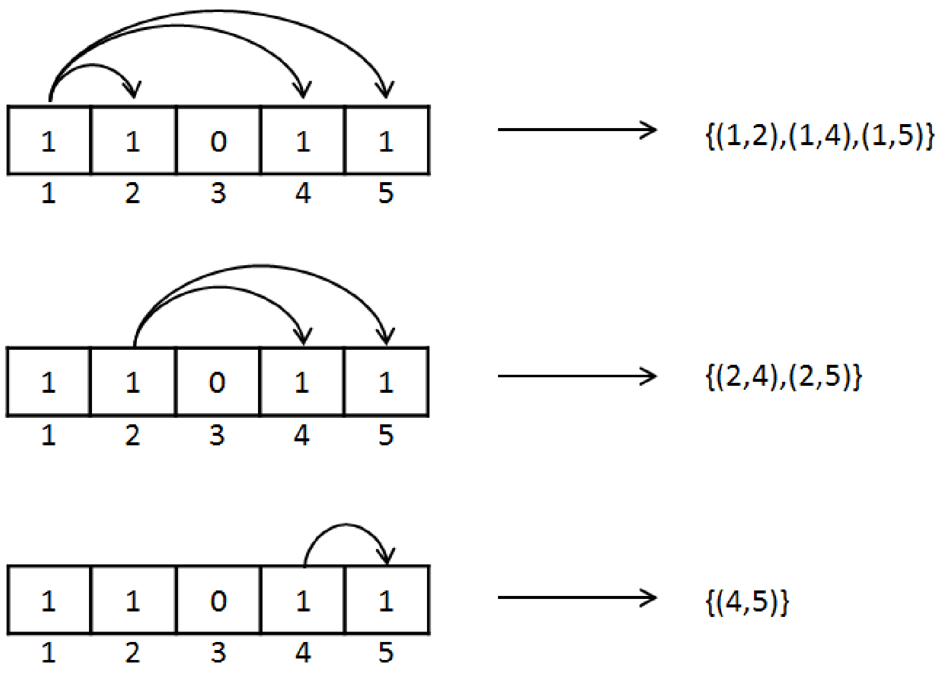


Figure 5.4: A toy example showing the process of obtaining co-occurrences from the BoW of an image.

5.4.2 Statistical Analysis

A co-occurrence graph generated using this approach captures all observed co-occurrences. The co-occurrences that appear in large number of images carry high edge weights compared to co-occurrences that appear rarely. These frequent co-occurrences make dense clusters of vertices while less frequent co-occurrences often link vertices from different clusters. The goal is to identify these dense clusters. The less frequent co-occurrences on the other hand add noise and make the identification difficult. In this case an approach that assigns an importance score to each co-occurrence can be helpful. We can then keep, the top-n (i.e., n most important) co-occurrences and feed the remaining graph to a partitioning algorithm. Instead of using raw frequency of a co-occurrence as the importance score, we present an approach to identify co-occurrences that are statistically more significant than others. The previous weights of the edges are replaced by these new scores.

Significant edges are chosen based on the binomial test. The null hypothesis is that visual words appear independently in each image. Those words that co-occur much more frequently than expected are the result of some activity in images and hence are chosen to be significant edges. The process starts by computing the probability, $P(v_i)$, of each word occurring in a random image as:

$$P(v_i) = F_i/N, \tag{5.2}$$

where F_i is the frequency of the word, i.e., the number of images containing this word, and N is the total number of images in the data set. The probability of any two words co-occurring together is calculated by computing the joint probability, $P(v_iv_j)$. As visual words are assumed to appear independently (null hypothesis) in each image, their joint probability can be calculated as:

$$P(v_iv_j) = P(v_i)P(v_j). \tag{5.3}$$

For a binomial formulation, the joint probability, $P(v_iv_j)$, represents the probability of success, or the chance that this co-occurrence will occur in an image. The probability of failure, $Q(v_iv_j)$, can be calculated as:

$$Q(v_iv_j) = 1 - P(v_iv_j). \tag{5.4}$$

We can calculate the probability of r (the number of images containing a co-occurrence) successes in N trials (the total number of images) using the binomial distribution as:

$$P[r, N] = \frac{N!}{r!(N-r)!} P(v_iv_j)^r Q(v_iv_j)^{(N-r)}. \tag{5.5}$$

The mean μ, and standard deviation σ, of a binomial distribution can be calculated as:

$$\mu = NP(v_iv_j), \text{ and } \sigma = \sqrt{NP(v_iv_j)Q(v_iv_j)} \tag{5.6}$$

We are interested in identifying co-occurrences that occurred more than their chance of occurrence. A z-score provides a way to calculate how far a particular occurrence is from a mean, in standard deviation units. The higher a z-score is, the more significant this co-occurrence is. The z-score is computed as:

$$z=\frac{x-\mu}{\sigma} \tag{5.7}$$

Where x (also called the score) represents the total number of images containing a co-occurrence. It is computed by iterating through all the Bag-of-Words vectors BoW_m, of images and counting the number of images that contain both words in the co-occurrence. For each co-occurrence, the total number of images containing it are counted, and a z-score is then computed. In a co-occurrence graph, the z-scores are assigned as the new edge weights. Details of this process are given in Algorithm 1.

Algorithm 1: Statistical analysis for finding significant edges

```
Data: A graph $G = (V, E)$ where an edge is a co-occurrence of two words and
      edge weight encodes the frequency of this co-occurrence
Result: Statistically significance score for each edge
1  for each $v_i \in V$ vertex in the graph do
2      Find probability of each vertex:
3      $P(v_i) = \frac{F_i}{N}$
4  for each edge $e_k \in E$ in the graph do
5      Compute joint probability of vertices $v_i$ and $v_j$ co-occurring together
        $P(v_i v_j) = P(v_i)P(v_j)$
6      Compute the probability of failure $Q(v_i v_j) = 1 - P(v_i v_j)$
7      Compute the mean of the binomial distribution $\mu = NP(v_i v_j)$
8      Compute the standard deviation: $\sigma = \sqrt{NP(v_i v_j)Q(v_i v_j)}$
9      for each Bag-of-Words vector $BoW_m$ of dataset images do
10         Compute $x = \begin{cases} x+1, \text{if } (BoW_{mi}, BoW_{mj}) > 0 \\ x, \text{otherwise.} \end{cases}$
11     $z_{ij} = \frac{x-\mu}{\sigma}$
```

5.4.3 Graph Visualization

The co-occurrence graph can be visualized to see its inherent structure using various graph visualization packages such as Gephi (Bastian *et al.*, 2009), GraphViz (Gansner and North, 2000), and iGraph (Csardi and Nepusz, 2006). This allows us to see dense subgraphs or clusters before applying a graph partitioning method. We used Gephi for visualization. If the graph is too dense or unable to fit in memory, only the top-n most significant co-occurrences are selected for visualization, which reduces unimportant edges and often reveals the underlying structure of the graph.

5.4.4 Normalized Cuts and Graph Partitioning

After eliminating insignificant edges, graphs may still contain unwanted links between highly connected clusters. We want to obtain all the dense clusters in the graph by removing edges between them. Normalized cuts (Shi and Malik, 2000) is used to partition the graph into the desired number of clusters.

The normalized cut takes care of both the total dissimilarity between different groups of vertices as well as the total similarity of vertices within the groups. For example, a weighted graph $G = (V, E)$, can be partitioned into two sets A and B, such that $A \cup B = V$, and, $A \cap B = \emptyset$, by removing or cutting edges connecting two parts. The decision where to place a *cut* depends on the degree of dissimilarity between the two pieces and can be measured as:

$$cut(A, B) = \sum_{u \in A, v \in B} w(u, v). \tag{5.8}$$

An ideal bipartition of the graph would be the one where this cut is minimum. Using this criterion alone for dissociation between groups, favors small sets of isolated vertices in the graph. To remove this bias, the authors present a new measure of dissociation between groups called *normalized cut (Ncut).* This criterion computes the cost of the cut as a fraction of the total edge connection to all the vertices in the graph. Now with this criterion the cut that partitions small isolated points gets a larger Ncut value.

$$Ncut(A, B) = \frac{cut(A, B)}{assoc(A, V)} + \frac{cut(A, B)}{assoc(B, V)}, \tag{5.9}$$

where *assoc(A, V)*, and *assoc(B, V)* are the total connections from vertices in A and B, to all vertices in the graph respectively.

$$assoc(A, V) = \sum_{u \in A, t \in V} w(u, t), \text{ and, } assoc(B, V) = \sum_{v \in B, t \in V} w(v, t). \tag{5.10}$$

In their approach they suggest formulating the minimization of this criterion as a generalized eigenvalue problem, and the eigenvectors can be used to obtain good partitions of the graph.

They describe the following two partitioning algorithms: recursive two-way Ncut and a simultaneous K-way cut with multiple eigenvectors. The recursive two-way Ncut approach is a hierarchical partitioning technique, creating a tree of partitions. The root of this tree contains all vertices of the graph. At each level, a partitioning problem $(D - W)x = \lambda Dx$ is solved for eigenvectors with the K smallest eigenvalues. Here W is the affinity (weight) matrix of the graph and D is a diagonal matrix containing the sum of the weights incident at each vertex. Only the second smallest eigenvalue is then used to bi-partition the graph as proposed by Shi and Malik (2000), although the next few eigenvectors also contain useful partitioning information. For each subgraph they again solve the partitioning problem and similarly use the second smallest eigenvalue to partition it recursively down to a fixed level. One obvious disadvantage of this approach that it is computationally wasteful as it only utilizes the second smallest eigenvector and ignores all others. For the Simultaneous K-way cut with multiple eigenvectors approach instead of using the second smallest eigenvalues, the top K eigenvectors are used (Shi and Malik, 2000).

For experiments in this chapter, the simultaneous K-way cut approach is used. The affinity matrix W can be defined as:

$$W_{ij} = \begin{cases} z_{ij}, \text{if } (v_i, v_j) \in E \\ 1, \text{otherwise.} \end{cases} \tag{5.11}$$

Where z_{ij} is the weight (z-score) of the edge calculated during statistical analysis. We use 1 as the default weight of diagonal elements as advised by (Shi and Malik, 2000).

5.4.5 Image Ranking

Once we have different partitions or emergent clusters containing sets of visual words that co-occur significantly more than their expectation, it is desirable to visualize these clusters. All images containing the words from each clusters are identified. Every image contains a subset of the significant co-occurrences from a cluster. The total number of significant co-occurrences that an image has is set as the rank of that image. The higher the total number of significant co-occurrences an image has, the higher the rank it gets. A higher rank means that the image is closer to the cluster. The number of edges in an image is also influenced by the total number of visual words in an image. The ranking process is described in Algorithm 2.

Algorithm 2: Displaying ranked list of images found in each cluster.

```
Data: Clusters along with visual words in them, and co-occurrence graph.
Result: A rank associated with each image describing its closeness to a
        cluster.
1 Generate an inverted index, $Idx$, for each word in the graph by computing a
    list of images containing it.
2 for each cluster do
3     Obtain a list of words in that cluster.
4     Obtain the significant co-occurrences that this cluster has.
5     for each co-occurrence do
6         Create a list of images that contain both words using the inverted
            index, $Idx$.
7     for each image that contains words from this cluster do
8         The total number of significant co-occurrences that this image has,
            gives the rank for this image.
9     Images are sorted based on this rank and visualized.
```

5.5 Experimental Setup and Results

In our previous method we use data mining techniques for finding emergent patterns. One of the problems that we faced was validating the generated patterns. The biggest hurdle was the unavailability of the ground truth data for these images. For example, the types of emergent patterns that are generated (e.g., stripes, dots and checks, bright dots etc.) were not known before, and that makes evaluation even harder. For the approach described here we follow a more principled approach and split the experimentation process into two stages. We first investigate emergent clusters generation on a very simple image dataset for which the ground truth information was already known. For this purpose we created a dataset (i.e., Toymix) containing 6000 images.

The dataset contains images of six objects as shown in Figure 5.5. There are 1000 images for each object. The dataset has low complexity, because objects in the images have small transformations (rotations, translations and scale changes). There are quite a large number of images per object and they also contain similar visual words. This causes visual word co-occurrences to repeat. As a result we expect emergent patterns to reflect object categories that we already know, and so can be validated. If the words in emergent clusters are related to these object categories, we can decide that the important co-occurrences contribute towards finding emergent patterns.

Once we get the clusters that can be linked to object categories, we can now experiment with a more complex scenario. We choose the Caltech-101 (Fei-Fei *et al.*, 2004b) dataset and use 20 object categories from it. These categories are chosen because they had at least 80 images. This produces a total of 1600 images from the 20 categories. In addition, we just use the first 80 images per object even when more images are available. This allows all categories to have equal weight and removes any bias toward categories having many images (e.g., face category has 435 images). Figure 5.6 shows images from each of the twenty categories. We extract SIFT (Lowe, 2004) descriptors from images that are quantized into 10,000 visual words to represent each image as BoW.

5.5.1 Graph Building and Significant Co-occurrence Selection

Once the co-occurrence graphs are built we found that there are significant differences in the total number of co-occurrences in both graphs for the same number of visual words. We do this analysis before applying the statistical analysis, and choosing top-n significant co-occurrences. For the Toymix dataset there are approximately 26 million edges, and for the Caltech-101 dataset there are 46 million edges. This significant difference is because of the images in the dataset. The Toymix dataset contains images that have very small transformations among images within an object category. As a result, there are many co-occurrences that repeat in many images. In Caltech-101 dataset, images within a single category have large variations. These variations cause images to have a different set of co-occurrences that results in producing a much larger number of co-occurrences. Figure 5.7(a) and 5.7(b) shows the edge weight distribution for the graphs containing 10,000 vertices generated for each dataset. The maximum edge weights are z-scores of 68 and 35 respectively for Toymix and Caltech-101 data sets. This can be linked back to the phenomenon explained in previous paragraph.

5.5.2 Graph Visualization Results

For visualization, edges that have high z-scores are more significant because such edges appeared more than expected and so should contain more important information. Up until this stage, no graph cutting or partitioning technique has been applied. The graphs are visualized after choosing 10,000, top-n co-occurrences. The resulting graphs are visualized using Gephi (Bastian *et al.*, 2009). The graph layout algorithm that is used is called OpenOrd (Martin *et al.*, 2011). Figures 5.8 and 5.9 show graphs generated from Toymix and Caltech-101 datasets respectively, containing 10,000 co-occurrences. Vertices of these graphs are visual words and are color coded by assigning a label to each word according to their dominant object category. For each word its occurrence count for each object category $c_i \in C$ is computed. The word occurrence count is the number of images from a category that contain this word. A word is assigned a particular category if it appeared most of the times in images from this category. This is done by computing a ratio of the highest category count, to the second highest category count, $r = \frac{\text{Second highest category count}}{\text{Highest category count}}$. For a word if this ratio, $r > 0.6667$, then this word is assigned the category in which it mostly appeared. A default category $c_{def} \notin C$ is assigned in cases where this criterion is not meet.

The first graph, shown in Figure 5.8 is for the Toymix dataset and contains six object categories. When viewed using a graph drawing layout algorithm, six natural categories emerge, without applying any graph partitioning or edge cutting technique. The words in each emergent cluster are dominated by a single category as depicted by different colors for each cluster. The colors used here are similar to the colors chosen for each category in Figure 5.5. As we can see, most of the clusters are completely separated from each other, while the two remotes (that are visually similar) have many overlapping edges. This intuitively makes sense because words in these clusters are very similar to each other. Surprisingly, these clusters have many similarities with the ball cluster. This could be because of small alphabetic letters in the ball images have a blob-like structure that is similar to the buttons on the remotes.

The second graph, depicted in Figure 5.9, is for 20 categories of Caltech-101 dataset. The graph has a complex structure but still there are some clusters. By looking at the colors of these clusters it is very difficult to find out any cluster that has words from a single object category. These clusters are a lot more dense, and do not show a clear separation of different objects categories because of many linking edges. It is very interesting to see these clusters emerging before applying any graph clustering approach. As these clusters are not well separated, assigning any category to them is not possible until the linking edges are removed. We use normalized cuts algorithm that focuses on removing these linkages between the clusters.

5.5.3 Graph Partitioning Results

The clusters shown in Figures 5.8 and 5.9 still have edges between them. Spectral graph clustering using normalized cuts is applied to the graphs to separate the clusters completely. For the simultaneous K-way cut approach K is selected as $K = 6$ and 20 for Toymix and Caltech-101 dataset respectively. The value of K is set according to the number of categories in each dataset.

For the Toymix dataset, the normalized cuts separated each cluster into a single dominating object category as shown in Figure 5.10. For words in each resulting cluster, images containing these words are read from an inverted index that is already been created. The resulting images are marked with the co-occurrences they have and total co-occurrences in each image are counted. This count serves as rank of each image. Refer to Algorithm 2 for more detail about this ranking procedure. A maximum of top 200 images are saved for each cluster in separate directories. The results show that, for some clusters, most of the top-ranked images are from one object category. For the Caltech-101 dataset, setting $K = 20$ splits the graph into twenty emergent clusters. When visualized, five of these clusters have images that are related to a single object category as shown in Figure 5.11 and 5.12. Some other clusters are related to some patterns and did not contain a single object category. These clusters are named as: background; stripes; corners; and borders patterns. The images containing these patterns are shown in Figures 5.12 and 5.13, and are bounded by red rectangles. Most of the remaining clusters have fewer numbers of co-occurrences, and contain images from different categories. We did not see any other emergent cluster related to any object category apart from the ones discussed here. We think the reason is that there is too much variation among images within a single category. It is interesting to see that emergent clusters are not only related to human-labeled object categories, but also various other interesting patterns.

5.6 Discussion

Chapter 5 described an approach that encodes visual word co-occurrences in a weighted undirected graph and applies statistical analysis on its edges to identify emergent edges or co-occurrences. The statistical analysis technique allows us to choose co-occurrence which appeared more than their chance of occurrence and are good candidate while looking for emergence. Hence they are grouped using a clustering approach to form emergent patterns. We found that these emergent patterns can represent objects and many other interesting patterns.

5.7 Summary

In this chapter, an approach to find emergent patterns in image datasets is presented. The approach represents co-occurrence of all the visual words from images in an undirected weighted graph. In this graph, emergent patterns lead to dense clusters of vertices having high edge density. Applying our statistical criteria to these co-occurrences assigns an importance score to them. This score is later used only to keep the top-n most important co-occurrences. Initial experiments are conducted on a simple image collection to validate the approach. We show that in the simple image datasets with low complexity our approach results in class identifiers. This assures that our method of finding importance score aids emergence. Later experiments on a more challenging datasets like Caltech-101 reveal that emergence can result in various interesting patterns including, but not limited to, some object categories.

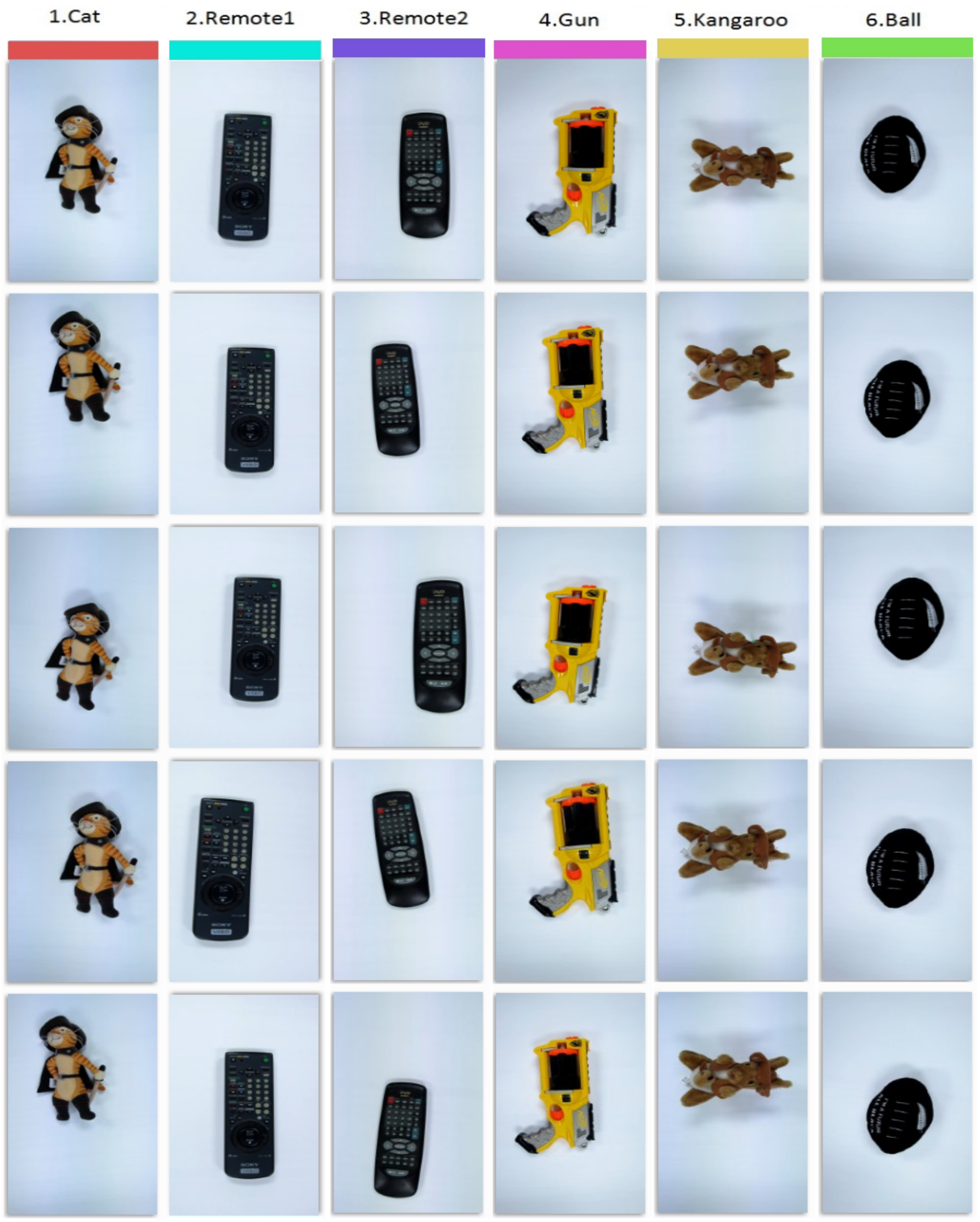


Figure 5.5: The Toymix dataset contains multiple instances of six objects. The figure shows five randomly chosen instances of each object. The color bars at the top of each image links each object to one of the cluster in the graph shown in Figure 5.8.

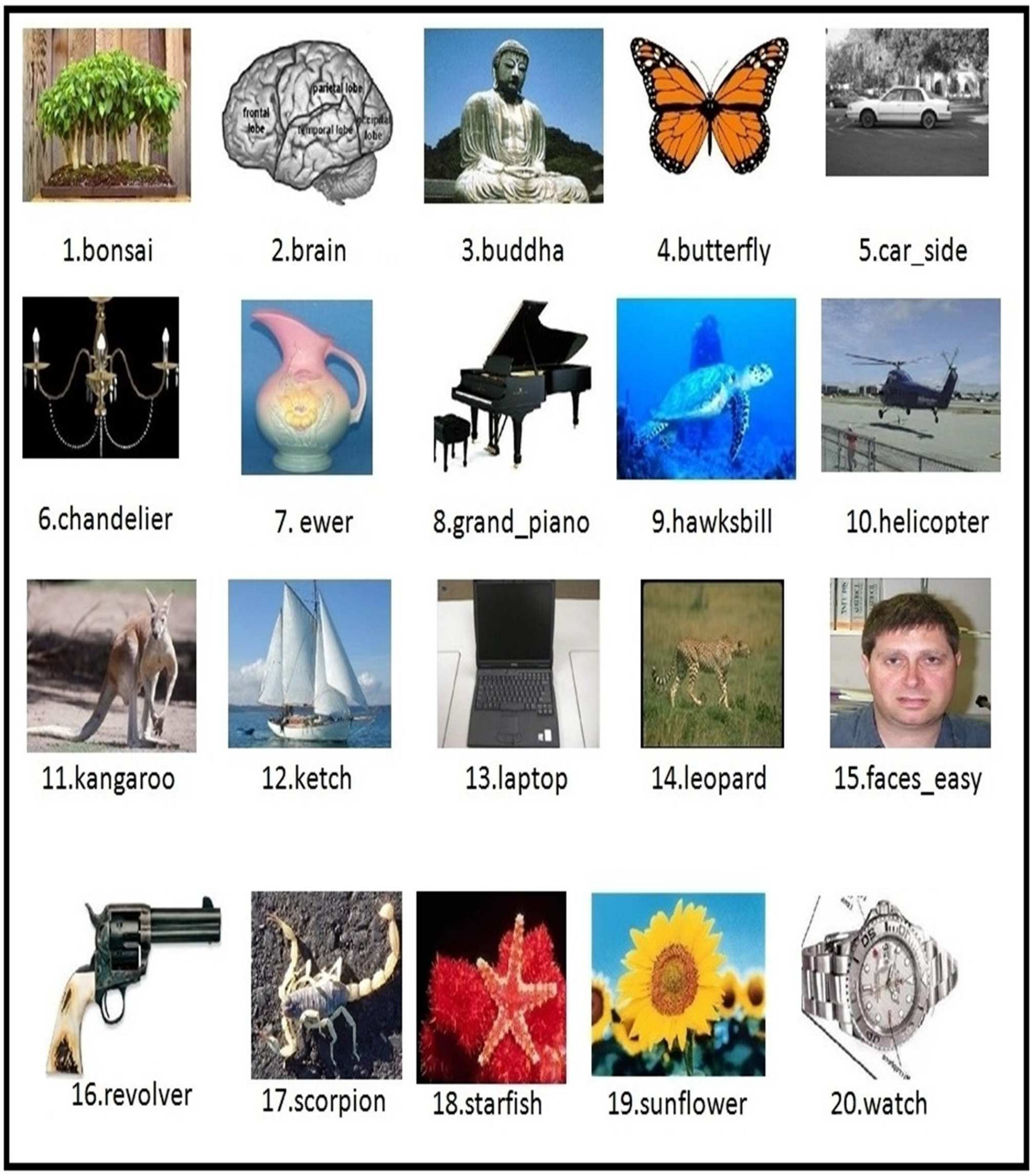
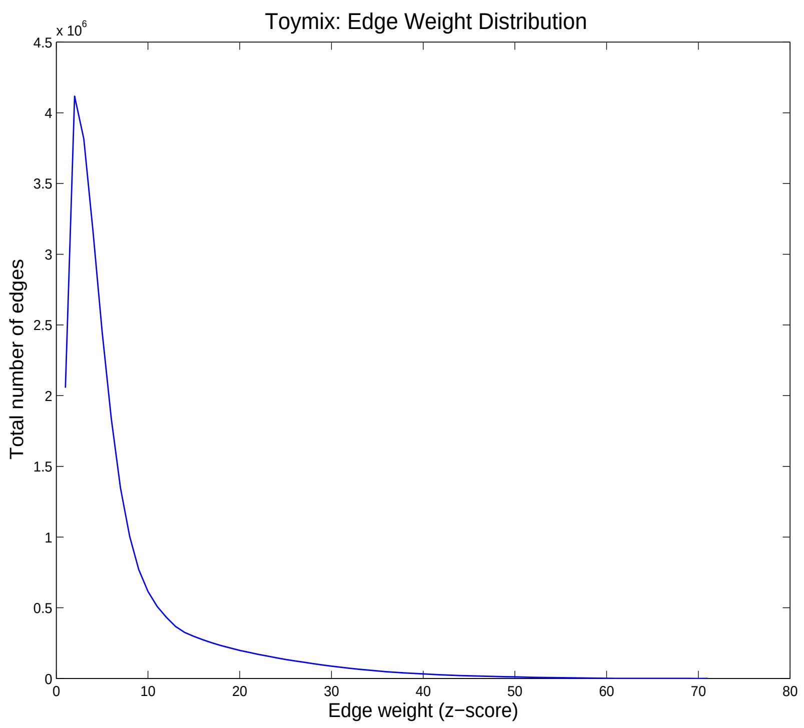


Figure 5.6: The twenty object categories selected from Caltech-101 dataset. Each of this category contains different instances of the category object type e.g., faces of different persons, different pianos, or helicopters etc.



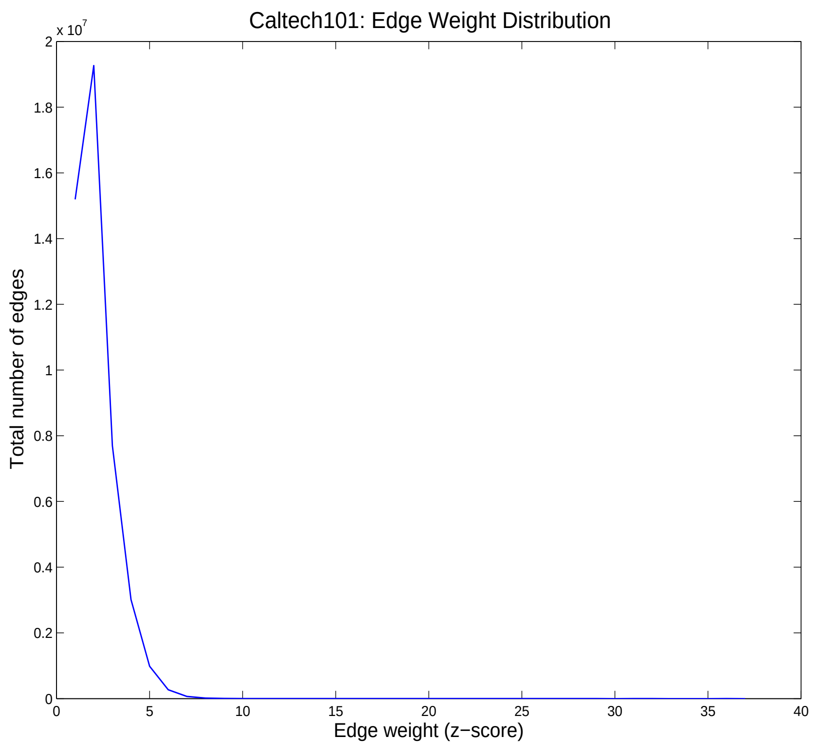


Figure 5.7: Edges weight distributions for both dataset for 10,000 visual words. The maximum z-scores of 68 and 35 are computed for Toymix and Caltech-101 datasets respectively.

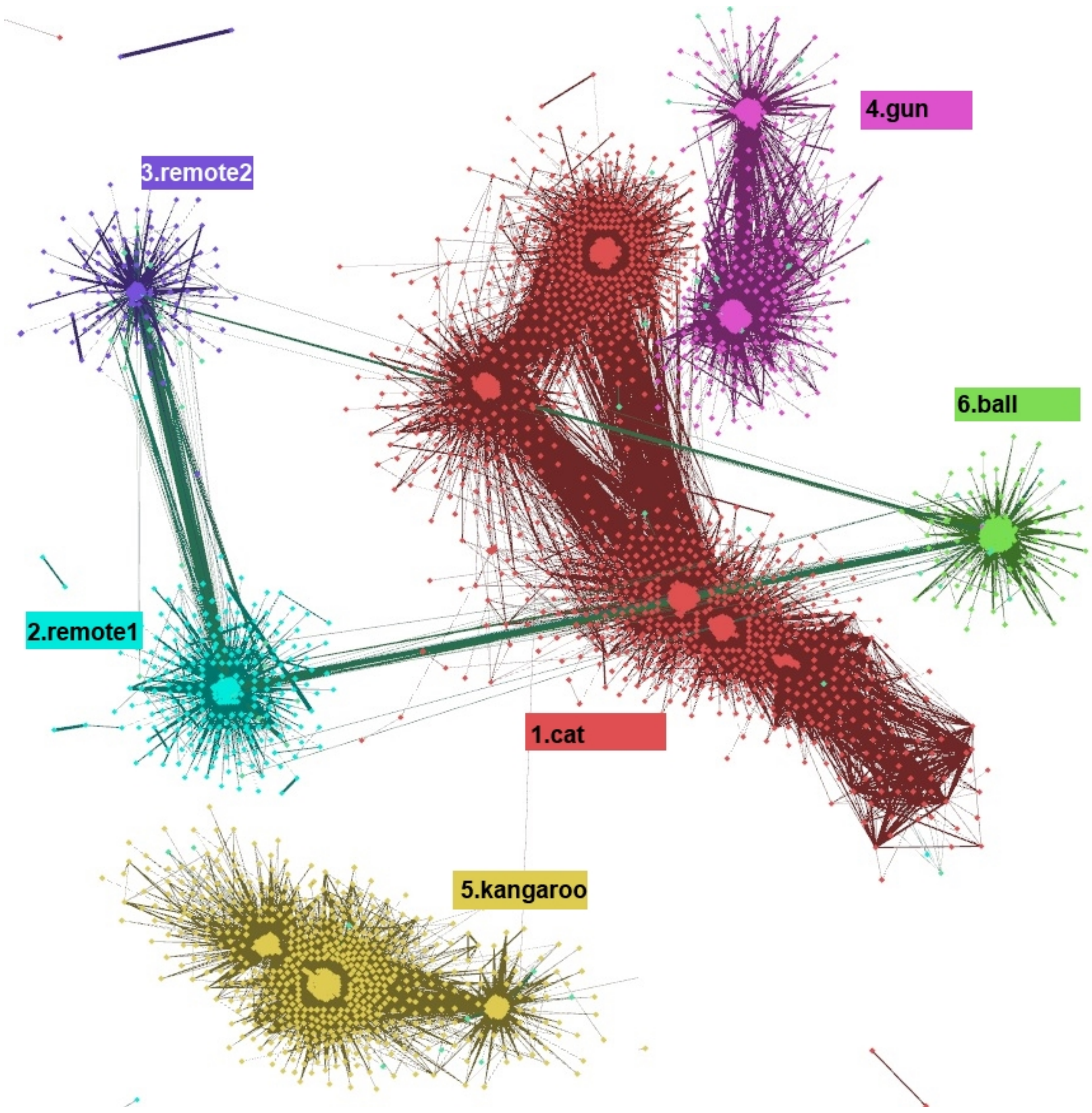


Figure 5.8: Graph structure obtained before using normalized cuts for 6 object categories from Toymix dataset. This graph contains top 10,000 significant co-occurrences. The graph is automatically partitioned in these categories (shown in different colors). Each cluster contain the visual words (vertices) that appear majority of times in a it. The figure is best viewed in color.

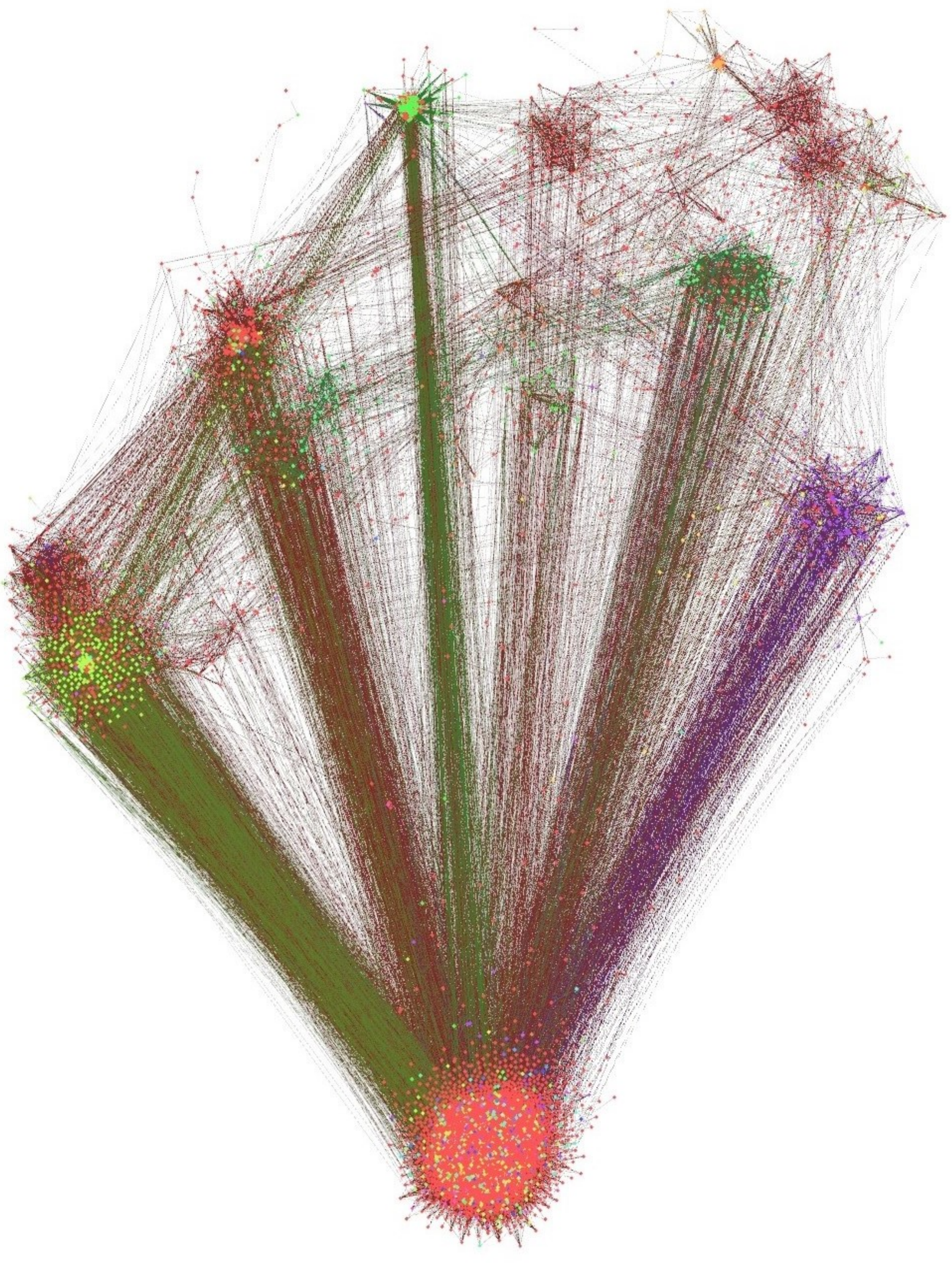


Figure 5.9: Graph structure obtained before using normalized cuts for 20 objects categories from Caltech-101 dataset. This graph contains top 10,000 significant co-occurrences. The graph show few dense clusters but none of the cluster has words (vertices) that mostly appear in a single object category. The figure is best viewed in color.

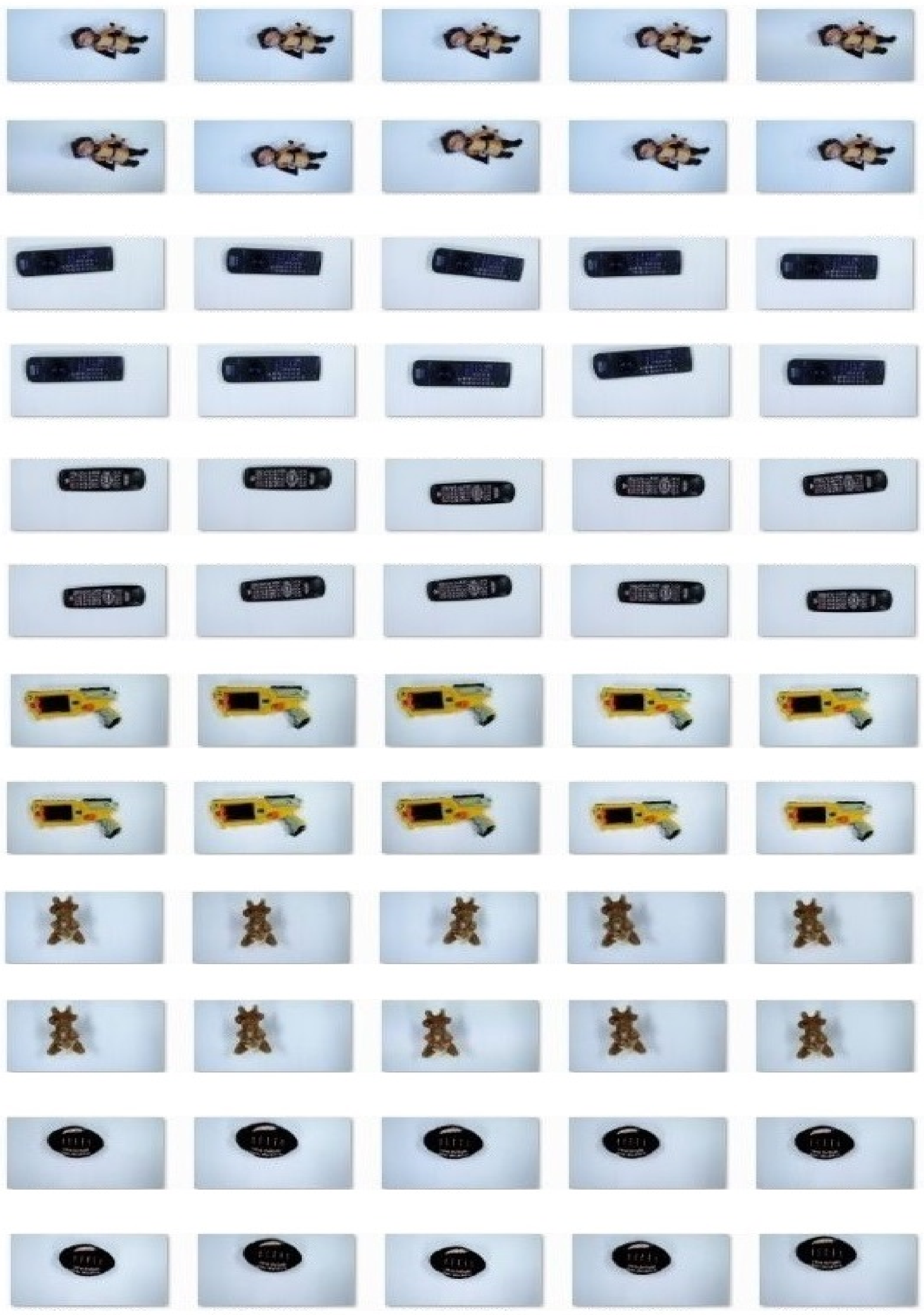


Figure 5.10: All 6 categories from Toymix dataset emerge after using the normalized cuts on the graph. Top 10 ranked images from each category are displayed.

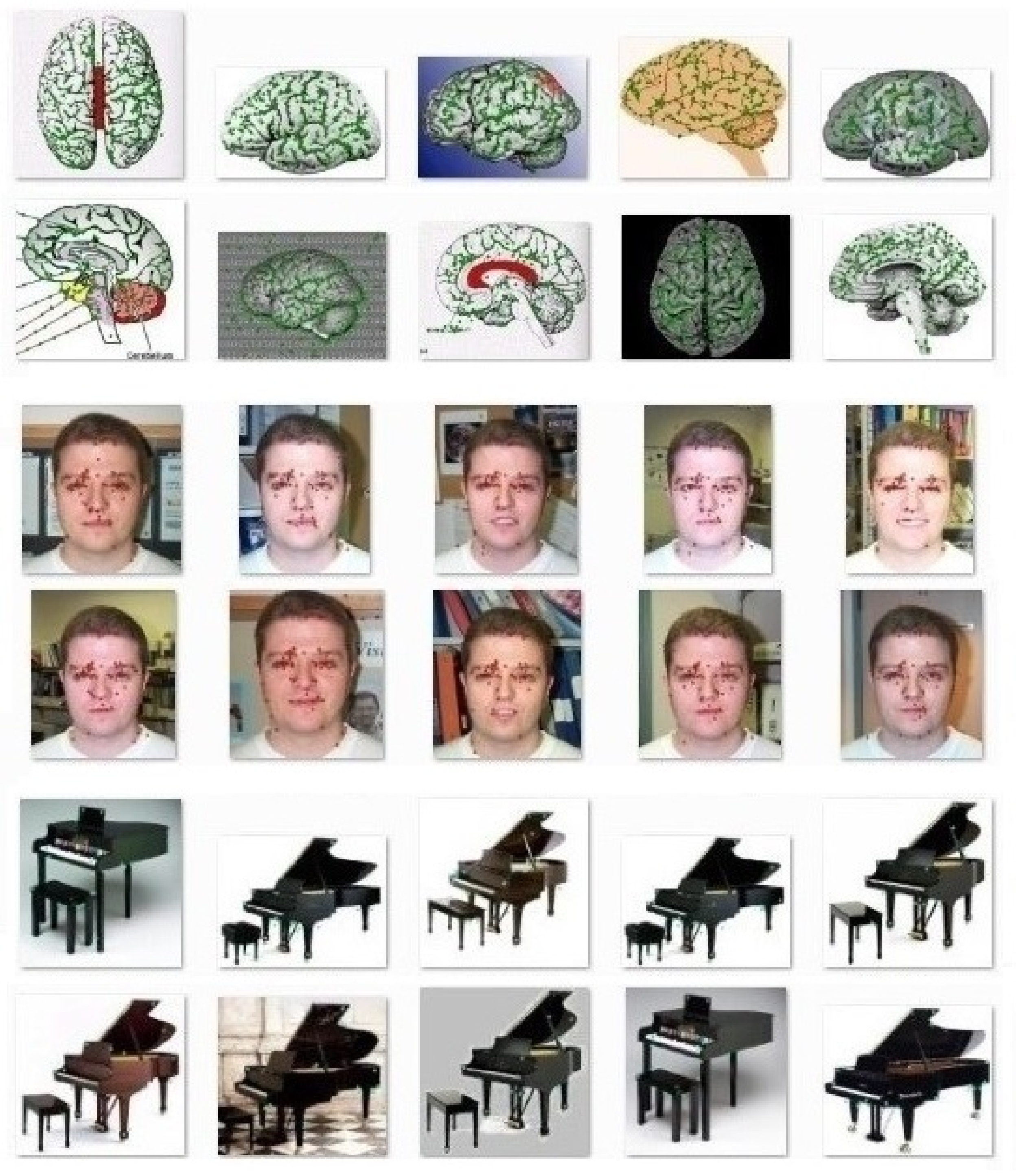


Figure 5.11: The first 3 emergent clusters out of 5 clusters that are related to object categories {*Brain, Faces_easy, Piano*} from the Caltech-101 dataset. These clusters emerge after using the normalized cuts on the co-occurrence graph. The top 10 ranked images from each category are displayed. The figure is best viewed in color.

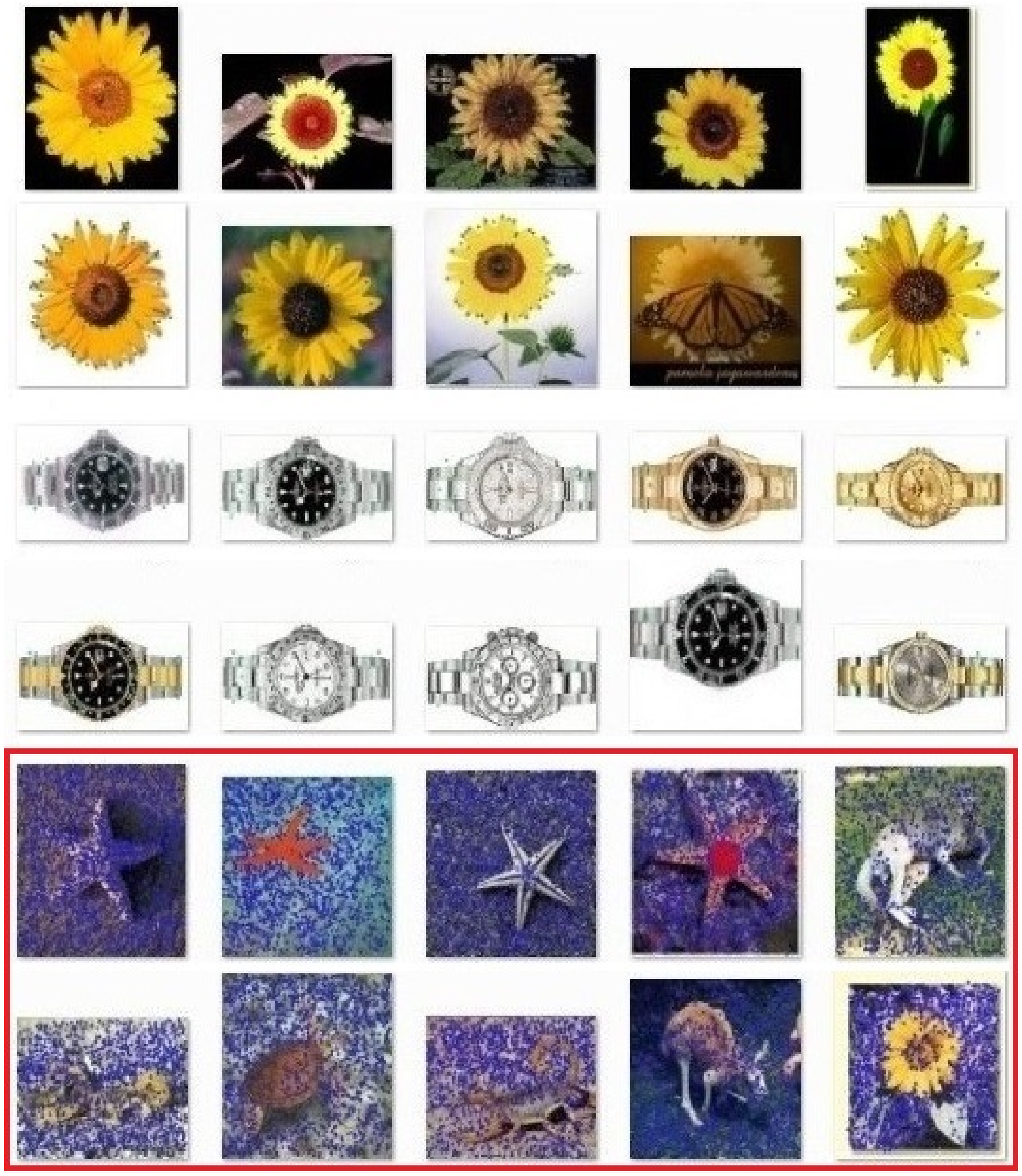


Figure 5.12: The last 2 emergent clusters out of 5 clusters that are related to object categories {*Sunflower, Watch*} from the Caltech-101 dataset. The bottom two rows shows an emergent cluster which is not related to any object category from the dataset. We think this cluster represent background or highly textured areas of images, hence we named it as {*Background*} cluster. In this non-object category images are put into red rectangle to separate them from the category ones. Top 10 ranked images from each cluster are displayed. The figure is best viewed in color.

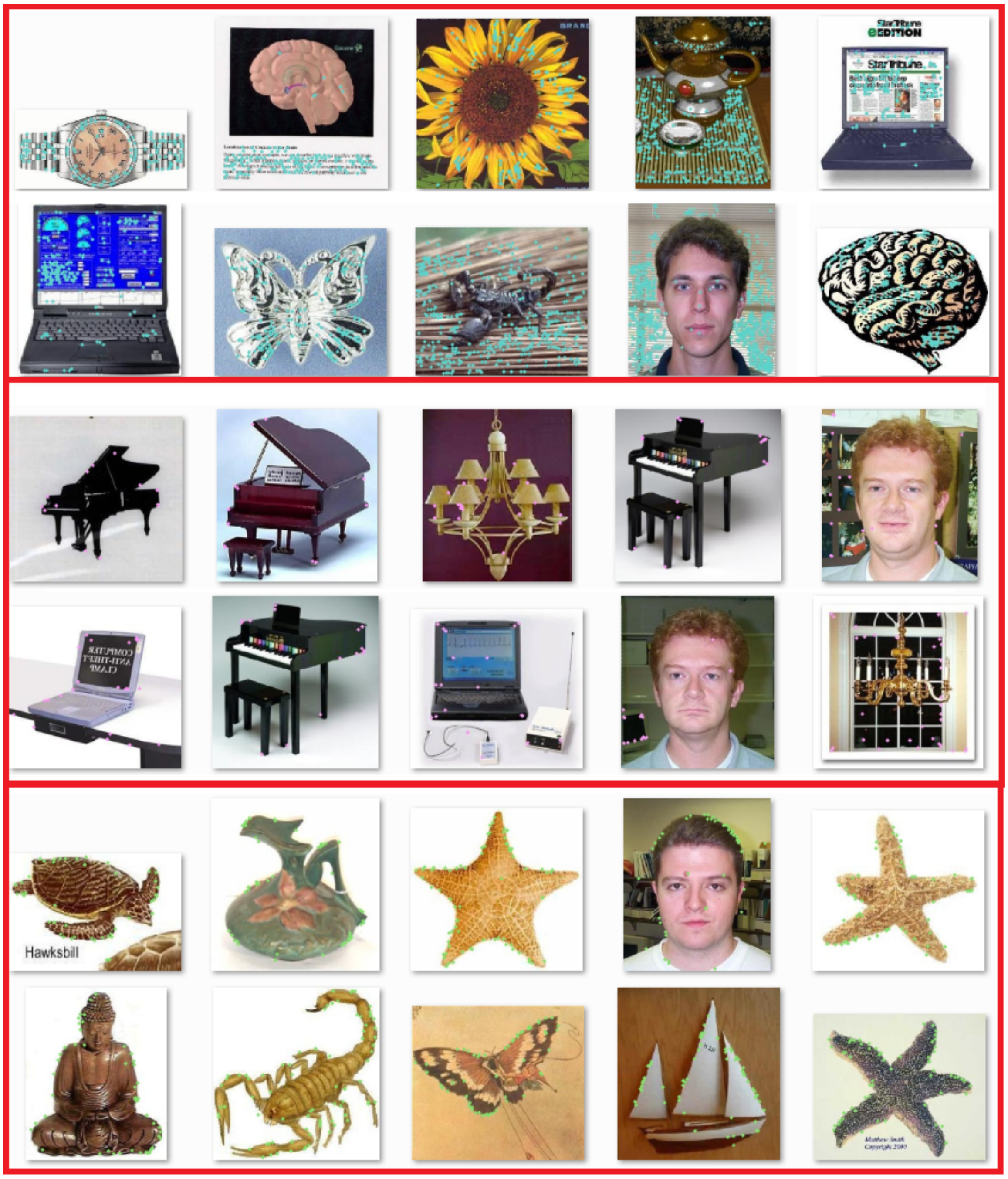


Figure 5.13: The 3 other emergent clusters that are not related to a single object category from Caltech-101 dataset. As features in majority of these images lie in the background we also assigned these as $\{Background\}$ clusters.

Chapter 6

Bag of Co-occurring Words (BoCoW)

Chapter 5 described an approach that encodes visual word co-occurrences in a weighted undirected graph and applies statistical analysis to its edges to identify emergent edges or co-occurrences. The statistical analysis technique allows us to choose co-occurrences which appeared more than their chance of occurrence and are a good candidate while looking for emergent behavior. These co-occurrences are grouped using a clustering technique which results in emergent patterns. We found that these emergent patterns can represent objects and many other interesting patterns.

In this chapter, we explore the behavior of emergent co-occurrences in an image retrieval scenario. The main motivation for this analysis is our observation from Chapter 5 that emergent patterns can also identify objects that are found in many images. Usually, in an image retrieval application, there are many images related to each scene or object of interest. We aim that by using emergent co-occurrences we would be able to capture information related to different objects in an image retrieval dataset. Bag-of-words (BoW) histogram is a common approach while performing image retrieval task which encodes visual words found in an image as their distribution. Our suggested approach called Bag-of-Co-occurring-Words (BoCoW) encodes emergent co-occurring words (CoW) found in an image as their distribution.

To measure the efficacy of our approach we conduct experiments on Oxford landmarks (Philbin *et al.*, 2007) and Paris buildings (Philbin *et al.*, 2008a) datasets which contain multiple images related to different scenes. We also conduct experiments to compare the discriminative power of BoCoW and BoW histograms. In an another analysis, we try to find out whether some co-occurrences are more important than others, by dividing co-occurrences into three categories according to their frequencies in the dataset. We also propose a novel approach that merges co-occurring words to an existing BoW histogram.

6.1 Introduction

The visual bag-of-words (BoW) model (Sivic and Zisserman, 2003) originally inspired from text documents relies on obtaining visual words by clustering a set of local features (for example SIFT (Lowe, 2004), and SURF (Bay *et al.*, 2006)), extracted from images in a dataset. Approximate K-means clustering using a kd-tree is a widely used approach (Lowe, 2004; Philbin *et al.*, 2007). Many modern image retrieval (Jégou *et al.*, 2008; Jegou *et al.*, 2007; Nister and Stewenius, 2006; Yang *et al.*, 2007), image classification (Csurka *et al.*, 2004; Nowak *et al.*, 2006), and object recognition systems (Chum *et al.*, 2007; Duygulu *et al.*, 2002; Lazebnik and Raginsky, 2009; Philbin *et al.*, 2008b) rely on the visual bag-of-words (BoW) model (Sivic and Zisserman, 2003) for accurate image matching. In image retrieval, many research efforts (Lowe, 2004; Nister and Stewenius, 2006; Philbin *et al.*, 2007, 2008b) focus on searching for duplicate, or similar, images in the dataset. Similar images often contain the same scenes, but the images are transformed, or have different viewpoints and occlusions.

Utilizing information from co-occurring visual words for an image retrieval application is not new. Morioka and Satoh (2010) for example, discovered visual words that are found in a confined neighborhood and encoded this spatial information in the BoW model to improve its performance. Traditional approaches first cluster the feature space and then based on spatial closeness, define pairs of co-occurring visual words. Their work is different because they suggest creating pairs of features (rather than visual words) based on their spatial closeness and then representing these pairs in a joint feature space. These pairs are used for clustering and calculating a Local Pairwise Codebook (LPC). Performance is evaluated on five datasets (two scene categorization tasks and three object categorizations). The approach gave competitive results to the state of the art on all these datasets.

Zhang *et al.* (2011) presented a technique called geometric preserving visual phrases (GVP). It encodes the neighborhood of a word as a visual phrase (a set of neighboring words). Performance is measured on two datasets, and results are compared with bag-of-words followed by RANSAC (Fischler and Bolles, 1981) based verification steps. Their approach outperformed the Bag-of-Words based method (Philbin *et al.*, 2007) and needed less memory and computation time. In Flickr 1M dataset, GVP outperforms the BoW by gaining 12% higher MAP, while, on Oxford dataset, the maximum MAP of 0.696 was achieved as compared to 0.634 in the BoW case.

Convolution Neural Network (CNN) is an approach that aims at building high level abstractions of image pixels in a buttom-up fashion. Some of these layers are convolution, nonlinear, pooling, fully connected, and an output layer. CNN are similar to our method in a way that both aim at building higher level representation in a bottom up manner. CNNs has the advantage that the learning process is carried out in hierarchical fashion and many level of abstractions are created which depends on the number of hidden layers. Our approach on the other hand only abstracts a single level information. CNNs has been applied in both supervised and unsupervised enviroments.

In the supervised learning case, the first breakthrough in using CNN for image classification task was made by Krizhevsky *et al.* (2012). They achieved a top-5 test error rate of 15.3% which was a staggering 10.8% lower than the second best in 2012 ILSVRC (ImageNet Large-Scale Visual Recognition Challenge). The architecture that they presented (which was called AlexNet) contains 650,000 neurons was made up of 5 convolution layers, max-pooling layers, dropout layers, and 3 fully connected layers. AlexNet was capable of classifying 1000 possible categories.

In ILSVRC 2014, Simonyan and Zisserman (2014) presented a 19 layers CNN architecture (which was called VGG Net) using 3x3 filters with stride and pad of 1 and 2x2 maxpooling layers with stride 2. They achieved a very low top-5 error rate of 7.3% and also reduced the number of required parameters. Their main contribution was to reinforce the notion that in order to get good performance the convolution neural network must have a deep network of layers.

In another paper by Microsoft Research Asia, He *et al.* (2016) presented 152 layered deep network (which was called ResNet) and won the ILSVRC 2015 with an incredible error rate of 3.6%. They presented the idea of a residual block which in which an input goes through a series of conv-relu-conv layers.

In a paper by Wang and Gupta (2015) presented an approach to learn CNN in unsupervised fashion from unlabelled videos. In order to learn meaningful image representations similar image patches across the video are tracked. These provide some kind of supervision and help identifying important pathes in the video. The learning process did not use a single labeled image from ImageNet, they achieved a mAP of 52%. This is surprsingle very close to 54.4% which is achieved on ImageNet.

The rest of the chapter is structured as follows. Section 6.2, explains the proposed approach by discussing the different stages of the process. Experiments and results are detailed in Sections 6.3 and 6.4. Detail evaluation of Bag-of-Co-occurring-Words (BoCoW) is performed in Sections 6.5 and 6.6. Section 6.7 proposes a method that combines Bag-of-Words (BoW) descriptor with emergent information. Finally, Section 6.8 summarizes the chapter.

6.2 Proposed Approach

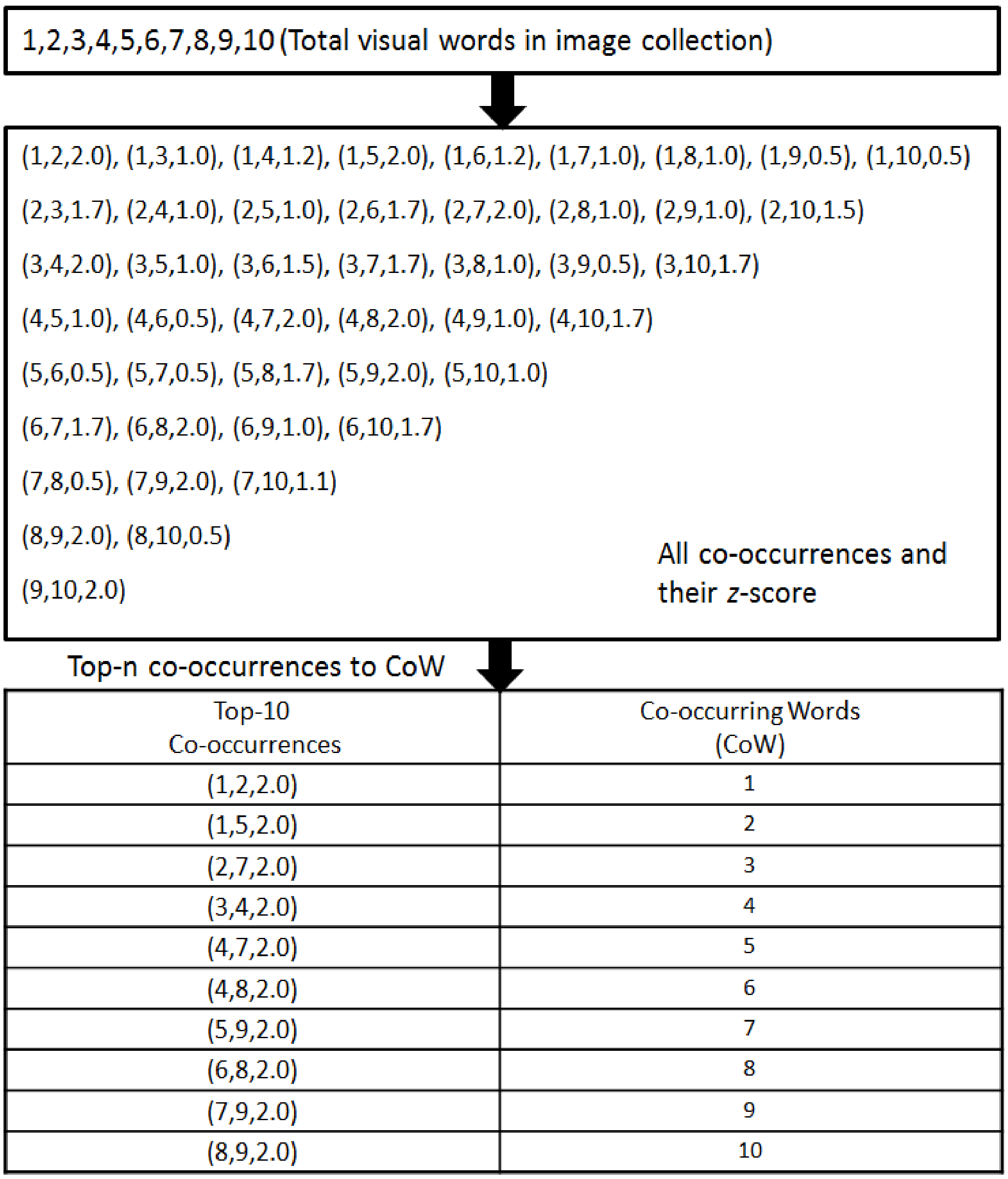
The proposed approach builds on the emerging cluster discovery method described in the Chapter 5. We start by performing the statistical analysis on all the co-occurrences obtained from an image collection using the method described earlier. All the co-occurrences are sorted by their z-scores and only top-n co-occurrences with the highest z-scores are kept. These co-occurrences are highly significant as they appeared more than their chance. The top-n co-occurrences are mapped to unique numbers called Co-occurring-Words (CoW). Figure 6.1 depicts this process using an example with 10 visual words in a collection.

Next step is to represent each image using this top-n (n=10 in the example) co-occurring-words. All visual words in an image are obtained and then all possible co-occurrence are formed using them. Out of these, only co-occurrences which are in the top-n set are kept. These co-occurrences are represented in a n-dimensional vector called Bag-of-Co-occurring-Words (BoCoW) as shown in Figure 6.2. In this vector, a value of 1 or 0 describes whether the image contains a co-occurrence from the top-n set or not. In the Figure 6.2 the BoCoW for an image is shown. The BoCoW depicts that the co-occurring word number 7, 9, and 10 are absent from this image. A BoCoW vector is created for every image in the collection.

Now the goal is to find out which images are similar to each other. A naive way is to count the number co-occurrences that are found in both images. Two images that share a large number of co-occurrences are considered closer to each other. As each co-occurrence has a different significance score than others (because of different z-scores), simply counting their presence in both images do not justify their importance and results in the poor matching criterion. Hence, a weight for each co-occurrence from the top-n set is computed using the log-likelihood value, L_k as shown in the equation.

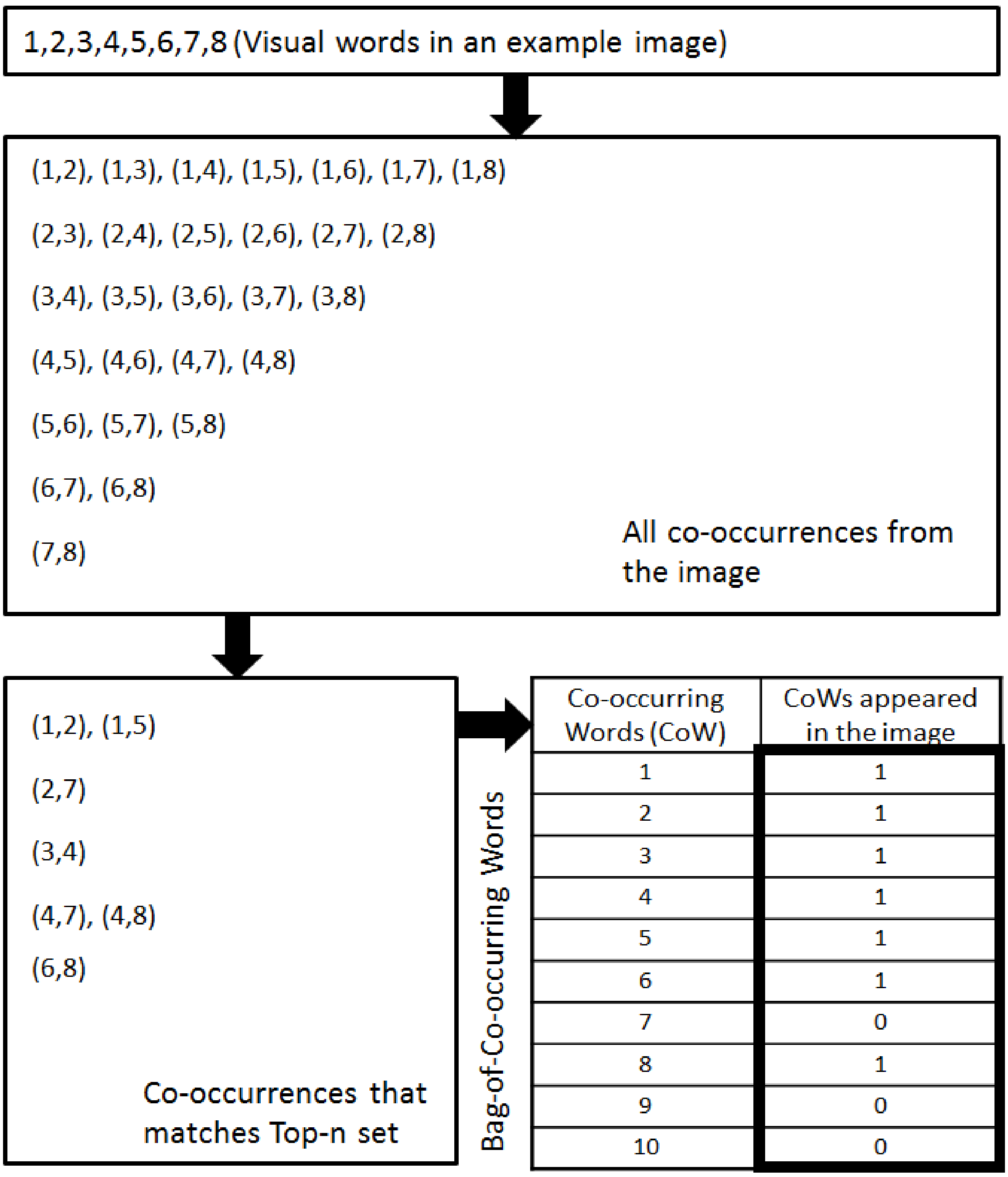
$$L_k = |\log\left(\frac{F_k}{N}\right)|, \tag{6.1}$$

where N is the total number of images, and F_k, is the number of images containing a co-occurrence k, where $k = 1, 2, 3, ..., n$. n is the total number of most important co-occurrences kept. As all the co-occurrences in the top-n set appeared at least once in the collection, it is not possible to have a case when $F_k == 0$.



Top-10 Co-occurrences	Co-occurring Words (CoW)
(1,2,2.0)	1
(1,5,2.0)	2
(2,7,2.0)	3
(3,4,2.0)	4
(4,7,2.0)	5
(4,8,2.0)	6
(5,9,2.0)	7
(6,8,2.0)	8
(7,9,2.0)	9
(8,9,2.0)	10

Figure 6.1: The process of obtaining CoW from top-n co-occurrences. The bottom table shows the top-10 co-occurrences with the highest z-score.



Co-occurring Words (CoW)	CoWs appeared in the image
1	1
2	1
3	1
4	1
5	1
6	1
7	0
8	1
9	0
10	0

Figure 6.2: The process of obtaining BoCoW for an example image. The final BoCoW vector is shown in the thick rectangle.

6.2.1 Matching Score Calculation for BoCoW

The performance of BoCoW is measured in an image retrieval scenario. Each query image is matched with all dataset images. Rather than using the naive approach to compute the similarity, we found that adding their log-likelihood values for each shared co-occurrences works better. The similarity score, $Sim(q, d)$ between BoCoWs of a query image q and a database image d is computed as follows:

$$Sim(q,d) = \sum_{k=1}^{n} \begin{cases} L_k, \text{if } (q_k = 1 \wedge d_k = 1) \\ 0, \text{otherwise.} \end{cases} \tag{6.2}$$

Here q and d are Bag-of-co-occurring words (BoCoW) of two images. L_k is the log likelihood values for all the top-n co-occurrences. In the equation, the log-likelihood value is only added if both compared images contain a co-occurring word.

6.2.2 Matching Score Calculation for BoW

The retrieval results using BoCoW are compared with the BoW based technique (Philbin *et al.*, 2007). The Euclidean distance, $Dist(Q,D)$ between the BoWs of a query image Q and a database image D is computed as:

$$Dist(Q,D) = \sqrt{\sum_{i=1}^{K}(Q_i - D_i)^2}, \tag{6.3}$$

where $Dist(Q,D)$ is the distance between the BoW representations of a query image and a dataset image. Here K is the number of dimensions of the BoW histogram i.e., the total number of cluster centres. To make both BoCoW and BoW based approaches comparable, the value of K (number of cluster centres) and n (number of top co-occurrences) are set the same.

6.2.3 Performance Metrics

To evaluate the performance of BoCoW, we use the mean average precision (MAP).

Precision and Recall

Precision in information retrieval is defined as the fraction of the retrieved documents (images) that are relevant to the query. In this case, the relevant documents are those dataset documents that are similar to the query. Precision, P, can be defined as:

$$P = \frac{|\{(\text{relevant documents}) \cap (\text{retrieved documents})\}|}{|\{\text{retrieved documents}\}|}. \tag{6.4}$$

Instead of taking into account all the retrieved document against a query. Precision at r or $P(r)$, is calculated at a particular rank, thus it only considers top-r results.

Recall, R, is the fraction of the documents relevant to the query that are retrieved. In simple words, we want to measure, how many of the relevant document are recovered, out of the total number of relevant documents for a query.

$$R = \frac{|\{(\text{relevant documents}) \cap (\text{retrieved documents})\}|}{|\{\text{relevant documents}\}|}. \tag{6.5}$$

If precision and recall is calculated at every rank, this can be used to plot a precision-recall curve. In this plot each precision P(R) value is plotted as a function of recall R.

Average Precision (AP), and Mean Average Precision (MAP)

For retrieval systems that return ranked results, the order of the results is crucial. For example, if a retrieval system returns top 10 results, out of which only 5 are relevant, then a good retrieval system will have all relevant documents in the top-5 ranks. This contrasts with precision and recall, which provide metrics based on a single rank value.

Average precision (AP) computes the precision at ranks where we have a relevant document for each query and divides it by the total number of relevant documents:

$$AP = \frac{\sum_{r=1}^{m} P(r) \times rel(r)}{|\{\text{relevant documents}\}|} \tag{6.6}$$

where, r is the rank, m represents the number of retrieved documents, precision at r is represented as $P(r)$, and $rel(r)$ is an indicator function equaling 1 if the item at rank r is a relevant document, zero otherwise. Average precision is the average over all relevant documents, and all the relevant documents that are not retrieved get a precision score of zero.

Mean average precision, MAP, on the other hand, is the mean of the average precisions for all queries.

$$\text{MAP} = \frac{\sum_{q=1}^{M} AP(q)}{M}, \tag{6.7}$$

where M is the number of total queries.

6.3 Experimental Details

The image retrieval experiment is conducted on two image datasets: Oxford landmarks (Philbin *et al.*, 2007) and Paris buildings (Philbin *et al.*, 2008a). The Oxford dataset contains 5062 images from 11 landmark locations in Oxford. For each landmark, there are 5 query images, so there are 55 queries in total. The Paris buildings dataset contains 6412 images that are also divided into 11 landmark places and for each landmark there are 5 query images. This dataset therefore has 55 queries in total. Figure 6.3 and 6.4, shows some images from each dataset. The entire dataset consists of high resolution (1024 x 768) images.

All the images in both the datasets are categorized into Good, Ok, Junk, or Absent category with respect to each query image as shown in Table 6.1. For both datasets, Absent and Junk category images are considered non-relevant. To compute the performance of an image retrieval method only Good and Ok images are considered positive images.

Table 6.1: Different categorization of images in Oxford and Paris datasets

Image groups	% of object visible
Good	at-least 50
Ok	at-least 25
Junk	less than 25
Absent	doesn't contain object

SIFT (Lowe, 2004) feature descriptors are obtained from both dataset images. On average, approximately 3000 features are extracted from each image. To evaluate the performance of different co-occurrences these are clustered into various numbers of visual words K. We experimented with 10,000, 50,000 and 100,000 visual words as these are used in (Philbin *et al.*, 2007, 2008a). Images in the datasets are represented as Bag-of-words histograms.

For creating BoCoW, the top-n co-occurrences are chosen. We experimented with 5 different values of top-n co-occurrences, which are 10,000 (10K), 50,000 (50K), 100,000 (100K), 500,000 (500K) and 1,000,000 (1M) co-occurrences. A query image is matched with all the dataset images and similarity scores are obtained for each query image as mentioned in the Section 6.2.1. The retrieved images are ranked and ordered according to the similarity score. A mean average precision (MAP) score is calculated for each method.

Figure 6.3: Few images from Oxford dataset.

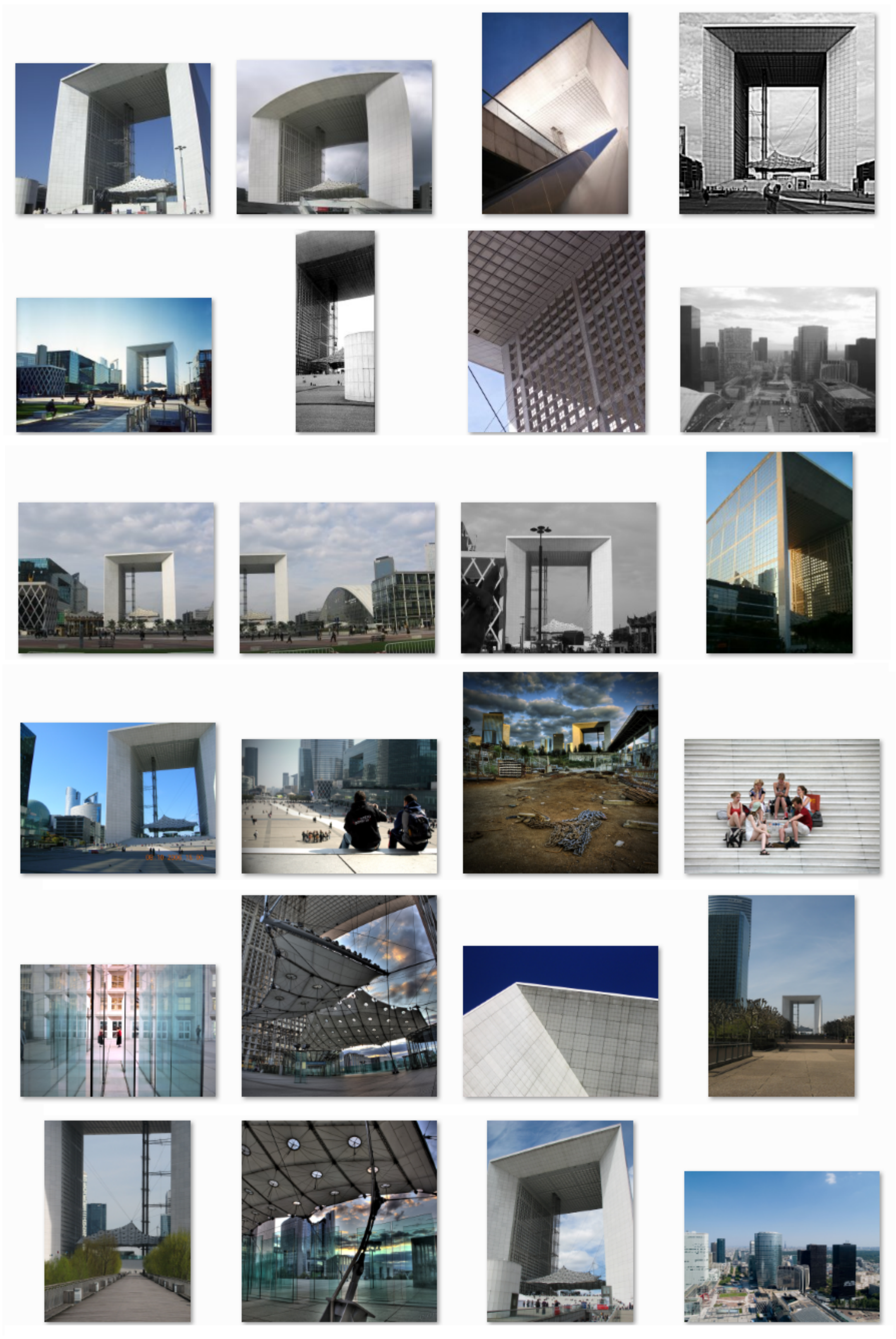


Figure 6.4: Some images from Paris datasets

Table 6.2: Number of positive images for each landmark in Oxford and Paris dataset. Images from Good and OK category are considered positive. Since in our experiments we are using emergent co-occurrences, it is highly likely that these co-occurrences will better represent landmarks that have more positive images.

Landmark name	positive images	Landmark name	positive images
radcliffe_camera	221	eiffel	289
all_souls	78	triomphe	281
christ_church	78	moulinrouge	237
magdalen	54	invalides	198
hertford	54	louvre	152
ashmolean	25	sacrecoeur	149
bodleian	24	pantheon	126
balliol	12	notredame	119
cornmarket	9	defense	117
keble	7	museedorsay	72
pitt_rivers	6	pompidou	51

6.4 Image Retrieval Results

6.4.1 Oxford Building Dataset - Results

Table 6.3 depicts MAP for different methods. In all cases increasing number of visual words results in increasing mean average precision (MAP). In the BoW case a maximum MAP of 0.398 is obtained when 100,000 visual words are used. This is slightly low as compared to the results shown in (Philbin *et al.*, 2007) in which they achieved a MAP of 0.535 using 100,000 visual words without any spatial matching. This is due to the way they query an image. In their approach they select a region of interest from the image containing the object. This limits visual words in the query to be the most accurate and refined description of object and certainly helps in retrieval. We, on the other hand are looking at emergent co-occurrences and hence use all the visual words in an image to generate Bag-of-co-occurring-words as shown in Figure 6.2.

For BoCoW cases, a maximum MAP of 0.31 is achieved when 500,000 CoWs are used which are extracted from 100,000 visual words. Based on the two data points we hypothesize that as the number of visual words and co-occurring words increases, the

Table 6.3: MAP for different settings of visual words and co-occurrence words in Oxford dataset. For each number of visual words multiple values of CoW are chosen to represent BoCoW.

Method	MAP
BoW [10K Words]	0.30182
BoW [50K Words]	0.34484
BoW [100K Words]	***0.39783***
BoCoW [10K Words - 10K CoW]	**0.11885**
BoCoW [10K Words - 50K CoW]	0.10074
BoCoW [10K Words - 100K CoW]	0.11687
BoCoW [10K Words - 500K CoW]	0.11354
BoCoW [10K Words - 1M CoW]	0.09874
BoCoW [50K Words - 10K CoW]	0.22522
BoCoW [50K Words - 50K CoW]	**0.24805**
BoCoW [50K Words - 100K CoW]	0.24532
BoCoW [50K Words - 500K CoW]	0.23169
BoCoW [50K Words - 1M CoW]	0.22688
BoCoW [100K Words - 10K CoW]	0.21011
BoCoW [100K Words - 50K CoW]	0.26647
BoCoW [100K Words - 100K CoW]	0.30033
BoCoW [100K Words - 500K CoW]	**0.31011**
BoCoW [100K Words - 1M CoW]	0.30481

MAP increases. In all cases the performance of BoW always performed better than BoCoW.

As we have seen that increasing visual words results in higher MAP. It is possible that MAP obtained by BoCoW is simply random and encoding co-occurrences into BoCoW does not really help. To check this we randomly chose n co-occurrences from list of all co-occurrences and built BoCoW. Results are shown in Table 6.4. We can see that random co-occurrences score is much lower which results in poor image retrieval performance. There is slight increase in MAP as we add more visual words and a maximum MAP of 0.136 is achieved. We think the main contributing factor causing this increase is the number of visual words and not the co-occurring words.

Table 6.4: MAP for different settings of visual words and co-occurrence words in Oxford dataset. For each number of visual words different numbers of CoW are randomly chosen to represent BoCoW.

Method	MAP
BoCoW [10K Words - 10K Random CoW]	0.07517
BoCoW [10K Words - 50K Random CoW]	0.08712
BoCoW [10K Words - 100K Random CoW]	0.07916
BoCoW [10K Words - 500K Random CoW]	0.08924
BoCoW [10K Words - 1M Random CoW]	**0.09149**
BoCoW [50K Words - 10K Random CoW]	0.11977
BoCoW [50K Words - 50K Random CoW]	**0.12163**
BoCoW [50K Words - 100K Random CoW]	0.11274
BoCoW [50K Words - 500K Random CoW]	0.10941
BoCoW [50K Words - 1M Random CoW]	0.11939
BoCoW [100K Words - 10K Random CoW]	0.12675
BoCoW [100K Words - 50K Random CoW]	0.13188
BoCoW [100K Words - 100K Random CoW]	0.12362
BoCoW [100K Words - 500K Random CoW]	0.12951
BoCoW [100K Words - 1M Random CoW]	***0.13676***

6.4.2 Paris Dataset - Results

The experiments on the Paris dataset are performed in a similar way as for Oxford dataset, and here again, the BoW approach performs much better than BoCoW cases. Results are depicted in Table 6.5. For BoW case maximum MAP of 0.371 is achieved using 100,000 visual words. For BoCoW case a highest MAP of 0.3602 is obtained when 500,000 co-occurrences and 100,000 visual words are used. Unlike Oxford dataset, the image retrieval performance for both techniques (BoW and BoCoW) on Paris dataset is very close.

Here again, based on the two data points we hypothesize that increasing visual words and co-occurring words do increases MAP. Now to check whether retrieval results for the BoCoW approach are better than random. We randomly choose n co-occurrences from the list of all co-occurrences and built BoCoW. Results are shown in Table 6.6. A slight increase in MAP is noted as we increase visual words and a maximum MAP of 0.1261 is achieved.

Table 6.5: MAP for different settings of visual words and co-occurrence words in Paris dataset. For each number of visual words multiple values of CoW are chosen to represent BoCoW.

Method	MAP
BoW [10K Words]	0.27115
BoW [50K Words]	0.32183
BoW [100K Words]	***0.37126***
BoCoW [10K Words - 10K CoW]	**0.15222**
BoCoW [10K Words - 50K CoW]	0.10748
BoCoW [10K Words - 100K CoW]	0.09737
BoCoW [10K Words - 500K CoW]	0.10466
BoCoW [10K Words - 1M CoW]	0.10594
BoCoW [50K Words - 10K CoW]	0.20391
BoCoW [50K Words - 50K CoW]	**0.23213**
BoCoW [50K Words - 100K CoW]	0.23107
BoCoW [50K Words - 500K CoW]	0.22412
BoCoW [50K Words - 1M CoW]	0.22154
BoCoW [100K Words - 10K CoW]	0.21516
BoCoW [100K Words - 50K CoW]	0.27147
BoCoW [100K Words - 100K CoW]	0.32342
BoCoW [100K Words - 500K CoW]	**0.36029**
BoCoW [100K Words - 1M CoW]	0.35314

6.4.3 Discussion - Image Retrieval Results

By looking at the results it is clear that overall but specially for the Oxford dataset the BoCoW approach performed below par when compared to BoW even when the most significant co-occurrences are used. A slight increase in the performance is achieved at the expense of a larger BoCoW.

Also in our experiments, the MAP achieved using BoW method is comparatively lower than (Philbin *et al.*, 2007, 2008a). The main reason is that for a query image they only selected a portion of an image (containing an object) restricting visual words which are the most refined information of an object. In our experiments as we are looking at emergent behavior and hence want to use all the visual words in an image. This results in adding extra visual words in a query image and is analogous to adding noise which ultimately reduces MAP.

For BoCoW, the maximum MAP of 0.3101 and 0.3602 is achieved for Oxford and

Paris dataset, as compared to MAP of 0.398 and 0.3712 in the BoW case. In most cases, the BoCoW approach did not perform well as the BoW method, but in a few cases, BoCoW performance was comparative to BoW. By looking at the numbers in table 6.3 and 6.5 we hypothesize that increasing the number of visual words and co-occurring words increases MAP.

Since this is the first time the two methods (BoW and BoCoW) are compared in an image retrieval scenario it is important to compare different characteristics of both representations. One such characteristic could be finding out the way information from images is represented. Is it discriminant enough? We assume that a good image retrieval approach represents word information in a way that visual word distribution related to each class remains very different from other classes. This is very important for good image retrieval. We are interested in finding out whether this is the case for the representations we are comparing in this chapter or not? Section 6.5 shows experiments to answer this question.

We are also interested in finding out if there are any subsets of co-occurrences among the most significant ones that are more important than others and store more distinctive information? We answer all these questions in section 6.6.

6.5 Words Distribution Analysis

In this section, we want to find out how two methods (BoW and BoCoW) represents words extracted from images for a image retrieval scenario. We want to find out whether these methods encode visual words in a way that results in a larger difference between an empirical distribution built over visual words from a single class versus another distribution built from images that do not contain this class. Usually, a technique that has a large difference between the two kinds of distributions will result in higher image retrieval performance.

Kullback-Leibler divergence (KLD) is a measure of difference (or divergence) between two probability distributions P and Q. The KLD from P to Q on a finite set I is defined as:

$$D_{KL}(P||Q) = \sum_{i \in I} P(i) log \frac{P(i)}{Q(i)}, \tag{6.8}$$

$D_{KL}(P||Q)$ (divergence) is the amount of information lost when Q is used to approximate P. i Here i is the dimension of the probability distributions. Sometimes

Table 6.6: MAP for different settings of visual words and co-occurrence words in Paris dataset. For each number of visual words different numbers of CoW are randomly chosen to represent BoCoW.

Method	MAP
BoCoW [10K Words - 10K Random CoW]	0.07127
BoCoW [10K Words - 50K Random CoW]	0.07833
BoCoW [10K Words - 100K Random CoW]	0.06857
BoCoW [10K Words - 500K Random CoW]	0.07836
BoCoW [10K Words - 1M Random CoW]	**0.09778**
BoCoW [50K Words - 10K Random CoW]	0.10975
BoCoW [50K Words - 50K Random CoW]	**0.11279**
BoCoW [50K Words - 100K Random CoW]	0.10993
BoCoW [50K Words - 500K Random CoW]	0.11022
BoCoW [50K Words - 1M Random CoW]	0.11236
BoCoW [100K Words - 10K Random CoW]	0.11522
BoCoW [100K Words - 50K Random CoW]	0.12375
BoCoW [100K Words - 100K Random CoW]	***0.12617***
BoCoW [100K Words - 500K Random CoW]	0.12192
BoCoW [100K Words - 1M Random CoW]	0.11979

$D_{KL}(P||Q)$ is used to measure the distance between probability distributions. But KLD is not a true distance metric.

$$P(C_i|W_j) = \frac{P(C_i, W_j)}{P(W_t)} \tag{6.9}$$

In our case, P could be a probability distribution of a single class and Q could be a probability distribution related to all other images (i.e., images that do not contain any class object). We are interested in finding out for which method (BoW or BoCoW) the difference between P and Q is large. The higher the difference the better the technique is able to identify different classes and hence results in better image retrieval performance. The experiments are conducted on both Oxford and Paris datasets. Algorithm 3 lists the approach for computing $D_{KL}(P||Q)$ between single class probability distributions P and all other images distribution Q.

Algorithm 3: Computing KL-divergence

```
Data: Image words and ground truth information,
Result: KLD between a class distribution $P$ and all other images distribution
        $Q$
1 for each class in a image dataset do
2     Get all positive images set using ground truth information,
3     Build a histogram using words in images
4     Compute probability distribution for single class histogram $P(C_j|W)$
5     Get all other images that do not contain positive images
6     Build a histogram for all other images
7     Compute probability distributions for all other image histograms $Q$
8 for each single class probability distribution do
9     Compute divergence,$D_{KL}(P||Q)$
```

6.5.1 Results and Discussion

Table 6.7 and 6.8 depicts the results for Kullback-Leibler divergence (KLD) between single class histograms and all other image histograms. Results are also compared with image retrieval performance (using mean average precision) of different approaches. In both the Oxford and Paris datasets, BoCoW method usually haves higher KLD than the BoW method. The higher divergence for BoCoW means that the difference between

P and Q is much higher than the difference for BoW approach. Therefore, we would expect co-occurring words (CoW) to be better at image retrieval. But this is not the case as MAP achieved by BoCoW methods is lower than BoW methods. It seems like our assumption that a good image retrieval system results in higher KLD score between the two distributions does quite hold here. As the way information is extracted for both approaches is very different (fixed visual words versus emergent co-occurring words), it could be the case that they have totally different range of KLD scores. In such a case having such comparison would not give meaningful results.

Table 6.7: KLD for different settings of visual words and co-occurrence words in Oxford dataset. For each number of visual words multiple values of CoW are chosen to represent BoCoW.

Method	MAP	KLD	KLD-(within)
BoW [10K Words]	0.30182	0.36711	2.89313
BoW [50K Words]	0.34484	0.89465	12.4232
BoW [100K Words]	0.39783	1.37374	17.2212
BoCoW [10K Words - 10K CoW]	0.11885	0.88586	12.6194
BoCoW [10K Words - 50K CoW]	0.10074	0.77851	12.8835
BoCoW [10K Words - 100K CoW]	0.11687	0.75337	11.8096
BoCoW [10K Words - 500K CoW]	0.1135	0.74355	12.4688
BoCoW [10K Words - 1M CoW	0.09874	0.76337	11.7592
BoCoW [50K Words - 10K CoW]	0.22522	2.25931	15.0036
BoCoW [50K Words - 50K CoW]	0.24805	2.47253	17.4402
BoCoW [50K Words - 100K CoW]	0.24532	2.59764	18.7322
BoCoW [50K Words - 500K CoW]	0.23169	2.76910	21.4195
BoCoW [50K Words - 1M CoW]	0.22688	2.80644	22.4015
BoCoW [100K Words - 10K CoW]	0.21011	9.21658	16.5050
BoCoW [100K Words - 50K CoW]	0.26647	8.26008	16.7495
BoCoW [100K Words - 100K CoW]	0.30033	7.71742	17.3038
BoCoW [100K Words - 500K CoW]	0.31011	7.17836	20.3834
BoCoW [100K Words - 1M CoW]	0.30481	6.92414	21.5641

Another reason for higher KLD for BoCoW method could be that the within class divergence is also large i.e. few CoWs are common between images of the same class. To test this hypothesis further experiments are conducted. In these experiments two distributions for each class are created by randomly choosing co-occurring words (CoW) and KLD between these two distributions is computed as done before. Results for these experiments are depicted in column four of Tables 6.7 and 6.8. Here KLD-(within) represents divergence between two distributions of the same class.

Looking at KLD-(within) for both the Oxford and Paris datasets, our hypothesis about larger within class divergence seems to explain some of it. For example for Oxford dataset at least up to 50,000 words within class divergence is usually increasing. But this doesn't quite hold true for the results for 100,000 words, as KLD-(within) remained the same. In this case, KLD between the classes increased and this results in increasing retrieval performance. For Paris dataset though KLD-(within) is continuously increasing along with KLD.

Another important thing to note is that KLD-(within) always remained higher than KLD. This means that the divergence within class distribution is higher than the divergence between a class distribution and all other images distribution. In other words every class distribution is closer to all other image distribution than itself. To understand this behavior we conducted our next set of experiments on an example dataset where distributions of words in each class are known.

6.5.2 Further Analysis and Results

We want to find out whether is it possible that probability distribution of single class can be closer to a probability distribution generated from images all other classes than itself. In other words when KLD-(within) is higher than KLD between objects. The whole approach is as follows:

Generate two datasets (two classes) from known distributions, then calculate KLD-(within) and between KLD and see what that looks like. For example say we have just three words (this is an example, we used more than 3 - up to 10000), A, B, and C, and the classes C1, C2, C3, and R:

- C1: 60% A's, 20% B's, 20% C's
- C2: 20% A's, 60% B's, 20% C's
- C3: 20% A's, 20% B's, 60% C's
- R: Random selection of A's, B's and C's

Then a bunch of samples from C1, C2, and C3 are generated by using the distribution of words in each class and then single class probability distribution are computed for them. For computing KLD for a class (for example, C1) we also need probability

Table 6.8: KLD for different settings of visual words and co-occurrence words in Paris dataset. For each number of visual words multiple values of CoW are chosen to represent BoCoW.

Method	MAP	KLD	KLD-(within)
BoW [10K Words]	0.27116	0.20359	0.12198
BoW [50K Words]	0.32184	0.35573	1.48464
BoW [100K Words]	0.37123	0.48486	3.94466
BoCoW [10K Words - 10K CoW]	0.15224	0.43004	0.67044
BoCoW [10K Words - 50K CoW]	0.10744	0.24656	0.52707
BoCoW [10K Words - 100K CoW]	0.09772	0.20182	0.42846
BoCoW [10K Words - 500K CoW]	0.10469	0.14708	0.51105
BoCoW [10K Words - 1M CoW]	0.10591	0.13719	0.56226
BoCoW [50K Words - 10K CoW]	0.20313	1.88084	6.03937
BoCoW [50K Words - 50K CoW]	0.23216	1.55898	4.51521
BoCoW [50K Words - 100K CoW]	0.2312	1.43551	4.90338
BoCoW [50K Words - 500K CoW]	0.22413	1.11315	4.98509
BoCoW [50K Words - 1M CoW]	0.22152	1.00201	5.38621
BoCoW [100K Words - 10K CoW]	0.21515	3.76387	9.88841
BoCoW [100K Words - 50K CoW]	0.27142	3.47656	9.43542
BoCoW [100K Words - 100K CoW]	0.32346	3.27579	9.25295
BoCoW [100K Words - 500K CoW]	0.36022	2.57656	9.20264
BoCoW [100K Words - 1M CoW]	0.35315	2.28668	9.29149

distribution computed from all other samples (i.e., samples from class C2, C3, and R). Samples from random class R try to mimic the real world scenario when a sample does not contain any particular class and adds noise to the probability distribution generated from all other samples. To compute KLD-(within) though, samples for each class are randomly split and two probability distributions are generated.

We conducted experiments with the various number of words, classes, samples per class (including random). For example, we tested with up to 10 classes (i.e., 2, 3, 5, and 10 classes), up to 10000 words (i.e., 100, 1000, and 10000 words), and, up to 1000 samples per class (i.e., 10, 100, and, 1000 samples). For generating all other probability histograms we also tried to look at the effect of adding more noise i.e., adding an increasing number of samples (i.e., 10, 100, and, 1000) from the random distribution class. The increasing number of samples from the random distribution class mimics the scenarios when there are few samples (for each class) in the dataset.

Figure 6.5 and 6.6 depicts the results for these experiments. The information provided in these tables is exactly the same but organized differently to show the different finding of the analysis. Columns KLDxR represents the scenario when x number of samples from random distribution class are added to the all other samples probability histogram. The main findings are:

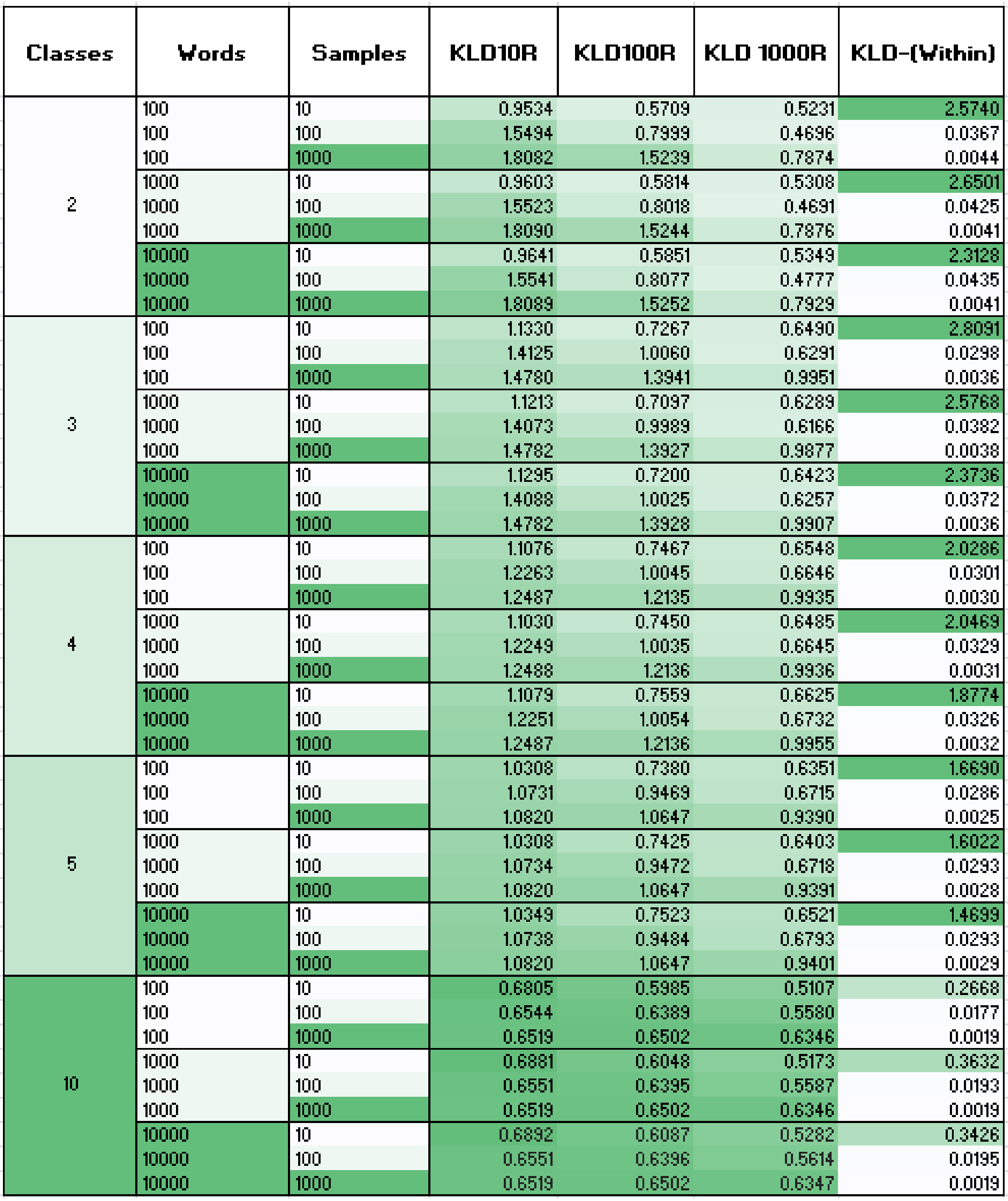
1. KLD-(within) can be higher than between class KLD when there are few classes and samples. For example, in our experiments within class KLD remained higher than between class KLD until the number of classes are less than 10 and the number of samples remained 10. It may be possible that this observation is particular to the class distribution rather than a general observation (shown in Figure 6.5).

2. Increasing the number of samples while keeping the number of words fixed increases between class KLD but decreases within class KLD (shown in Figure 6.5).

3. Increasing the number of words while keeping the number of samples fixed usually does not much affect both KLD's (shown in Figure 6.6).

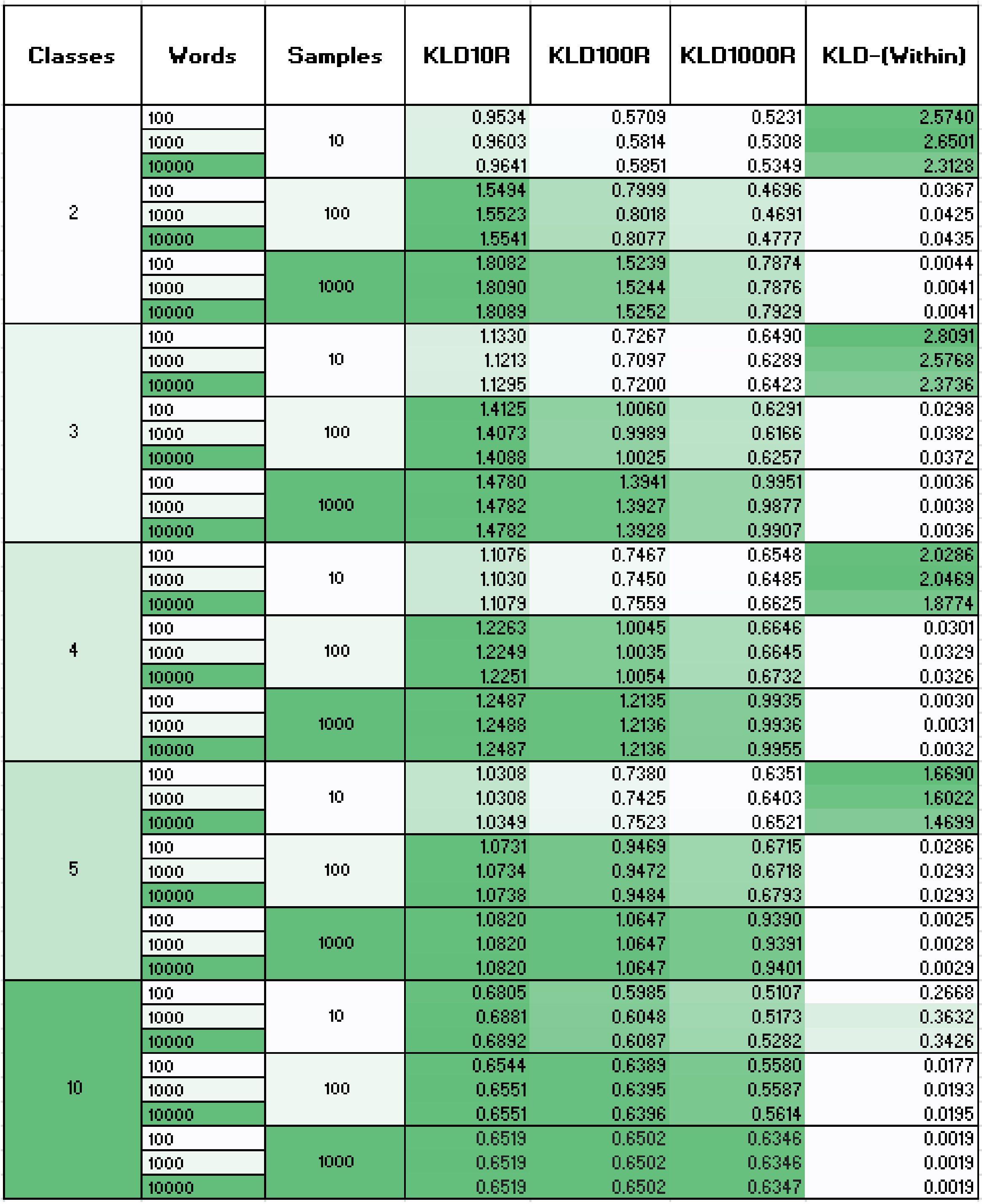
6.6 Frequency Based Analysis

The main motivation for this analysis is our criterion for selecting significant co-occurrences resulting in co-occurrences having different frequencies. Some significant co-occurrences are frequent that is they appear many times while others are rare that



Classes	Words	Samples	KLD10R	KLD100R	KLD 1000R	KLD-(Within)
2	100	10	0.9534	0.5709	0.5231	2.5740
	100	100	1.5494	0.7999	0.4696	0.0367
	100	1000	1.8082	1.5239	0.7874	0.0044
	1000	10	0.9603	0.5814	0.5308	2.6501
	1000	100	1.5523	0.8018	0.4691	0.0425
	1000	1000	1.8090	1.5244	0.7876	0.0041
	10000	10	0.9641	0.5851	0.5349	2.3128
	10000	100	1.5541	0.8077	0.4777	0.0435
	10000	1000	1.8089	1.5252	0.7929	0.0041
3	100	10	1.1330	0.7267	0.6490	2.8091
	100	100	1.4125	1.0060	0.6291	0.0298
	100	1000	1.4780	1.3941	0.9951	0.0036
	1000	10	1.1213	0.7097	0.6289	2.5768
	1000	100	1.4073	0.9989	0.6166	0.0382
	1000	1000	1.4782	1.3927	0.9877	0.0038
	10000	10	1.1295	0.7200	0.6423	2.3736
	10000	100	1.4088	1.0025	0.6257	0.0372
	10000	1000	1.4782	1.3928	0.9907	0.0036
4	100	10	1.1076	0.7467	0.6548	2.0286
	100	100	1.2263	1.0045	0.6646	0.0301
	100	1000	1.2487	1.2135	0.9935	0.0030
	1000	10	1.1030	0.7450	0.6485	2.0469
	1000	100	1.2249	1.0035	0.6645	0.0329
	1000	1000	1.2488	1.2136	0.9936	0.0031
	10000	10	1.1079	0.7559	0.6625	1.8774
	10000	100	1.2251	1.0054	0.6732	0.0326
	10000	1000	1.2487	1.2136	0.9955	0.0032
5	100	10	1.0308	0.7380	0.6351	1.6690
	100	100	1.0731	0.9469	0.6715	0.0286
	100	1000	1.0820	1.0647	0.9390	0.0025
	1000	10	1.0308	0.7425	0.6403	1.6022
	1000	100	1.0734	0.9472	0.6718	0.0293
	1000	1000	1.0820	1.0647	0.9391	0.0028
	10000	10	1.0349	0.7523	0.6521	1.4699
	10000	100	1.0738	0.9484	0.6793	0.0293
	10000	1000	1.0820	1.0647	0.9401	0.0029
10	100	10	0.6805	0.5985	0.5107	0.2668
	100	100	0.6544	0.6389	0.5580	0.0177
	100	1000	0.6519	0.6502	0.6346	0.0019
	1000	10	0.6881	0.6048	0.5173	0.3632
	1000	100	0.6551	0.6395	0.5587	0.0193
	1000	1000	0.6519	0.6502	0.6346	0.0019
	10000	10	0.6892	0.6087	0.5282	0.3426
	10000	100	0.6551	0.6396	0.5614	0.0195
	10000	1000	0.6519	0.6502	0.6347	0.0019

Figure 6.5: KLD and within class KLD for an exemplar dataset where class distributions are known. Columns KLDxR represents the scenario when x number of samples from random distribution class are added to the all other samples probability histogram.



Classes	Words	Samples	KLD10R	KLD100R	KLD1000R	KLD-(Within)
2	100	10	0.9534	0.5709	0.5231	2.5740
	1000		0.9603	0.5814	0.5308	2.6501
	10000		0.9641	0.5851	0.5349	2.3128
	100	100	1.5494	0.7999	0.4696	0.0367
	1000		1.5523	0.8018	0.4691	0.0425
	10000		1.5541	0.8077	0.4777	0.0435
	100	1000	1.8082	1.5239	0.7874	0.0044
	1000		1.8090	1.5244	0.7876	0.0041
	10000		1.8089	1.5252	0.7929	0.0041
3	100	10	1.1330	0.7267	0.6490	2.8091
	1000		1.1213	0.7097	0.6289	2.5768
	10000		1.1295	0.7200	0.6423	2.3736
	100	100	1.4125	1.0060	0.6291	0.0298
	1000		1.4073	0.9989	0.6166	0.0382
	10000		1.4088	1.0025	0.6257	0.0372
	100	1000	1.4780	1.3941	0.9951	0.0036
	1000		1.4782	1.3927	0.9877	0.0038
	10000		1.4782	1.3928	0.9907	0.0036
4	100	10	1.1076	0.7467	0.6548	2.0286
	1000		1.1030	0.7450	0.6485	2.0469
	10000		1.1079	0.7559	0.6625	1.8774
	100	100	1.2263	1.0045	0.6646	0.0301
	1000		1.2249	1.0035	0.6645	0.0329
	10000		1.2251	1.0054	0.6732	0.0326
	100	1000	1.2487	1.2135	0.9935	0.0030
	1000		1.2488	1.2136	0.9936	0.0031
	10000		1.2487	1.2136	0.9955	0.0032
5	100	10	1.0308	0.7380	0.6351	1.6690
	1000		1.0308	0.7425	0.6403	1.6022
	10000		1.0349	0.7523	0.6521	1.4699
	100	100	1.0731	0.9469	0.6715	0.0286
	1000		1.0734	0.9472	0.6718	0.0293
	10000		1.0738	0.9484	0.6793	0.0293
	100	1000	1.0820	1.0647	0.9390	0.0025
	1000		1.0820	1.0647	0.9391	0.0028
	10000		1.0820	1.0647	0.9401	0.0029
10	100	10	0.6805	0.5985	0.5107	0.2668
	1000		0.6881	0.6048	0.5173	0.3632
	10000		0.6892	0.6087	0.5282	0.3426
	100	100	0.6544	0.6389	0.5580	0.0177
	1000		0.6551	0.6395	0.5587	0.0193
	10000		0.6551	0.6396	0.5614	0.0195
	100	1000	0.6519	0.6502	0.6346	0.0019
	1000		0.6519	0.6502	0.6346	0.0019
	10000		0.6519	0.6502	0.6347	0.0019

Figure 6.6: KLD and within class KLD for an exemplar dataset where class distributions are known. This table depicts same information as shown Figure 6.5 but ordered differently to make certain pattern clear.

is they appear only a few times in entire dataset. We want to find out are there any subsets of co-occurrences (using their frequency) among the top-n most significant co-occurrences that produce better image retrieval results?

To test this we use frequency information of each co-occurrence that is the number if times it appeared in the dataset. Initially co-occurrences with top-n z-score values are obtained and then categorized into frequent, rare and in-range co-occurrences using their frequencies. All co-occurrence with frequencies in third quartile (Q3) are considered frequent co-occurrences. While co-occurrences with frequencies in the first quartile (Q1) are considered rare. And co-occurrences with frequencies between Q1 and Q3 are considered in-range co-occurrences. Bag-of-Co-occurring-words (BoCoW) are built for each category and image retrieval performance is measured on Paris and Oxford dataset.

Tables 6.9 and 6.10 show image retrieval results on Oxford and Paris datasets. For each number of visual words multiple values of co-occurring words (CoW) are chosen to represent BoCoW. Top five MAP scores are highlighted for each value of initial visual words to show any pattern, while the highest score is italicized as well. Results on Oxford dataset suggest that choosing co-occurrences from a single category results in maximum MAP of 0.28726 which is lower than what we obtained by using all kinds of co-occurrences i.e., 0.31011. For Paris dataset the maximum MAP of 0.36834 is achieved by using co-occurrences that have frequencies in the range of Q1 and Q3. This is slightly higher than what we obtained while we used all categories of co-occurrences i.e., 0.3602.

By looking at results it is clear that there is not any particular category of co-occurring words that always results in higher MAP scores. In fact, the best performing co-occurrence category is changing for each setting of visual words. Also, multiple categories of co-occurrences produce comparable results by choosing the same number of initial visual words.

6.7 Expansion of BoW

Until now we have compares two methods (i.e., BoW and BoCoW) which work very differently from each other. BoW method uses information extracted from each image while BoCoW uses emergent information from the whole dataset which may contain information related to each scene. Although the performance achieved by BoCoW was lower than BoW it still contains meaningful information. It would be interesting to

Table 6.9: Image retrieval performance for different categories of significant co-occurrences on Oxford dataset. Top five MAP scores are highlighted for each set of visual words while the highest score is italicized.

Method	MAP-(Frequent)	MAP-(In range)	MAP-(Rare)
BoCoW [10K Words - 10K CoW]	**0.10910**	0.08804	***0.14675***
BoCoW [10K Words - 50K CoW]	0.09022	0.09505	**0.10919**
BoCoW [10K Words - 100K CoW]	0.09807	0.09493	**0.10794**
BoCoW [10K Words - 500K CoW]	0.10003	**0.10691**	0.09487
BoCoW [10K Words - 1M CoW]	0.09365	0.08988	0.10464
BoCoW [50K Words - 10K CoW]	0.12988	0.17825	0.12114
BoCoW [50K Words - 50K CoW]	0.14144	**0.22287**	0.19049
BoCoW [50K Words - 100K CoW]	0.14007	**0.23097**	0.19697
BoCoW [50K Words - 500K CoW]	0.12803	**0.19993**	***0.24688***
BoCoW [50K Words - 1M CoW]	0.12749	0.19043	**0.24343**
BoCoW [100K Words - 10K CoW]	0.09841	0.09485	0.09617
BoCoW [100K Words - 50K CoW]	0.22424	0.10986	0.09384
BoCoW [100K Words - 100K CoW]	**0.25452**	0.13754	0.09145
BoCoW [100K Words - 500K CoW]	**0.25898**	**0.26359**	0.09629
BoCoW [100K Words - 1M CoW]	**0.25887**	***0.28726***	0.08929

see whether adding emergent information to the BoW based histograms could affect retrieval performance. We adapted a naive approach expanding BoW histograms (as shown in Figure 6.7) in order to add this extra information.

Since in previous experiments we did not find a category of co-occurrences that contains more distinctive information we suggest using all co-occurrences. We obtain all the co-occurrences and add them to BoW histogram to generate an expanded representation.

We also included frequency of each co-occurrence in an image in the new histogram. For these extended BoW histograms, the standard TF-IDF based weighting scheme is used instead. Here, the term frequency (TF) is the normalized frequency of a word in a given image. The document frequency (DF) is the total number of documents containing the term. This weighting system up-weights less frequent words and down-weights frequent words. Euclidean distance is computed to measure the closeness between the histograms.

Table 6.10: Image retrieval performance for different categories of significant co-occurrences on Paris dataset. Top five MAP scores are highlighted for each set of visual words while the highest score is italicized.

Method	MAP-(Frequent)	MAP-(In range)	MAP-(Rare)
BoCoW [10K Words - 10K CoW]	0.089616	0.108758	**0.131023**
BoCoW [10K Words - 50K CoW]	0.095947	0.092742	***0.140777***
BoCoW [10K Words - 100K CoW]	0.089418	0.099179	**0.130487**
BoCoW [10K Words - 500K CoW]	0.089808	**0.109712**	**0.105991**
BoCoW [10K Words - 1M CoW]	0.095015	0.096251	0.085993
BoCoW [50K Words - 10K CoW]	0.177972	0.103492	0.094911
BoCoW [50K Words - 50K CoW]	0.198932	0.185144	0.080202
BoCoW [50K Words - 100K CoW]	**0.200811**	**0.233884**	0.117454
BoCoW [50K Words - 500K CoW]	0.188999	***0.283383***	0.174583
BoCoW [50K Words - 1M CoW]	0.179611	**0.277217**	**0.207053**
BoCoW [100K Words - 10K CoW]	0.149951	0.089408	0.094434
BoCoW [100K Words - 50K CoW]	0.268844	0.121383	0.089574
BoCoW [100K Words - 100K CoW]	**0.307723**	0.182822	0.096010
BoCoW [100K Words - 500K CoW]	**0.312947**	**0.319414**	0.131313
BoCoW [100K Words - 1M CoW]	**0.320445**	***0.368341***	0.198149

6.7.1 Term Frequency-Inverse Document Frequency (TF-IDF)

This weighting scheme reflects the importance of a word in the corpus. The term frequency or TF is the measure of how frequently the term is in a document and is calculated as the total number of times a term or word occurs in a document. This raw frequency is normalized by dividing it by the total number of terms in a document. The normalization helps to remove the bias that gives very high term frequency to words in a very long document.

$$TF = \frac{\text{Frequency of a Term}}{\text{Total terms in document}}. \tag{6.10}$$

Inverse document frequency or IDF is the measure of importance of a term. A rare term is given a higher importance as compared to a frequent term. For example a word 'the' is less important as compared to 'White House' or 'Chinese garden' because it appears in the majority of documents.

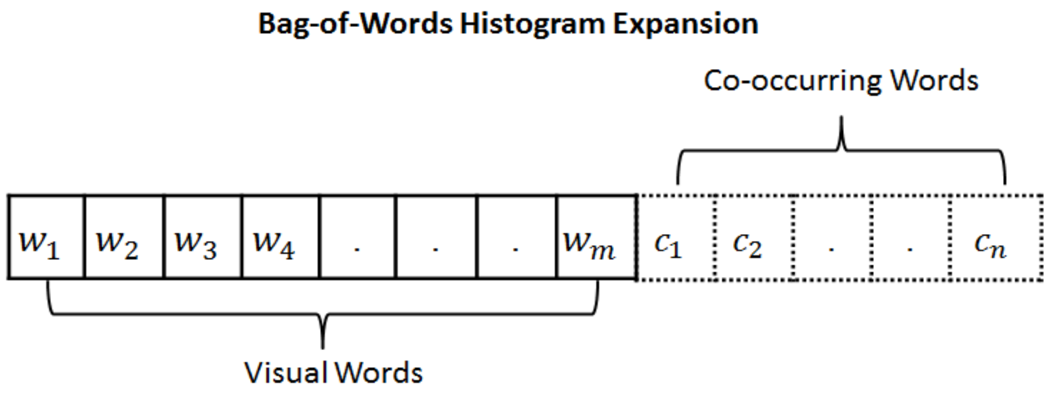

Bag-of-Words Histogram Expansion
Co-occurring Words
w_1 w_2 w_3 w_4 . . . w_m
c_1 c_2 . . c_n
Visual Words

Figure 6.7: Expansion of BoW histogram by concatenating CoWs at the end of histogram. Here w represents visual words and c represents co-occurring words.

$$IDF = -\log\left(\frac{\text{Total documents}}{\text{Documents containing a term}}\right). \tag{6.11}$$

6.7.2 Performance Measurement of BoW Expansion

Retrieval performance is measured by calculating MAP on both datasets. Table 6.11 and 6.12 show the comparison of standard and extended versions of the BoW approach for both datasets. On Oxford dataset merging emergent co-occurrences with the BoW did not increase the performance rather a slight decrease is noted in most cases. For the Paris dataset similar observations are made using 10,000 and 50,000 visual words. However for 100,000 visual words case MAP increased around 1.4% and reached 0.3854. To measure the significance of these results Wilcoxon signed-rank test is performed. Wilcoxon signed-rank test is a non-parametric statistical hypothesis test that assumes that the population is not normally distributed. We perform a paired test with confidence level set to 0.95. The test set here is MAP obtained by each query.

The null and alternative hypothesis are defined as:

$$H_0\text{:There is no difference between the two histograms} \tag{6.12}$$

$$H_1\text{:There is a difference between the two histograms} \tag{6.13}$$

Table 6.11: The comparison of standard BoW, BoCoW and extended BoW approached on Oxford dataset.

Method	MAP
BoW [10K Words]	**0.30182**
BoCoW [10K Words - 10K CoW]	0.11885
BoW [10K Words + [10K Words - 10K CoW]]	0.295661
BoW [50K Words]	**0.34484**
BoCoW [50K Words - 50K CoW]	0.24805
BoW [50K Words + [50K Words - 50K CoW]]	0.340179
BoW [100K Words]	**0.39783**
BoCoW [100K Words - 500K CoW]	0.31011
BoW [100K Words + [100K Words - 500K CoW]]	0.37912

To cater for multiple comparison error we used 'holm' correction to adjust p-values. On Oxford dataset for all visual words cases, we do not reject the null hypothesis as change in MAP obtained using extended BoW histogram is not found significant as p-values is greater than alpha (0.05). Similarly on Paris dataset for all cases we do not reject the null hypothesis. The slight increase noted in the 100,000 visual word case was also found insignificant.

6.8 Summary

In this chapter, we introduced a novel approach to represent images in the form of BoCoW. The performance of the new approach is measured on two datasets in an image retrieval scenario. We found that using co-occurrences extracted from emergent behavior for image representation performs poorly. We explored how the two representations encode information and which one is better using KLD score. We found it very hard to associate KLD score with image retrieval performance and felt the need for more investigation. We also experimented to find out any subset of co-occurrences that may contain more information than others. We also conducted experiments that focuses on combining Bag-of-Words information with emergent information. We found that in most cases adding these co-occurrences to the BoW representation did not improve the result, but rather decreased the MAP. As a future work, we recommend a careful analysis of the ranking function of the BoW expansion method and understanding KLD relationship with image retrieval performance.

Table 6.12: The comparison of standard BoW, BoCoW and extended BoW approached on Paris dataset.

Method	MAP
BoW [10K Words]	**0.2711**
BoCoW [10K Words - 10K CoW]	0.1522
BoW [10K Words + [10K Words - 10K CoW]]	0.2625
BoW [50K Words]	**0.3218**
BoCoW [50K Words - 50K CoW]	0.2321
BoW [50K Words + [50K Words - 50K CoW]]	0.3092
BoW [100K Words]	0.3712
BoCoW [100K Words - 500K CoW]	0.3602
BoW [100K Words + [100K Words - 500K CoW]]	**0.3854**

Chapter 7

A Feature Compression Strategy for Large Scale Image Collections

Note: Some portions of this chapter are based on previously published work (Khan *et al.*, 2012b).

This chapter presents a technique to compress SIFT feature descriptors without any need for training data. This is achieved by discarding the less significant bits from each dimension of the descriptor. Later, this method is compared with another compression schemes from the literature and the standard SIFT descriptor. The performance is evaluated in three different scenarios in the presence of various image transformations. In some cases the suggested approach achieved higher accuracy than standard SIFT. The main contribution that this chapter makes, is to suggest a feature compression technique and a detailed comparison with other approaches in the literature. The other contribution is a dataset for image retrieval applications that contain images from various places at the University of Otago.

7.1 Introduction

Many applications that deal with a large number of images often use various low-level features, e.g., SIFT (Lowe, 2004), SURF (Bay *et al.*, 2006), GLOH (Mikolajczyk and Schmid, 2005) and PCASIFT (Zickler and Efros, 2007) etc. These algorithms extract keypoints from an image and then represent the information in the form of high-dimensional feature vectors. Because of this high dimensionality these features suffer from the curse of dimensionality and have high memory requirements. The size of the descriptor becomes a real challenge for applications running on a single machine. The problem becomes worse when dealing with applications involving mobile or embedded devices. One possible solution is to generate fewer features. Unfortunately, fewer features results in significantly worse performance for image retrieval applications (Khan *et al.*, 2012). One alternative is to keep all the features, but reduce the memory footprint of each.

In this chapter, we present a feature compression scheme that reduces the size of SIFT by discarding less significant bits per dimension of feature descriptor. The scheme is useful for saving the features on the disk. We compare our method with the technique presented by Stommel (2010) and standard SIFT. The method of Stommel reduces the size of the descriptor down to just 1 bit per dimension. We evaluate the performance of all the approaches in different scenarios.

7.2 Related Work

The concepts of feature size and dimensionality reduction and the problems of curse of dimensionality are not new and are interlinked with each other. Zickler and Efros (2007) used principal component analysis (PCA) to reduce the size and dimensions of the SIFT descriptor. In their work PCA is applied to the normalized gradient patch across each key point to reduce the descriptor to just 36 dimensions. The descriptor is called PCA-SIFT and is capable of very high performance. PCA requires off-line training to estimate the covariance matrix, later used for PCA projection. Another descriptor BRIEF (Calonder *et al.*, 2010) uses a very short binary string descriptor based on naive Bayes comparison of image patches using either 256 or 128 bits. This descriptor is fast and gives performance competitive to SURF (Bay *et al.*, 2006) and U-SURF(Bay *et al.*, 2006) descriptors. Zhao *et al.* (2010) reduced the SIFT descriptor to just 36 dimensions by applying kernel projection on the orientation gradient patches rather than using smoothed weighted histograms. The generated descriptor is short and tolerant to geometric distortions. The approach is named KPB-SIFT and does not require a training stage.

Williams and Ledwich (2004) reduced the numbers and size of the SIFT descriptor by ignoring rotational invariance - an appropriate choice for indoor environments. Chandrasekhar *et al.* (2009) presented a technique based on transforming coding. They showed that SIFT and SURF descriptors can be reduced to less than 2-bits per dimension, providing a compression rate of 16 times relative to the conventional floating point representations. Features are encoded by first applying PCA and then scalar quantizing each dimension using arithmetic coding. An inverse process is applied during decoding. The approach produces better performance using 57-bits per descriptor that results in negligible image matching error. Stommel (2010) and Stommel *et al.* (2011) introduced binary descriptors that use just 1 bit per dimension. The median value of each element is used as a threshold to choose the value of the bit. i.e., 1 or 0. Johnson (2010) introduced a compression approach for feature descriptor and did not require any decompression during the matching process. The size of the feature is reduced by an order of magnitude, and still they achieve a detection rate of 95%. They converted SIFT, SURF, and GLOH into a canonical form that gives better results than the original descriptors. Brown *et al.* (2011) introduced a descriptor learning technique that uses linear and nonlinear dimensionality reduction along with linear discriminant analysis (LDA) and optimization methods to find the optimal parameters. The number of bits needed is further reduced to just 2 bits per dimension and their approach still maintains a good error rate. They also suggested the need for a variable number of bits for each dimension as the variance on each dimension can differ substantially across the descriptor.

In all these methods, the size and dimensionality reduction is achieved by either following a complex preprocessing step or a training stage is used to learn different parameters. These parameters are then used to produce a compressed descriptor. In some cases, specially the ones that deals with huge collection of images, getting the training data that is a representative of all kind of images in the dataset is very difficult. Our goal is to show how well a simple feature reduction approach can perform, without the need for any training data to extract compression parameters. We do that by comparing our method with other approaches in multiple image retrieval scenarios.

7.3 The Feature Compression Techniques

The presented technique is very simple, and it compresses the feature descriptor by keeping the most significant bits or discarding the least significant bits per dimension. The 8-bin vector depicted in Figure 7.1 is a one of the dimensions of the SIFT descriptor. The standard SIFT descriptor has 128 dimensions similar to this, and each dimension uses all 8 bits, hence, making a descriptor of 128 bytes. We suggest using fewer bits per dimension. By doing so, we can reduce the size of the SIFT descriptor. The figure shows different sizes of the compressed descriptor (i.e., adding up 128 of these dimensions) if we use 2, 4, 6, or all the 8 bits per dimension. This scheme performs encoding when ever a feature is saved to the disk in compressed form, and decoding is performed, when ever the feature is read from disk.

Encoding Stage

The following operations are performed during the encoding stage.

- Keeping the desired number of bits from each dimension of the SIFT descriptor. Until this point the total number of dimensions of the SIFT descriptor remains 128.
- Packing the remaining bits to recreate 8-bits representation. This results in a compressed descriptor with a fewer number of dimensions. Few zero bits are added if the last dimension has less than 8 bits in it.
- Saving the compressed features.

Before the matching stage, a set of features are read from the disk and are decoded before performing any matching.

Decoding Stage

The following operations are performed during the decoding stage.

- Reading a feature from memory
- Unpacking the reduced feature (e.g., 32 or 64 bytes, etc.) to again form a 128 dimensional feature vector. For this operation zero bits are added to each dimension to again make 8 bits per dimension.

After the decoding the features can now be used for matching.

7.4 Experimental Setup

In our experiments we compare four different compression schemes and the standard SIFT descriptor:

SIFT-6 Use the 6 most significant bits per dimension.

SIFT-4 Use the 4 most significant bits per dimension.

SIFT-2 Use the 2 most significant bits per dimension.

SIFT-1 Use the method of Stommel (2010) to compress the descriptor to 1 bit per dimension.

The first three schemes use the presented method. SIFT-2, SIFT-4, and SIFT-6 discard the least significant bits and hence do not require training data to find compression parameters. However, the method needs to pack and unpack bits during encoding and decoding stage as described before. Once we unpack the bits, we can use Euclidean distance to compare features. Two features having euclidean distance less than a predefined value are considered similar. We experimented with 5 different threshold values (i.e. 150, 170, 190, 210,and 230) to determine which results in better accuracy. We found that 210 produces maximum results and is used in our experiments. The second approach, i.e., SIFT-1 (Stommel, 2010), requires estimation of the median for each dimension from a training set of features. All the database images are used as training set. They suggest Hamming distance for matching features, which is very fast.

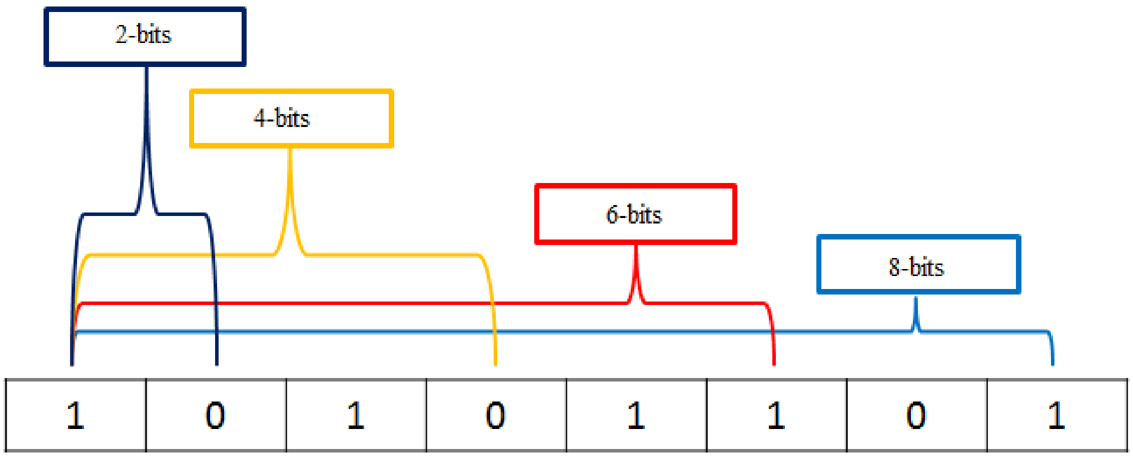


Figure 7.1: Our suggested feature compression approach is illustrated. The feature descriptor is compressed by only keeping the most significant bits. The 8-binned vector depicts a single dimension of the SIFT descriptor, and in a standard SIFT descriptor there are 128 dimensions. Moreover, in the standard method, all of the 8 bits are used per dimension, hence, making a descriptor of 128 bytes. If we use less number of bits per dimension, then SIFT descriptor size can be reduced. The figure shows different sizes of the compressed descriptor (i.e., adding up 128 of these dimensions) if we use 2, 4, 6, or all the 8 bits per dimension.

Section 7.4.1, describes the benchmark datasets used in our experiments. The three different sets of experiments based on image retrieval are performed . In all these experiments, a retrieved image is considered a correct match if it gets the maximum score, i.e., it is at the top of the retrieved images list. The matching score for a database image depends on the number of features that are correctly matched with the query. The performance of a compression method is measured by its *Accuracy*, which is the fraction of the total correctly matched queries to the total number of queries.

$$Accuracy = \frac{\text{Total\_Correct\_Matches}}{\text{Total\_Query\_Images}} \tag{7.1}$$

7.4.1 Benchmark Datasets

UK Benchmark Dataset

This dataset is presented in a paper by Nister and Stewenius (2006) and contains 10,196 images, and there are four images per scene in the dataset as shown in Figure 7.10. These images have different transformations, i.e., scale, rotation, illumination and view-point changes. In our experiments, we use first 4000 images from this dataset. The three images of every scene are kept as database image, and the fourth image is used for querying.

INRIA Holiday Dataset

This Dataset Jegou *et al.* (2008) contains 1491 images. Out of these, 991 images kept as database, and 500 images are for querying. Some query images have just one database image. Few images of this dataset are shown in Figure 7.11.

Otago University Dataset

This dataset contains 2000 images of both indoor and outdoor scenes. The data is prepared by us in the same manner as UK Bench data set i.e., there are four images per scene, and in total there are 500 scenes. We use the first three images of every scene as dataset images and the fourth image is kept for querying. The images contain different transformations like rotation, translation, viewpoint change, scaling and illumination changes and are much more challenging compared to the UK Bench Dataset. Some images of this dataset are shown in Figure 7.12.

7.5 Results

Section 7.5.1 describes our first experiment in which the compression methods are tested for robustness to several image transformations. Then in Section 7.5.2, the methods are tested using real image retrieval datasets. Finally, in Section 7.5.3, the methods are tested on the same image retrieval datasets while using keeping one image per scene as database images. This experiment provides a stringent test for the approaches because there is only one correct database image to retrieve.

7.5.1 Performance against Transformations

First, the performance of the three compression approaches is tested against various transformations on the images test bed provided by Khan *et al.* (2011). The experiments are carefully designed to test the matching performance in different transformations like rotation, blurring, illumination changes, noise, viewpoint and scale changes. In these experiments, the first 500 scenes from the UK Benchmark dataset (Nister and Stewenius, 2006) are used. There is only one database image and one query image per scene. The database images are transformed by applying various image transformations. The pre-transformed image is used as a query image. The goal is to retrieve the transformed image correctly. We start by testing the matching performance in the presence of rotation in the images. The trained images are rotated by 40, 135, 215, 250 and 300 degrees. From Figure 7.2 it can be seen that rotating does not cause any degradation in performance for any of the techniques.

The next experiment uses Gaussian blur, at three different levels of smoothing, i.e., $\sigma = 5$, 10 and 20. The results depicted in Figure 7.3 show that all compression approaches are robust to moderate level of blurring, and the performance do not degrade until the blurring becomes extreme. Even in the presence of excessive blurring, the accuracy of SIFT and SIFT-6 remained higher than 90%. The results of illumination changes are shown in Figures 7.4 and 7.5. In this case, a constant illumination value is either added or subtracted to each database image. Again, all methods perform well even up to quite significant changes in illumination.

Figure 7.6 shows results for various types and level of image noise. Salt and pepper noise causes the highest reduction in performance while SIFT-6 being the most robust method. Finally, Figure 7.7 shows results for the scale and viewpoint changes. These images are manually chosen from UK Bench dataset and are provided by (Khan *et al.*, 2011). Figure 7.7 depicts the results of both the scenarios in a single chart. We found a

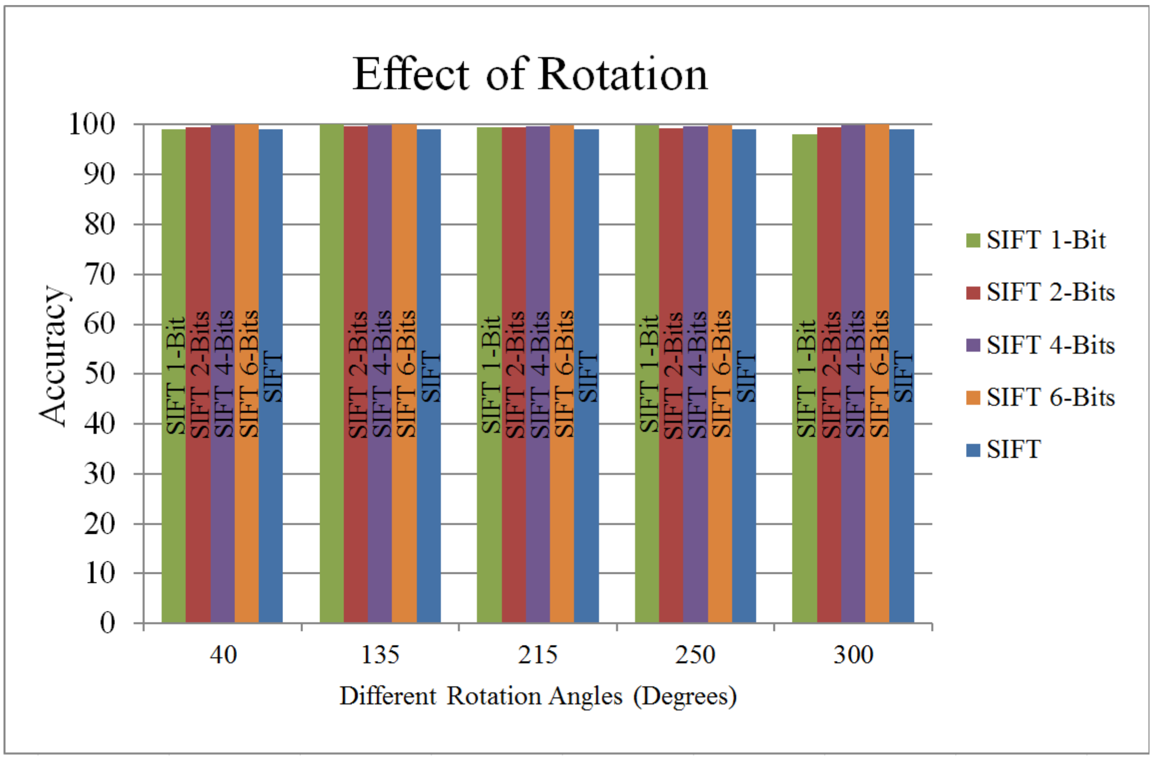


Figure 7.2: Matching performance in the presence of five different rotations angles applied to the database images.

decrease in the performance of all methods especially in the presence of large viewpoint changes.

7.5.2 Image Retrieval Scenario

In this section, we test the methods against three real datasets in an image retrieval scenario. The statistics about these datasets are described in Table 7.2. The image retrieval results are shown in Figure 7.8. We can see that all approaches performed well while SIFT-1 method performed slightly better than all other methods. For Otago University dataset the accuracies of all the approaches dropped significantly.

7.5.3 One Training Image

The final experiment is designed to check the matching accuracy and distinctiveness of the compressed descriptors in a scenario where only a single database image is available. The test is conducted on the first 1000 scenes of the UK Bench Dataset. The first image of every scene is kept as the database image while the last image is chosen as a query image. The matching results are depicted in Figure. 7.9. We can see that SIFT-1 and

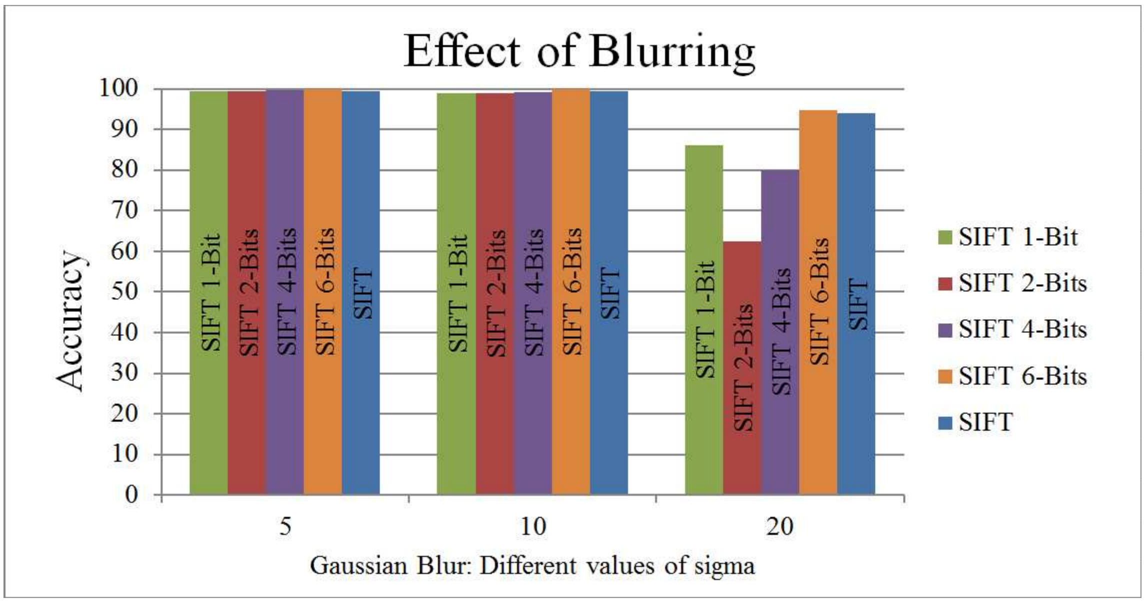


Figure 7.3: Matching performance against blurred images of three different sigmas (represented by x-axis) in Gaussian blur.

SIFT-4 performed better than SIFT. For other approaches at least an eight percent drop in accuracy is recorded.

7.5.4 Compression / Decompression Timings

This section discusses the timings required for compressing and decompressing a descriptor. A descriptor with a larger memory footprint takes less time to compress and decompress as compared to a descriptor with smaller memory footprint. Table 7.1 depicts the average compression and decompression timings for single feature descriptor. As the compressed features are stored in a file, the timings listed here also include file reading or writing time. For an image, compression and decompression, each takes 4 to 9 seconds depending upon the total number of descriptors in that image and the compression scheme.

7.6 Summary

In this Chapter, we investigated the effect of various image transformations on compressed descriptors in an image retrieval application. We also evaluated a simple feature reduction approach that does not require training.

We have found that the SIFT 1-bit is a competitive approach with such a small memory footprint. The key to its strength lies in the way the threshold is chosen to

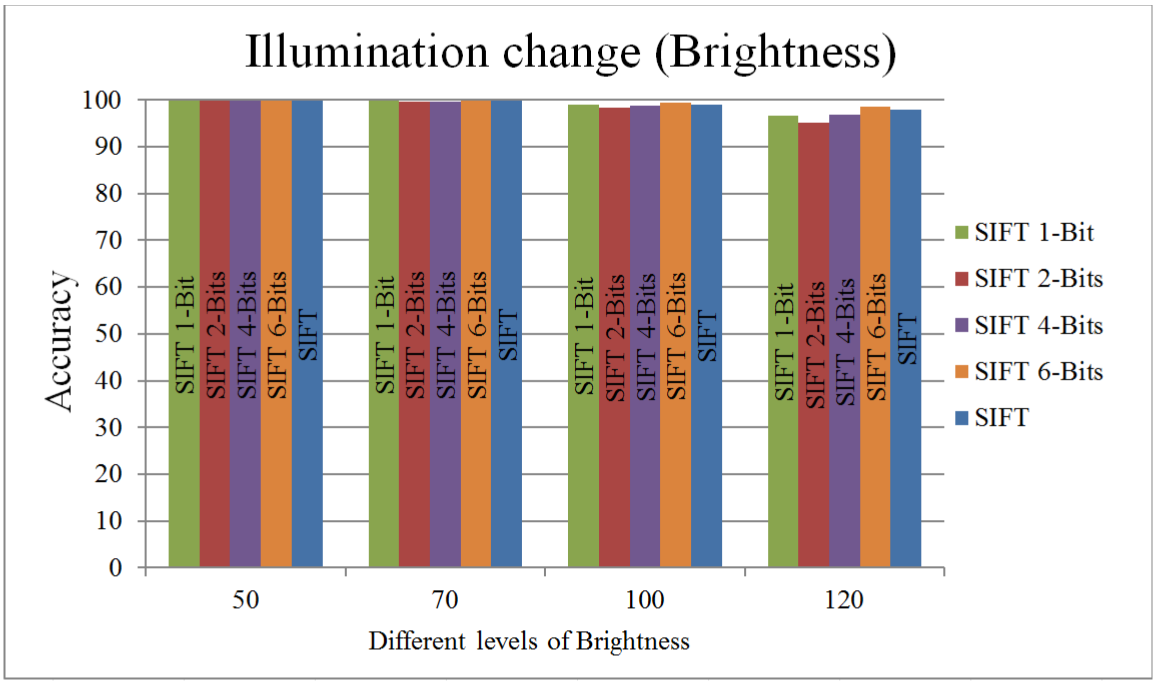


Figure 7.4: Matching performance of different reduction approaches compared with SIFT descriptor in the presence of brightness or addition of light.

cluster the real values to binary. Using the median value is key to its success. Stommel (2010) found that descriptor values are not symmetrically distributed, and many values in the descriptors occur in the least significant bits. Their approach is fast because the Hamming distance is used for feature comparison. The performance of our method is competitive to (Stommel, 2010) and in some cases better than SIFT. The main reason for this is that by only keeping the most important information we reduce noise or unnecessary information in the descriptor. Though this is not the generic behavior and in some cases we could also loose significant information. It depends on image dataset

Table 7.1: Average compression and decompression timings (in milliseconds) for single feature descriptor. As the compressed features are stored in a file, the timings listed here also include file reading or writing time.

Compression Scheme	Compression	Decompression
SIFT 2-Bit	2.68ms	2.52ms
SIFT 4-Bit	2.12ms	1.91ms
SIFT 6-Bit	1.33ms	1.25ms

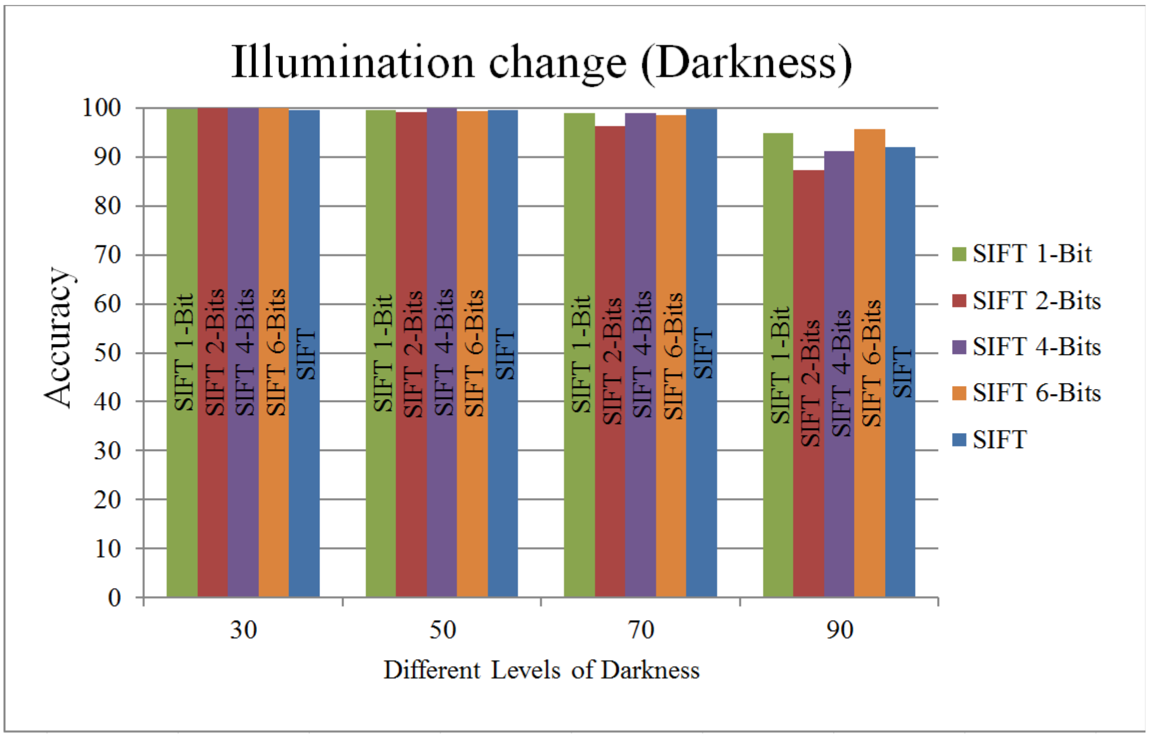


Figure 7.5: Matching performance of different reduction approaches compared with SIFT descriptor in the presence of darkness or reduction of light.

being experimented. One of the main advantage of our method is that the method does not need any training at all. Our approach, however, is slower because of packing and unpacking bits during encoding and decoding stages. All of the approaches are found to be robust under different image transformations.

Table 7.2: Three Benchmark Datasets used for checking the image retrieval results.

Dataset	Database	Query	Total Images
UK Bench	3000	1000	4000
INRIA Holiday	1491	500	1991
Otago University	1500	500	2000

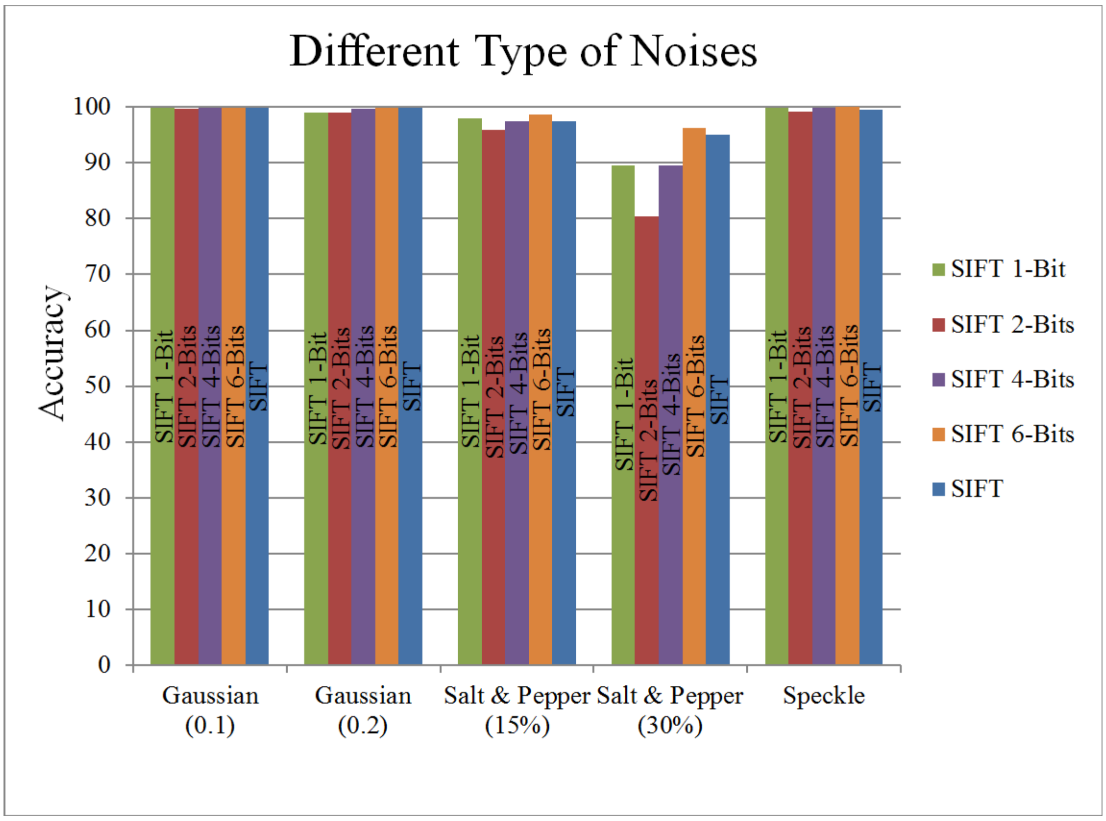


Figure 7.6: Matching performance of different reduction approaches compared with SIFT descriptor in the presence of different noise.

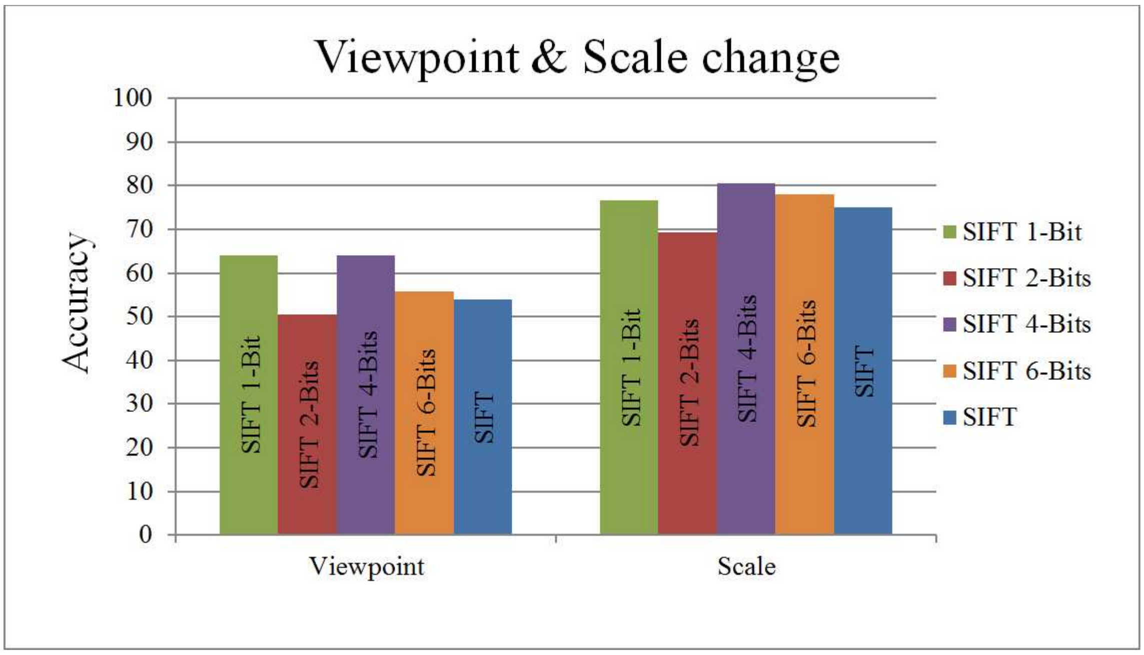


Figure 7.7: Matching performance of different reduction approaches in two different scenarios i.e., viewpoint and scale change.

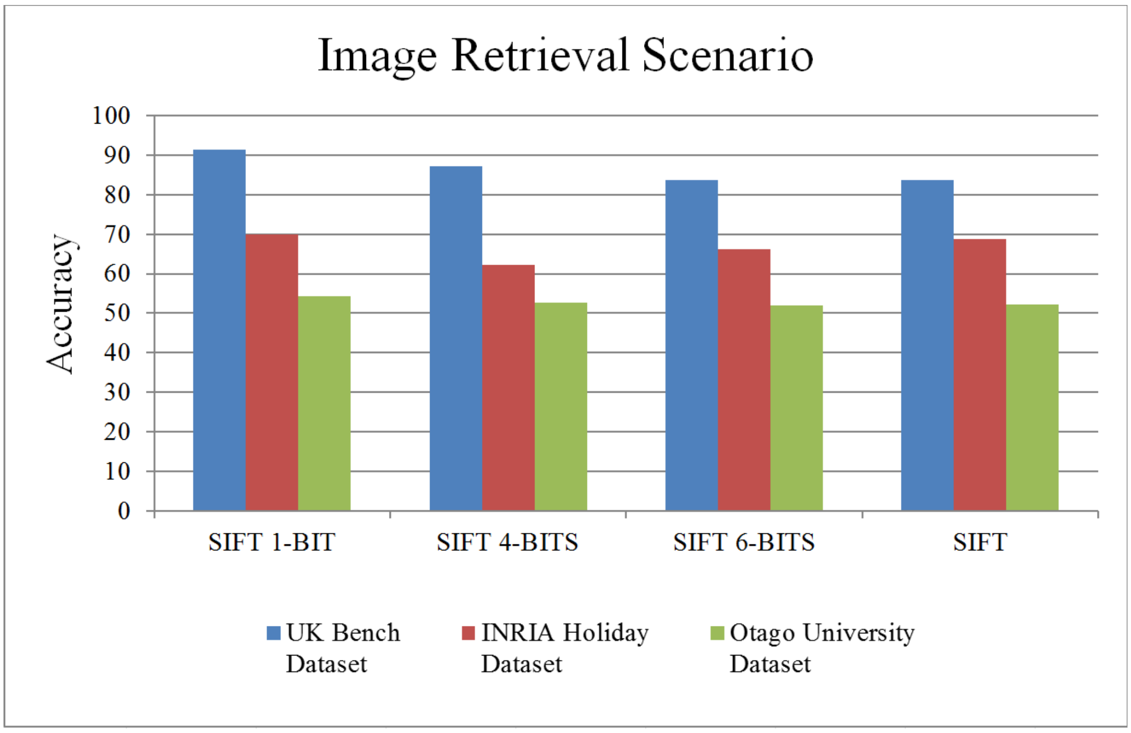


Figure 7.8: Matching performance of various compression approaches in an image retrieval scenario for various image datasets.

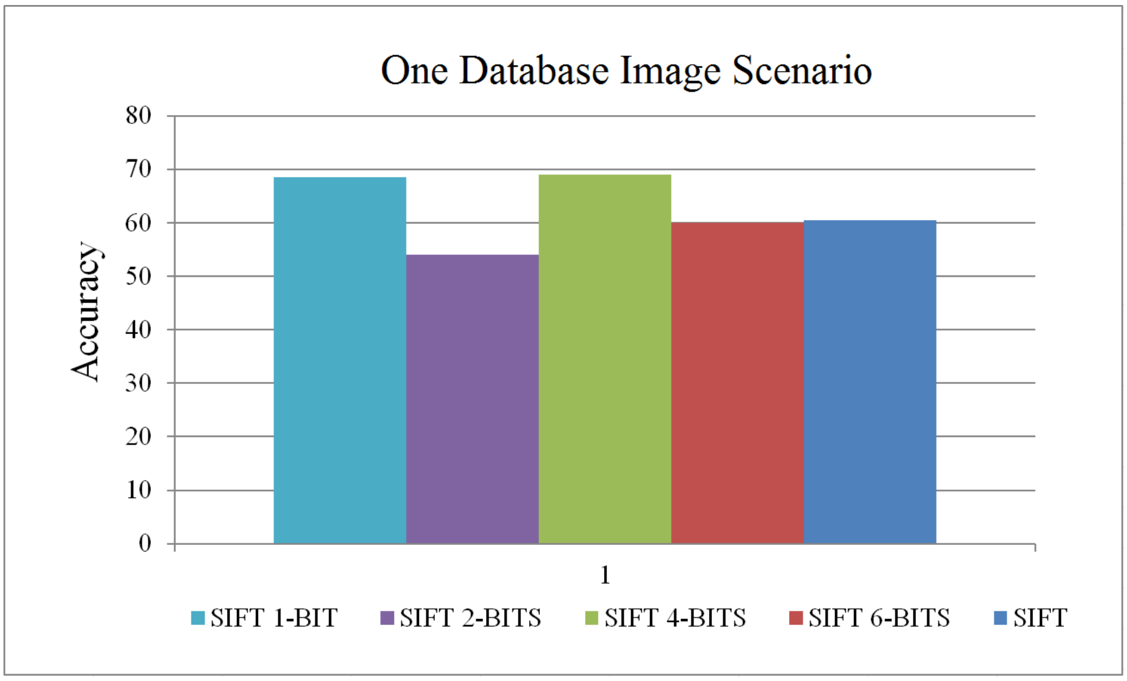


Figure 7.9: Matching performance of different approaches in an image retrieval scenario when there is only one database image.

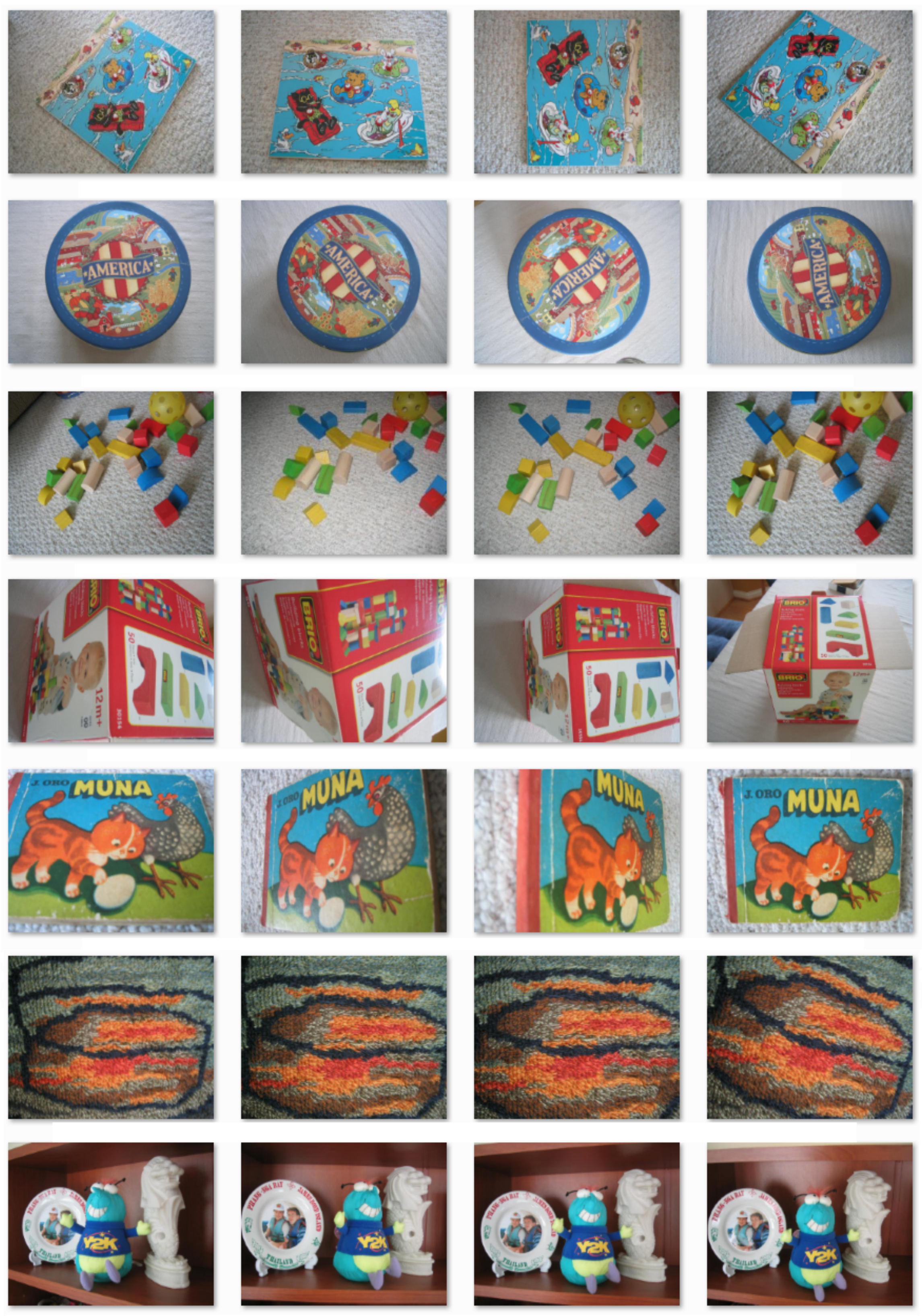


Figure 7.10: UK Benchmark dataset (Nister and Stewenius, 2006).

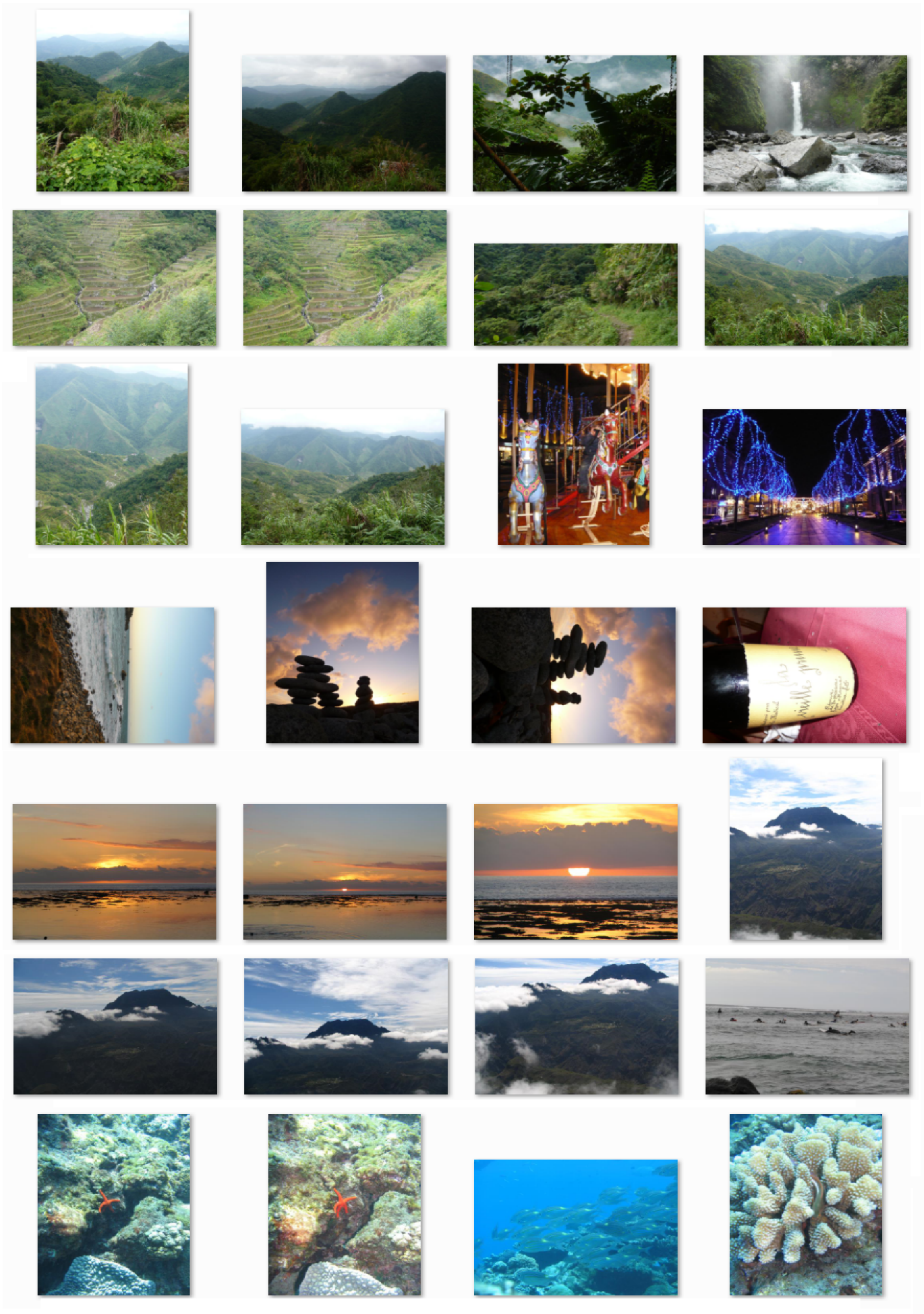


Figure 7.11: INRIA Holiday dataset (Jegou *et al.*, 2008)

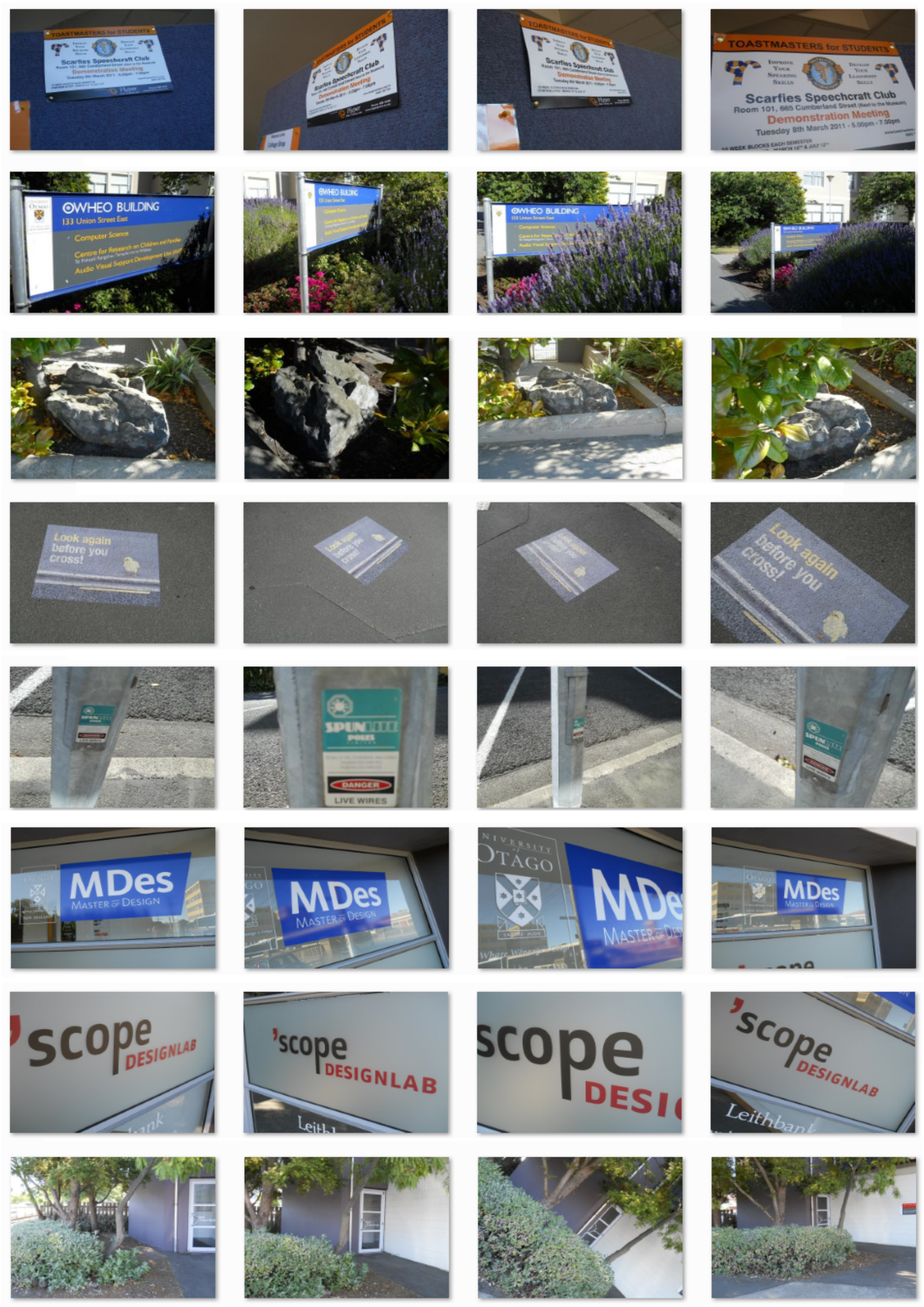

TOASTMASTERS for STUDENTS
Scarfies Speechcraft Club
Room 101, 665 Cumberland Street
Demonstration Meeting
Tuesday 8th March 2011 - 5.00pm - 7.00pm
OWHEO BUILDING
133 Union Street East
Computer Science
Centre for Research on Children and Families
Audio Visual Support Development Unit
Look again before you cross!
SPUNLITE POLES
DANGER
LIVE WIRES
MDes
MASTER of DESIGN
UNIVERSITY of OTAGO
'scope DESIGNLAB
Leithbank

Figure 7.12: Otago University dataset

Chapter 8

Conclusion

Computer vision and data mining strategies offer various methods to find visual patterns from images. The literature contains many approaches to discover visual patterns that are related to object or scene categories. Approaches to find emergent patterns which are generic, complex, and latent, have largely been ignored. This thesis investigated methods to discover such patterns in a large collection of images.

The first technique that is used to discover emergent patterns in images is based on data mining algorithms. The itemset mining algorithms are used to find emergent patterns. In these techniques the bag-of-words (BoW) representations of images is used to create a transactional dataset. To discover emergent patterns that frequently occurred in the dataset, the frequent pattern growth (FP-Growth) algorithm is used. The frequent itemsets with high confidence value are retained for visualization. To discover emergent patterns that rarely occurred in the dataset, the rare pattern tree (RP-Tree) algorithm is used. This technique is adapted from FP-Growth and finds rare patterns within minimum and maximum support thresholds. The patterns that are discovered using these methods are: *stripes and parallel lines*; *dots and checks*; *bright dots*; *single lines*; *intersections*; and *frames.*

The second approach presented in the thesis is a novel method that finds emergent patterns using graph theoretic algorithms. This approach represent visual words co-occurrences in images in a co-occurrence graph. Significant co-occurrences are chosen based on a binomial test, and each co-occurrence is assigned a z-score as the measure of it's importance. Normalized cuts are then used to extract the emergent clusters from a graph that only contains the top-n co-occurrences with the highest z-score values. Using this approach various interesting patterns, including some patterns that are related to object categories are revealed .

Another strategy that is described in the thesis uses important co-occurrences to obtain a bag-of-co-occurring-words (BoCoW) representation for each image. This technique provides an alternative to bag-of-words (BoW) for representing images. The top-n co-occurrences from the entire dataset are obtained and encoded in co-occurring-words (CoW). Then the CoWs found in each image are represented in an n-dimensional vector to form a BoCoW. When experimented in an image retrieval scenario, the results show that in all the cases BoW approach achieved higher mean average precision (MAP) than BoCoW. For further analysis, the top-n co-occurrences are divided into frequent, rare, and in-range co-occurrence categories, according to their occurrence frequency in the dataset. The image retrieval experiments could not find any subset of co-occurrences performs better than others. Later emergent co-occurrences are merged with BoW approach to create an expanded variant of BoW. The results show that in few cases, the BoW expansion method resulted in better MAP than the standard BoW but statistical significance test revealed it to be insignificant.

The thesis also presented an approach to compress the size of SIFT features to reduce memory requirements. The method reduces feature size by keeping only the most significant bits in each dimension of SIFT feature vector. The technique has been shown to have higher accuracy than standard SIFT in an image retrieval experiment with a smaller memory footprint.

This thesis makes the following contributions to the computer vision and data mining areas:

- **The emergent pattern mining techniques** described in Chapter 4, discover emergent patterns using itemset mining strategies. The kind of patterns that are discovered are: *stripes and parallel lines*; *dots and checks*; *bright dots*; *single lines*; *intersections*; and *frames.*

- **The emergent clusters discovery technique** described in Chapter 5, formulates the problem of finding emergent patterns as a subgraph mining problem. The visual words co-occurrences are represented in a graph. The statistical significance criterion determines the co-occurrences that appeared more than a random chance would allow, and hence are more important than others. The emergent patterns form dense clusters in the graph which are separated using normalized cuts.

- **The Bag-of-Co-occurring-Words (BoCoW) technique** described in Chapter 6, encodes significant co-occurrences into an n-dimensional vector that provides an alternative to BoW representation. This technique is used to compare the performance of emergent co-occurrences in an image retrieval scenario.

- **The feature compression technique** described in Chapter 7, provides a method to compress the SIFT descriptor to reduce its size. This is critical for an application that deals with a large number of images. The compressed approach has been found better than SIFT with a much smaller memory footprint.

- **The Toymix and Otago University datasets** used in Chapter 5 and 7, provides ground truth information. The Toymix dataset contains 6000 images from six objects. This dataset is used to evaluate the performance of the graph-based approach for finding emergent clusters. The Otago University dataset contains 2000 images and is designed for image retrieval applications. There are four images per scene, out of which, three images are used for training and one is used for testing.

8.1 Future Work

In this thesis, I explored multiple ways to discover emergent patterns from a large image collection without any supervision. Different factors can influence the pattern extraction process. Due to time constraints it was not possible for me to address these challenges in a single Ph.D. The contributions of this thesis raise the following issues for future work:

- Currently, I have only experimented with SIFT features. Since, a particular type of feature descriptor captures image properties in a specific way. It would be interesting to see how these patterns change, by changing the type of the feature or using multiple features together.

- In this thesis, I used association rules for finding important itemsets (based on the strength of relationship). Association rules could also be used to obtain higher level semantics. As an example, if four corner features appear together in neighborhood, then it may refer to a *frame* pattern etc.

- In this thesis the extracted emergent patterns are considered to have a flat structure. Instead, they could be organized in a hierarchy, to discover higher level relationships among them. As a possible direction, itemset mining could be performed using doublets (i.e., combination of two words), instead of individual visual word.

- When illustrating FIM and RIM techniques patterns are chosen randomly for visualization. Instead, a better method would be to cluster similar patterns and then see what this cluster looks like.
- In this work, I have only explored the bottom-up methods for pattern extraction. Other approaches such as latent Dirichlet allocation (LDA), that is based on top-down strategies could also be experimented with.
- In our latest experiments we showed that our feature compression technique can save a significant amount of memory. I plan to incorporate this into our work. Also, there is a need of doing few experiments in order to better understand the efficacy of this approach.

Bibliography

Adda, M., Wu, L., *et al.* (2007). Rare itemset mining. In *Machine Learning and Applications, 2007. ICMLA 2007. Sixth International Conference on*, 73–80. IEEE.

Agrawal, R., Imielinski, T., and Swami, A. N. (1993). Mining Association Rules between Sets of Items in Large Databases. In *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, Washington, D.C., May 26-28, 1993*, 207–216. ACM Press.

Agrawal, R., Srikant, R., *et al.* (1994a). Fast algorithms for mining association rules. In *Proc. 20th int. conf. very large data bases, VLDB*, 487–499.

Agrawal, R., Srikant, R., *et al.* (1994b). Fast algorithms for mining association rules. In *Proc. 20th int. conf. very large data bases, VLDB*, 487–499.

Antonie, M., Zaiane, O., and Coman, A. (2001). Application of data mining techniques for medical image classification. *MDM/KDD*, 94–101.

Baeza-Yates, R. A. and Ribeiro-Neto, B. (1999). *Modern Information Retrieval.* Addison-Wesley Longman Publishing Co., Inc.

Bastian, M., Heymann, S., and Jacomy, M. (2009). Gephi: An Open Source Software for Exploring and Manipulating Networks.

Bay, H., Tuytelaars, T., and Van Gool, L. (2006). Surf: Speeded up robust features. In *Computer Vision–ECCV 2006*, 404–417. Springer.

Bhatt, C. A. and Kankanhalli, M. S. (2011). Multimedia Data Mining: State of the Art and Challenges. *Multimedia Tools Appl.*, *51*(1), 35–76.

Binmore, K. and Davies, J. (2001). *Calculus: Concepts and Methods.* Cambridge University Press.

Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent dirichlet allocation. *the Journal of machine Learning research*, *3*, 993–1022.

Brin, S., Motwani, R., Ullman, J. D., and Tsur, S. (1997). Dynamic itemset counting and implication rules for market basket data. In *ACM SIGMOD Record*, 255–264. ACM.

Brin, S. and Page, L. (1998). The Anatomy of a Large-scale Hypertextual Web Search Engine. In *Proceedings of the Seventh International Conference on World Wide Web 7*, 107–117. Elsevier Science Publishers B. V.

Broder, A. (1997). On the resemblance and containment of documents. In *Compression and Complexity of Sequences 1997. Proceedings*, 21–29. IEEE.

Brown, M., Hua, G., and Winder, S. A. J. (2011). Discriminative Learning of Local Image Descriptors. *IEEE Trans. Pattern Anal. Mach. Intell.*, *33*(1), 43–57.

Calonder, M., Lepetit, V., Strecha, C., and Fua, P. (2010). BRIEF: Binary Robust Independent Elementary Features. In *Proceedings of the 11th European Conference on Computer Vision: Part IV*, 778–792. Springer-Verlag.

Cao, L. and Fei-Fei, L. (2007). Spatially coherent latent topic model for concurrent segmentation and classification of objects and scenes. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, 1–8. IEEE.

Chandrasekhar, V., Takacs, G., Chen, D., Tsai, S. S., Singh, J., and Girod, B. (2009). Transform coding of image feature descriptors.

Chum, O. and Matas, J. (2010). Unsupervised discovery of co-occurrence in sparse high dimensional data. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, 3416–3423. IEEE.

Chum, O., Philbin, J., Sivic, J., Isard, M., and Zisserman, A. (2007). Total recall: Automatic query expansion with a generative feature model for object retrieval. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, 1–8. IEEE.

Csardi, G. and Nepusz, T. (2006). The igraph software package for complex network research. *InterJournal*, 1695.

Csurka, G., Dance, C., Fan, L., Willamowski, J., and Bray, C. (2004). Visual categorization with bags of keypoints. In *Workshop on statistical learning in computer vision, ECCV*, 1–2.

Dalal, N. and Triggs, B. (2005). Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, 886–893. IEEE.

Dong, G. and Li, J. (1999). Efficient Mining of Emerging Patterns: Discovering Trends and Differences. In *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 43–52. ACM.

Duygulu, P., Barnard, K., de Freitas, J. F., and Forsyth, D. A. (2002). Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary. In *Computer VisionECCV 2002*, 97–112. Springer.

Fei-Fei, L., Fergus, R., and Perona, P. (2004a). Learning Generative Visual Models from Few Training Examples: An Incremental Bayesian Approach Tested on 101 Object Categories. In *Computer Vision and Pattern Recognition Workshop, 2004. CVPRW '04. Conference on*, 178–178.

Fei-Fei, L., Fergus, R., and Perona, P. (2004b). Learning generative visual models from few training examples: an incremental Bayesian approach tested on 101 object categories. In *Workshop on Generative-Model Based Vision*. IEEE.

Fernando, B., Fromont, E., and Tuytelaars, T. (2012). Effective use of frequent itemset mining for image classification. In *Computer Vision–ECCV 2012*, 214–227. Springer.

Fischler, M. A. and Bolles, R. C. (1981). Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Commun. ACM*, *24*(6), 381–395.

Gansner, E. R. and North, S. C. (2000). An open graph visualization system and its applications to software engineering. *SOFTWARE - PRACTICE AND EXPERIENCE*, 1203–1233.

Gao, J., Hu, Y., Liu, J., and Yang, R. (2009). Unsupervised learning of high-order structural semantics from images. In *Computer Vision, 2009 IEEE 12th International Conference on*, 2122–2129. IEEE.

Gilbert, A., Illingworth, J., Bowden, R., and England, G. X. (2008). Scale invariant action recognition using compound features mined from dense spatiotemporal corners. In *In ECCV*, 222–233. Springer.

Goethals, B. and Zaki, M. J. (Eds.) (2003). *FIMI '03, Frequent Itemset Mining Implementations, Proceedings of the ICDM 2003 Workshop on Frequent Itemset Mining Implementations, 19 December 2003, Melbourne, Florida, USA*. CEUR-WS.org.

Grauman, K. and Darrell, T. (2006). Unsupervised learning of categories from sets of partially matching image features. In *CVPR*, 2596–2603. IEEE.

Han, J., Cheng, H., Xin, D., and Yan, X. (2007). Frequent pattern mining: current status and future directions. *Data Mining and Knowledge Discovery*, *15*(1), 55–86.

Han, J. and Kamber, M. (2006). *Data Mining, Southeast Asia Edition: Concepts and Techniques*. Morgan kaufmann.

Han, J., Pei, J., and Yin, Y. (2000). Mining Frequent Patterns Without Candidate Generation. *SIGMOD Rec.*, 1–12.

Harris, C. and Stephens, M. (1988). A combined corner and edge detector. In *Alvey vision conference*, Volume 15, 50. Manchester, UK.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 770–778.

Hofmann, T. (2001). Unsupervised learning by probabilistic latent semantic analysis. *Machine learning*, *42*(1-2), 177–196.

Indyk, P. and Motwani, R. (1998). Approximate Nearest Neighbors: Towards Removing the Curse of Dimensionality. In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, 604–613. ACM.

Inokuchi, A., Washio, T., and Motoda, H. (2000). An apriori-based algorithm for mining frequent substructures from graph data. In *Principles of Data Mining and Knowledge Discovery*, 13–23. Springer.

Jegou, H., Douze, M., and Schmid, C. (2008). Hamming Embedding and Weak Geometric Consistency for Large Scale Image Search. In *Proceedings of the 10th European Conference on Computer Vision: Part I*, 304–317. Springer-Verlag.

Jégou, H., Douze, M., and Schmid, C. (2008). Hamming embedding and weak geometric consistency for large scale image search. In *Computer Vision–ECCV 2008*, 304–317. Springer.

Jegou, H., Harzallah, H., and Schmid, C. (2007). A contextual dissimilarity measure for accurate and efficient image search. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, 1–8. IEEE.

Jiang, M. C. and Coenen, F. (2009). Graph-based image classification by weighting scheme. In *Applications and Innovations in Intelligent Systems XVI*, 63–76. Springer.

Jiang, Y.-G., Ngo, C.-W., and Yang, J. (2007). Towards optimal bag-of-features for object categorization and semantic video retrieval. In *Proceedings of the 6th ACM international conference on Image and video retrieval*, 494–501. ACM.

Johnson, M. (2010). Generalized Descriptor Compression for Storage and Matching. In *BMVC*, 1–11. British Machine Vision Association.

Kadir, T. and Brady, M. (2001). Saliency, scale and image description. *International Journal of Computer Vision*, *45*(2), 83–105.

Khan, N. Y., McCane, B., and Mills, S. (2012). Feature set reduction for image matching in large scale environments. In *IVCNZ*, 67–72. ACM.

Khan, N. Y., McCane, B., and Wyvill, G. (2011). SIFT and SURF Performance Evaluation Against Various Image Deformations on Benchmark Dataset. In *Proceedings of the 2011 International Conference on Digital Image Computing: Techniques and Applications*, 501–506. IEEE Computer Society.

Khan, U. M., McCane, B., and Trotman, A. (2012a). Emergent Semantic Patterns In Large Scale Image Dataset: A Datamining Approach. In *Digital Image Computing Techniques and Applications (DICTA)*, 1–8. IEEE.

Khan, U. M., McCane, B., and Trotman, A. (2012b). A Feature Compression Scheme for Large Scale Image Retrieval Systems. In *Proceedings of the 27th Conference on Image and Vision Computing New Zealand*, 492–496. ACM.

Khan, U. M., Mills, S., McCane, B., and Trotman, A. (2014). Emergent Properties from Feature Co-occurrence in Image Collections. In *ICPR*, 2347–2352. IEEE.

Kim, G., Faloutsos, C., and Hebert, M. (2008). Unsupervised modeling of object categories using link analysis techniques. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, 1–8. IEEE.

Kleban, J., Xie, X., and Ma, W.-Y. (2008). Spatial pyramid mining for logo detection in natural scenes. In *Multimedia and Expo, 2008 IEEE International Conference on*, 1077–1080. IEEE.

Koh, Y. S. and Rountree, N. (2005). Finding Sporadic Rules Using Apriori-inverse. In *Proceedings of the 9th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining*, 97–106. Springer-Verlag.

Kotsiantis, S. and Kanellopoulos, D. (2006). Association rules mining: A recent overview. *GESTS International Transactions on Computer Science and Engineering*, 71–82.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, 1097–1105.

Kuramochi, M. and Karypis, G. (2001). Frequent subgraph discovery. In *Data Mining, 2001. ICDM 2001, Proceedings IEEE International Conference on*, 313–320. IEEE.

Lauw, H. W., Lim, E.-P., Pang, H., and Tan, T.-T. (2005). Social network discovery by mining spatio-temporal events. *Computational & Mathematical Organization Theory*, *11*(2), 97–118.

Lazebnik, S. and Raginsky, M. (2009). Supervised learning of quantizer codebooks by information loss minimization. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, *31*(7), 1294–1309.

Lee, Y. J. and Grauman, K. (2009). Shape discovery from unlabeled image collections. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, 2254–2261. IEEE.

Leordeanu, M. and Hebert, M. (2005). A spectral technique for correspondence problems using pairwise constraints. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, 1482–1489. IEEE.

Liu, D. and Chen, T. (2007). A topic-motion model for unsupervised video object discovery. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, 1–8. IEEE.

Lloyd, S. P. (1982). Least squares quantization in pcm. *IEEE Transactions on Information Theory*, 129–137.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. Volume 60, 91–110. Springer.

Malik, H. H. (2006). Clustering web images using association rules, interestingness measures, and hypergraph partitions. In *ICWE 06: Proceedings of the 6th international conference on Web engineering*, 48–55. ACM Press.

Mark J. Huiskes, B. T. and Lew, M. S. (2010). New Trends and Ideas in Visual Concept Detection: The MIR Flickr Retrieval Evaluation Initiative. In *MIR '10: Proceedings of the 2010 ACM International Conference on Multimedia Information Retrieval*, 527–536. ACM.

Martin, S., Brown, W. M., Klavans, R., and Boyack, K. W. (2011). OpenOrd: an open-source toolbox for large graph layout. In *IS&T/SPIE Electronic Imaging*, Volume 7868, 786806–786806–11. International Society for Optics and Photonics.

Martinet, J. and Satoh, S. (2007). A study of intra-modal association rules for visual modality representation. In *Content-Based Multimedia Indexing, 2007. CBMI'07. International Workshop on*, 344–350. IEEE.

Matas, J., Chum, O., Urban, M., and Pajdla, T. (2004). Robust wide-baseline stereo from maximally stable extremal regions. *Image and vision computing*, 761–767.

Megalooikonomou, V., Davatzikos, C., and Herskovits, E. H. (1999). Mining lesion-deficit associations in a brain image database. In *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, 347–351. ACM.

Mikolajczyk, K., Leibe, B., and Schiele, B. (2005). Local features for object class recognition. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, 1792–1799. IEEE.

Mikolajczyk, K. and Schmid, C. (2001). Indexing based on scale invariant interest points. In *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, 525–531. IEEE.

Mikolajczyk, K. and Schmid, C. (2005). A performance evaluation of local descriptors. Volume 27, 1615–1630. IEEE.

Mikolajczyk, K., Tuytelaars, T., Schmid, C., Zisserman, A., Matas, J., Schaffalitzky, F., Kadir, T., and Van Gool, L. (2005). A comparison of affine region detectors. *International journal of computer vision*, *65*(1-2), 43–72.

Mohri, M., Rostamizadeh, A., and Talwalkar, A. (2012). *Foundations of Machine Learning*. The MIT Press.

Morioka, N. and Satoh, S. (2010). Building compact local pairwise codebook with joint feature space clustering. In *Computer Vision–ECCV 2010*, 692–705. Springer.

Nister, D. and Stewenius, H. (2006). Scalable Recognition with a Vocabulary Tree. In *CVPR*, 2161–2168. IEEE Computer Society.

Nowak, E., Jurie, F., and Triggs, B. (2006). Sampling strategies for bag-of-features image classification. In *Computer Vision–ECCV 2006*, 490–503. Springer.

Nowak, S., Nagel, K., and Liebetrau, J. (2011). The CLEF 2011 Photo Annotation and Concept-based Retrieval Tasks. In *CLEF (Notebook Papers/Labs/Workshop)*, 1–25.

Ordonez, C. and Omiecinski, E. (1999). Discovering association rules based on image content.

Pan, H., Han, Q., Yin, G., Zhang, W., Li, J., and Ni, J. (2008). A ROI-Based Mining Method with Medical Domain Knowledge Guidance. In *Proceedings of the 2008 International Conference on Internet Computing in Science and Engineering*, 91–97. IEEE Computer Society.

Paul, Dooren, P. V., Blondel, V. D., Blondel, V. D., and Dooren, P. V. (2004). A Measure Of Similarity Between Graph Vertices. With Applications To Synonym Extraction And Web Searching. , (46), 2004.

Payet, N. and Todorovic, S. (2010). From a set of shapes to object discovery. In *Computer Vision–ECCV 2010*, 57–70. Springer.

Philbin, J., Chum, O., Isard, M., Sivic, J., and Zisserman, A. (2007). Object retrieval with large vocabularies and fast spatial matching. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, 1–8. IEEE.

Philbin, J., Chum, O., Isard, M., Sivic, J., and Zisserman, A. (2008a). Lost in quantization: Improving particular object retrieval in large scale image databases. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, 1–8. IEEE.

Philbin, J., Chum, O., Isard, M., Sivic, J., and Zisserman, A. (2008b). Lost in quantization: Improving particular object retrieval in large scale image databases. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, 1–8. IEEE.

Philbin, J., Sivic, J., and Zisserman, A. (2011). Geometric latent dirichlet allocation on a matching graph for large-scale image datasets. *International journal of computer vision*, *95*(2), 138–153.

Quack, T., Ferrari, V., and Gool, L. V. (2006). Gool. Video mining with frequent itemset configurations. In *In Proc. CIVR*, 360–369. Springer.

Quack, T., Ferrari, V., Leibe, B., and Gool, L. J. V. (2007). Efficient Mining of Frequent and Distinctive Feature Configurations. In *ICCV*, 1–8. IEEE.

Quack, T., Leibe, B., and Van Gool, L. (2008). World-scale Mining of Objects and Events from Community Photo Collections. In *Proceedings of the 2008 International Conference on Content-based Image and Video Retrieval*, 47–56. ACM.

Quelhas, P., Monay, F., Odobez, J.-M., Gatica-Perez, D., and Tuytelaars, T. (2007a). A thousand words in a scene. *IEEE transactions on pattern analysis and machine intelligence*, *29*(9), 1575–1589.

Quelhas, P., Monay, F., Odobez, J.-M., Gatica-Perez, D., and Tuytelaars, T. (2007b). A thousand words in a scene. *IEEE transactions on pattern analysis and machine intelligence*, *29*(9), 1575–1589.

Rajendran, P. and Madheswaran, M. (2010). Hybrid Medical Image Classification Using Association Rule Mining with Decision Tree Algorithm. *CoRR*.

Rushing, J. A., Ranganath, H. S., Hinke, T. H., and Graves, S. J. (2001). Using association rules as texture features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *23*(8), 845–858.

Russell, B. C., Freeman, W. T., Efros, A. A., Sivic, J., and Zisserman, A. (2006). Using multiple segmentations to discover objects and their extent in image collections. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, Volume 2, 1605–1614. IEEE.

Shi, J. and Malik, J. (2000). Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence*, *22*(8), 888–905.

Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Sivic, J., Russell, B. C., Efros, A. A., Zisserman, A., and Freeman, W. T. (2005). Discovering objects and their location in images. In *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, Volume 1, 370–377. IEEE.

Sivic, J., Russell, B. C., Zisserman, A., Freeman, W. T., and Efros, A. A. (2008). Unsupervised discovery of visual object class hierarchies. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, 1–8. IEEE.

Sivic, J. and Zisserman, A. (2003). Video Google: A Text Retrieval Approach to Object Matching in Videos. In *Proceedings of the International Conference on Computer Vision*, 1470–1477. IEEE.

Sivic, J. and Zisserman, A. (2004). Video data mining using configurations of viewpoint invariant regions. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, I–488. IEEE.

Stilou, S., Bamidis, P., Maglaveras, N., and Pappas, C. (2001). Mining association rules from clinical databases: an intelligent diagnostic process in healthcare. *Studies in health technology and informatics*, (2), 1399–1403.

Stommel, M. (2010). Binarising SIFT-Descriptors to Reduce the Curse of Dimensionality in Histogram-Based Object Recognition. *IJSIP*, 25–36.

Stommel, M., Langer, M., Herzog, O., and Kuhnert, K.-D. (2011). A Fast, Robust and Low Bit-Rate Representation for SIFT and SURF Features. In *SSRR*, 278–283.

Sun, M. and Van Hamme, H. (2011). Image pattern discovery by using the spatial closeness of visual code words. In *Image Processing (ICIP), 2011 18th IEEE International Conference on*, 205–208. IEEE.

Szathmary, L., Napoli, A., and Valtchev, P. (2007). Towards rare itemset mining. In *Tools with Artificial Intelligence, 2007. ICTAI 2007. 19th IEEE International Conference on*, 305–312. IEEE.

Tan, P.-N., Steinbach, M., and Kumar, V. (2005). *Introduction to Data Mining, (First Edition)*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc.

Tang, J. and Lewis, P. H. (2008). Non-negative matrix factorisation for object class discovery and image auto-annotation. In *Proceedings of the 2008 international conference on Content-based image and video retrieval*, 105–112. ACM.

Thomee, B. and Popescu, A. (2012). Overview of the ImageCLEF 2012 Flickr Photo Annotation and Retrieval Task. In *CLEF (Online Working Notes/Labs/Workshop)*, 54–58.

Tirilly, P., Claveau, V., and Gros, P. (2008). Language Modeling for Bag-of-visual Words Image Categorization. In *Proceedings of the 2008 International Conference on Content-based Image and Video Retrieval*, 249–258. ACM.

Troiano, L., Scibelli, G., and Birtolo, C. (2009). A Fast Algorithm for Mining Rare Itemsets. *ISDA*, *9*, 1149–1155.

Tsai, C.-F. (2012). Bag-of-words representation in image annotation: A review. *ISRN Artificial Intelligence*, *2012*.

Tsang, S., Koh, Y. S., and Dobbie, G. (2011). RP-tree: Rare pattern tree mining. In *International Conference on Data Warehousing and Knowledge Discovery*, 277–288. Springer.

Tuytelaars, T. and Mikolajczyk, K. (2008). Local invariant feature detectors: a survey. *Foundations and Trends® in Computer Graphics and Vision*, 177–280.

Wang, H., Zhao, G., and Yuan, J. (2014). Visual pattern discovery in image and video data: a brief survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 24–37.

Wang, X. and Grimson, E. (2008). Spatial latent dirichlet allocation. In *Advances in Neural Information Processing Systems*, 1577–1584.

Wang, X. and Gupta, A. (2015). Unsupervised learning of visual representations using videos. In *Proceedings of the IEEE International Conference on Computer Vision*, 2794–2802.

Williams, L. and Ledwich, S. (2004). Reduced SIFT Features For Image Retrieval and Indoor Localisation. In *Australian Conference on Robotics and Automation. Australian Robotics and Automation Association.*

Yan, X. and Han, J. (2002). gspan: Graph-based substructure pattern mining. In *Data Mining, 2002. ICDM 2003. Proceedings. 2002 IEEE International Conference on*, 721–724. IEEE.

Yang, L., Meer, P., and Foran, D. J. (2007). Multiple class segmentation using a unified framework over mean-shift patches. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, 1–8. IEEE.

Yuan, J. and Wu, Y. (2007). Spatial random partition for common visual pattern discovery. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, 1–8. IEEE.

Yuan, J., Wu, Y., and Yang, M. (2007a). From frequent itemsets to semantically meaningful visual patterns. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, 864–873. ACM.

Yuan, J., Wu, Y., and Yang, M. (2007b). From Frequent Itemsets to Semantically Meaningful Visual Patterns. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 864–873. ACM.

Zaki, M. J. (2000). Scalable algorithms for association mining. *IEEE Transactions on Knowledge and Data Engineering*, *12*(3), 372–390.

Zhang, J., Marszałek, M., Lazebnik, S., and Schmid, C. (2007). Local features and kernels for classification of texture and object categories: A comprehensive study. *International journal of computer vision*, *73*(2), 213–238.

Zhang, Y. and Chen, T. (2009). Efficient kernels for identifying unbounded-order spatial features. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, 1762–1769. IEEE.

Zhang, Y., Jia, Z., and Chen, T. (2011). Image retrieval with geometry-preserving visual phrases. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, 809–816. IEEE.

Zhao, G., Chen, L., Chen, G., and Yuan, J. (2010). KPB-SIFT: a compact local feature descriptor. In *ACM Multimedia*, 1175–1178. ACM.

Zhao, G. and Yuan, J. (2011). Discovering thematic patterns in videos via cohesive sub-graph mining. In *Data Mining (ICDM), 2011 IEEE 11th International Conference on*, 1260–1265. IEEE.

Zheng, Y.-T., Neo, S.-Y., Chua, T.-S., and Tian, Q. (2009). Toward a higher-level visual representation for object-based image retrieval. *The Visual Computer*, 13–23.

Zickler, S. and Efros, A. A. (2007). Detection of Multiple Deformable Objects using PCA-SIFT. In *AAAI*, 1127–1132. AAAI Press.