Lappeenranta
University of Technology

Teemu Kinnunen

BAG-OF-FEATURES APPROACH TO UNSUPERVISED VISUAL OBJECT CATEGORISATION

Thesis for the degree of Doctor of Science (Technology) to be presented with due permission for public examination and criticism in the Auditorium 1381 at Lappeenranta University of Technology, Lappeenranta, Finland on the 12th of December, 2011, at noon.

Acta Universitatis
Lappeenrantaensis 457

Supervisors Professor Heikki Kälviäinen
Professor Joni-Kristian Kämäräinen
Associate Professor Lasse Lensu

Machine Vision and Pattern Recognition Laboratory
Department of Information Technology
Faculty of Technology Management
Lappeenranta University of Technology
Finland

Reviewers Dr. Tech. Jorma Laaksonen
School of Science
Department of Information and Computer Science
Aalto University
Finland

Associate Professor Pavel Zemčík
Image and Video Processing Research Group
Department of Computer Graphics and Multimedia
Brno University of Technology
Czech Republic

Opponents Professor Tinne Tuytelaars
ESAT-PSI
Electrotechnical department of the faculty of Engineering
Katholieke Universiteit Leuven
Belgium

Dr. Tech. Jorma Laaksonen
School of Science
Department of Information and Computer Science
Aalto University
Finland

ISBN 978-952-265-181-5
ISBN 978-952-265-182-2 (PDF)
ISSN 1456-4491

Lappeenrannan teknillinen yliopisto
Digipaino 2011

Preface

This thesis draws from the research work carried out in the VisiQ project during the years 2008-2011. The project is a joint effort of Machine Vision and Pattern Recognition Laboratory (MVPR) in the Lappeenranta University of Technology and Media Technology Laboratory in Aalto University.

First of all, I want to thank my supervisors Professor Heikki Kälviäinen, Professor Joni Kämäräinen and Associate Professor Lasse Lensu. This thesis would not have been possible without your guidance.

For the financial support, I want to thank the VisiQ project, the Academy of Finland for funding the VisiQ Project and late Pasi Saarinen for doing important work to have the right circumstances to the people in the industry.

I want to thank the Aalto University Media Laboratory for giving me a workspace and for inspiring atmosphere. I want to thank the head of the laboratory Professor Pirkko Oittinen and its staff Mari, Jussi, Jan, Henri, Sami, Raisa, Stina and all the researchers named Mikko.

I also want to thank people working the MVPR Laboratory in the Lappeenranta University of Technology for numerous interesting discussions from various topics. I want to thank all the workers in the MVPR lab Jukka, Janne, Tuomas, Tomi, Teemu, Jussi, Pekka, Toni, Leena, Jarmo, Jani, Nataliya, Ivan, Ville, Arto, Ilmari and Tarja.

Finally, I would like to express my gratitude to my family for the support and Satu for the patience and her support.

Lappeenranta, December 2011

Teemu Kinnunen

Abstract

Teemu Kinnunen
Bag-of-Features Approach to Unsupervised Visual Object Categorisation
Lappeenranta, 2011
106 p.

Acta Universitatis Lappeenrantaensis 457
Diss. Lappeenranta University of Technology

ISBN 978-952-265-181-5
ISBN 978-952-265-182-2 (PDF)
ISSN 1456-4491

The large and growing number of digital images is making manual image search laborious. Only a fraction of the images contain metadata that can be used to search for a particular type of image. Thus, the main research question of this thesis is whether it is possible to learn visual object categories directly from images. Computers process images as long lists of pixels that do not have a clear connection to high-level semantics which could be used in the image search. There are various methods introduced in the literature to extract low-level image features and also approaches to connect these low-level features with high-level semantics. One of these approaches is called Bag-of-Features which is studied in the thesis. In the Bag-of-Features approach, the images are described using a visual codebook. The codebook is built from the descriptions of the image patches using clustering. The images are described by matching descriptions of image patches with the visual codebook and computing the number of matches for each code.

In this thesis, unsupervised visual object categorisation using the Bag-of-Features approach is studied. The goal is to find groups of similar images, e.g., images that contain an object from the same category. The standard Bag-of-Features approach is improved by using spatial information and visual saliency. It was found that the performance of the visual object categorisation can be improved by using spatial information of local features to verify the matches. However, this process is computationally heavy, and thus, the number of images must be limited in the spatial matching, for example, by using the Bag-of-Features method as in this study. Different approaches for saliency detection are studied and a new method based on the Hessian-Affine local feature detector is proposed. The new method achieves comparable results with current state-of-the-art. The visual object categorisation performance was improved by using foreground segmentation based on saliency information, especially when the background could be considered as clutter.

Keywords: bag-of-features, self-organizing map, local feature, unsupervised visual object categorisation, spatial verification, saliency detection, computer vision

UDC 004.932:004.92

Abbreviations and symbols

1-NN	1-Nearest Neighbours classifier
2-D	2 Dimensional
3-D	3 Dimensional
ANN	Artificial Neural Network
BMU	Best Matching Unit
BoF	Bag-of-Features
BoBoF	Bag-of-Bag-of-Features
CBIR	Content-Based Image Retrieval
DoG	Difference-of-Gaussian
FG	Foreground
FGFG	Foreground - Foreground
FGBG	Foreground - Background
GLOH	Gradient Location Orientation Histogram
HC	Histogram-based Contrast
HOG	Histogram of Oriented Gradients
HVS	Human Visual System
IDF	Inverse Document Frequency
LBP	Local Binary Pattern
LDA	Latent Dirichlet Allocation
LLE	Locally Linear Embedding
LP	Learning Predictor
MDS	Multi-Dimensional Scaling
MSER	Maximally Stable Extreme Regions
PCA	Principal Component Analysis
RANSAC	RANdom SAmple Consensus
RC	Region-based Contrast
RCC	Region-based Contrast Cut
RGB	Red-Green-Blue colour space
ROC	Receiver Operator Characteristics
SIFT	Scale Invariant Feature Transform
SOM	Self-Organizing Map
SURF	Speeded-Up Robust Features
SVM	Support Vector Machine

TF	Term Frequency
TF-IDF	Term Frequency-Inverse Document Frequency
UVOC	Unsupervised Visual Object Categorisation
VOC	Visual Object Categorisation

$\boldsymbol{A}$	Saliency map
bm	Index of the best match
$\boldsymbol{CB}$	Codebook matrix
$\boldsymbol{D}$	Descriptor matrix
$\boldsymbol{D_c}$	A list of common local feature descriptors
$\boldsymbol{d_i}$	Descriptor of the local feature
dx	A gradient of a local patch on x-axis
dy	A gradient of a local patch on y-axis
$\boldsymbol{f}$	Codebook histogram
$\boldsymbol{F}$	A list of codebook histograms. I.e. $f_1, \ldots, f_{N_{img}} \in \boldsymbol{F}$
$\hat{\boldsymbol{f}}$	Normalised codebook histogram
$fScore$	Sum of N_{lm} best matching local feature descriptors
g	Gaussian function
$\boldsymbol{G}$	An image smoothed with a Gaussian filter
$\Delta\boldsymbol{G}$	A difference of Gaussian image
$\boldsymbol{H}$	Hessian matrix
$\boldsymbol{I}$	An (grey-level) image
k	A factor that is used to build a scale scape in SIFT
$\boldsymbol{L}$	Spatial parameters of the local feature (x,y,rotation,scale)
$\boldsymbol{M}$	Second order moment matrix for Harris corners
$maxDist$	Threshold for accepting local feature match
$minHits$	Minimum number of matches for a local feature to be accepted
N	Size of the image patch in pixels
N_c	Number of categories
N_{cb}	Size of the codebook (number of codes)
N_{cand}	Number of candidate images for spatial matching
N_d	Number of dimensions
N_k	Size of the neighbourhood
N_{lf}	Number of local features
N_{lm}	Number of local features used to compute $fScore$
N_m	Number of best matches

$P(t, i)$	Performance of node (t,i)
$perf_{voc}$	Average classification accuracy over the classes
$perf_{uvoc}$	Average categorisation accuracy over the categories
R	Cornerness of a specific image region
$\boldsymbol{S}$	Image similarity matrix
s	Scale value in the local feature detection
$\boldsymbol{T}$	Transformation matrix between the images
t_{max}	Maximum number of iterations
$\boldsymbol{x}$	Spatial location (x,y) of a pixel in the image
$\boldsymbol{x}_{bm}$	Coordinate of the (BMU) node in a SOM
$\boldsymbol{x}_{c}$	Coordinate of the (neighbouring) node in a SOM
$\boldsymbol{X}$	Ground truth category labels
$\boldsymbol{X_i}$	Samples belonging to the ground truth category i
$\boldsymbol{Y}$	Predicted category labels
$\boldsymbol{Y_i}$	Samples assigned to the cluster i
α_0	Initial learning rate
α_{final}	Final learning rate
β	A user given parameter to define threshold for corner
ϵ	Threshold to stop learning process
λ_0	Initial neighbourhood weight for Neural Gas
λ_{final}	Final neighbourhood weight for Neural Gas
σ	Deviation used for local feature detection to capture scales
τ	Learning rate function for the SOM learning algorithm

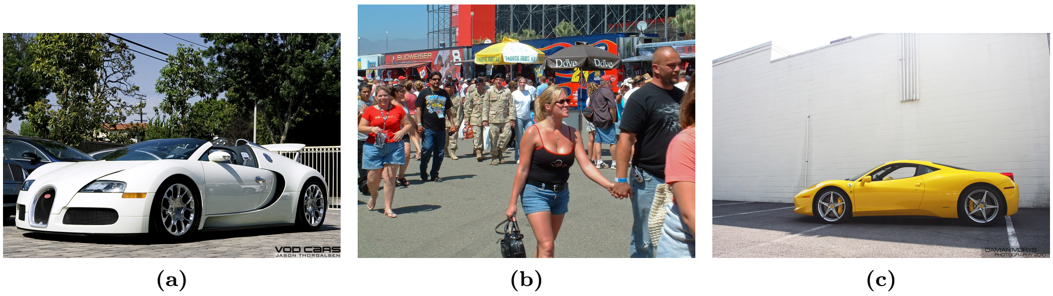
Contents

Chapter I
Introduction

The number of digital images has increased dramatically during the last decade. This originates from the popularity of digital cameras and the fact that nearly all mobile phones contain a built-in camera. The increasing number of images has lead to many image sharing services such as Flickr and Picasa, and also digital art sharing services such as DigitalArt and devianART. Nowadays, these image sharing services contain billions of images, e.g., Flickr alone already contains more than 6 billion images [54]. Despite the many image sharing services, only a fraction of the images are stored on the image sharing services; the majority of the images are stored in the photographers' personal computers and mobile phones.

Because of such high number of images, it is not possible to manually browse through all the images to find a particular type of image. Therefore, the image sharing services provide an image search for the users, to search for images from the massive image collections by typing in keywords. However, all of these services have one serious limitation. The content of each image must be described using metadata, i.e., by giving tags as in Flickr, or by giving a representative name as in DigitalArt and devianArt, and uploading images to the correct predefined category. This causes two problems: i) Images need to be described manually which is laborious, especially if it is done afterwards; ii) Descriptions of the images might vary significantly which makes the search impractical without intelligent cross-referencing keywords or use of taxonomies. For example, one might give the same tag for different kinds of images or give different tags for the same image. The problem of giving the same tag for different kinds of images is illustrated in Fig. 1.1. It shows example images from Flickr with the tag "sport car". Fig. 1.1a and 1.1b are very different, whereas Figs. 1.1a and 1.1c are more similar because both of the images actually contain a sport car. The difference between the Figs. 1.1a and 1.1b illustrates the problem of manual labelling of the images. Especially in the cases where there are many people labelling their own image collections and then another person is making a search, severe differences can occur. Of course, these images were chosen on purpose to empahise the problem.

One obvious solution to the manual image search problem is to use computers to organise



(a) (b) (c)

Figure 1.1: Three examples from Flickr with the tag “sport car”: (a) Sport car image by Jason Thorgalsen; (b) Sport car image by Stephen Dyrgas; (c) Sport car image by Damian Morys.

and find a particular type of images because the computers offer a great amount of computational power and they never get exhausted. However, it is not a straightforward matter to use computers to search images because the computers store and process colour images as a long list of pixels which do not have a clear connection to any high-level concepts which could be used to assist users to search images. This can be simply shown in practice by computing the pixel-wise differences of the images shown in Fig. 1.1. At first, the images must be resized to be able to make pixel-wise comparisons. Next, pixel-wise difference images are computed by computing the difference of each pair of pixels. For the images 1.1a and 1.1b the mean of the pixel-wise distances is 84.05, whereas for the images 1.1a and 1.1c the mean of the distances is 124.3. For the image pair 1.1b and 1.1c the mean of the distances is 97.07. According to the mean of the distances, the most similar pair of images are the Figs. 1.1a and 1.1b and the most dissimilar pair of images is Figs. 1.1a and 1.1c which do not agree with the higher level concepts. This simple example shows that pixel information cannot be used directly to find similar images or to organise a collection of images.

Smeulders et al. [85] made a comprehensive study on Content Based Image Retrieval systems (CBIR) prior 2000. One of their contributions was that they divided the problem of recognising real world objects using visual information into two problems: the sensory gap and the semantic gap. The sensory gap was defined as the gap between the object in the real 3-D world and the captured 2-D image. When a real world object is captured into a 2-D image, some of the information is lost, e.g., we cannot be sure what is behind the object because of occlusions. The semantic gap was defined as the difficulty of connecting extracted low-level features with the high-level concepts. There are many methods of extracting low-level features such as edges [8], lines and curves [20], blobs [60], colour histograms, etc., but it is not self-evident how these low-level features should be connected to the high-level concepts. However, by defining these two gaps, researchers can concentrate on closing one of the gaps in their research. In this thesis, the focus is on the semantic gap, i.e., we have a set of images that we want to organise based on high-level concepts.

Due to the existence of the semantic gap and the problem of laborious manual labelling, CBIR has received significant amount of attention from computer vision research and it has become one of the hot topics in computer vision [16]. It has lead to several approaches to connect low-level features to high-level concepts for automatic labelling. In this thesis, low-level features are connected to the high-level concepts which are defined as image categories. This task is called Visual Object Categorisation (VOC) which refers to the problem of detecting the category of the image. To solve the problem, one needs to extract low-level features successfully despite the existence of the sensory gap and then find the connection between the low-level features and the high-level concepts to make a connection over the semantic gap.

In VOC, low-level features are extracted from the images and then connected to the high-level concepts. Many of the current VOC methods [2, 3, 11, 14, 17, 21, 36, 38, 96] are based on local features, particularly in Scale Invariant Feature Transform (SIFT) [60]. A local feature is a description of a detected region in the image. Regions are detected using local feature detectors which are discussed in Sec. 3.1.1. Description can be an $N \times N$ grey-level patch [58] of a detected region or it can be a histogram of gradients [60]. The idea is that one can use local features to find similar regions from different images. The most trivial way to compute the similarity between the images is to compute the number of similar regions in the images using the local features [38]. One popular approach using these local features to describe the content of the whole image originates from text document search, where documents are described as occurrences of a predefined vocabulary, i.e., a set words. This approach is called the Bag-of-Words approach [6, 59]. In the VOC, visual words, i.e., local feature descriptors, are used instead of textual words. This approach is called the Bag-of-Features approach [84, 14].

1.1 Background

The most important works related to this thesis are [14, 92]. Csurka et al. [14] introduce the BoF approach for VOC which is extended to UVOC in this thesis. In this work, we utilise the performance measure from Tuytelaars et al. [92] and compare our results to the method in [92].

The BoF approach [84, 14] is illustrated in Fig. 1.2 and discussed more in detail in Chapter 3. First, regions of local features are detected from images. Second, these regions are converted into scale and rotation invariant descriptors in the local feature description step [60]. In the third step, a codebook is constructed using the descriptors of local features. In the study by Csurka et al. [14], the codebook generation was performed during the training phase using the k-Means clustering algorithm. In the best methods, however, the training ground truth is used to refine and probe more efficient codebooks [36, 58]. In the feature generation step, the extracted local features are matched against the generated codebook. A standard feature is the frequency vector over the codebook codes – "a bag of features". Finally, a category is assigned by feeding the feature vector to a classifier, such as the support vector machine (SVM) as was introduced by Csurka et al.

The annual Pascal VOC competition datasets [28] have become the standard benchmark for the supervised VOC. The annual competition attracts many research groups to submit their solutions to VOC, object detection and segmentation tasks. The Pascal image set itself has been updated annually by increasing the number of classes from 4

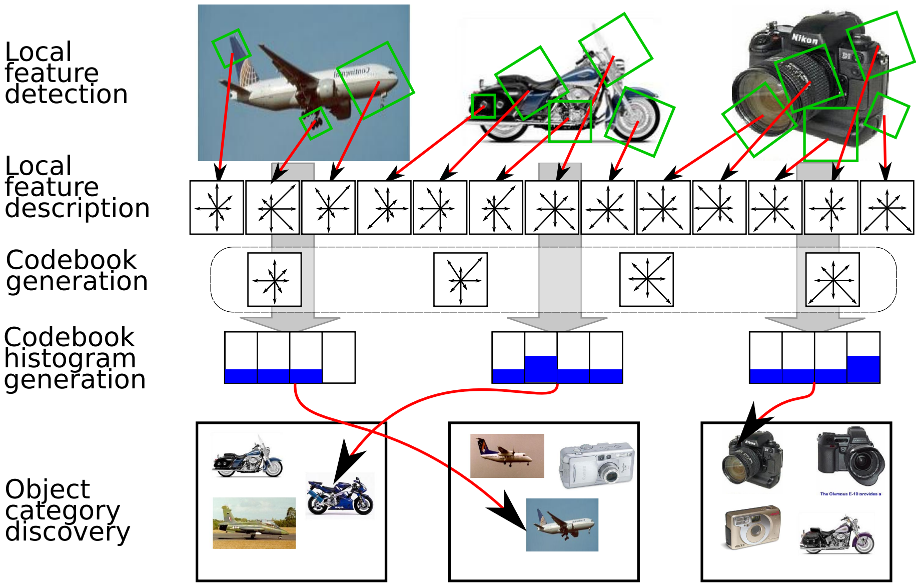

Local feature detection
Local feature description
Codebook generation
Codebook histogram generation
Object category discovery

Figure 1.2: Bag-of-Features approach applied to the supervised VOC. In the first row, detected local features are drawn with green rectangles. In the second row, detected local features are described by computing gradients in 8 directions that are illustrated with arrows. In the third row, visual vocabulary is built and codes are shown. In the fourth row, codebook histograms are shown. In the fifth row, members of three different classes are shown.

to 20, and also the number of the images has increased from hundreds to thousands. The rapid development in VOC has increased the mean average precision of the VOC methods evaluated in the Pascal VOC competition from 56.9% (2008) [26] to 77.1% (2010) [24]. The winner of the Pascal VOC 2010 was developed by Song et al. [87]. Their method integrates contextual information into the typical VOC method. They were using Context-SVM that can take an advantage of the context information. Their approach first predicts different visual objects in the image, and then refines the prediction based on context information gathered in the first prediction.

Albeit, the supervised VOC methods have been evolving rapidly and the performance of the state-of-the-art VOC method has increased dramatically in the annual Pascal VOC competitions [24, 26], the supervised VOC is now facing problems, especially when the number of classes is increased from tens to thousands. Deng el al. [17] studied the scalability of the supervised VOC. The first scalability issue that they found is the rapid increase of computation needed for training classifiers. They stated that it took 1 hour to teach a linear SVM classifier and 8 hours to test using a single 2.66GHz CPU. Even though the training can be done in parallel (e.g. by training each classifier on separate CPU), the limitations approaches quite fast. Also the required amount of memory grows quickly and becomes easily a bottleneck. The second issue that they found is the collapse of classification performance when the number of classes increases. This finding also supports our results published in [50]. The performance decreases from 34% with 200 classes to 6.4% with 10,000 classes [17]. However, people can recognize accurately more than 30,000 categories [5].

It is laborious to obtain training data for a large number of categories and also the supervised VOC has scalability issues as was shown by Deng et al. [17]. Thus this thesis focuses on Unsupervised Visual Object Categorisation (UVOC). In this thesis, unsupervised learning methods, especially self-organisation, are studied in order to develop a new method for UVOC. The benefit of UVOC is that it does not need training images which can be too laborious to obtain. In UVOC, the goal is to find images that belong to the same group or category, i.e. images that contain an object from the same category.

UVOC has been used for two different tasks: specific object discovery and object category discovery. In the first task, the problem is to find all instances of the specific object, such as a popular building or place, in unsupervised manner. In this task, the input is a set of images and then the method needs to find which of the images contain the same specific object [11]. In the latter task, the method needs to discover which of the images contain an object from the same category, e.g. cars, aeroplanes, faces. This problem is even more difficult because the appearance of the objects can vary more than in the first task. In this thesis, the latter problem is studied. Methods for finding groups of images that contain objects of the same class are explored and evaluated in many experiments.

Grauman and Darrell [38] introduced a method that compares the local features of each image with all the other images and then computes the number of matching local features. The number of matches defines the similarity between a pair of images. A graph is built by connecting images with edges. The weights of the edges are based on the similarity of the pair of images. After this, from the graph that was built, initial object categories are clustered with the Normalised Cuts algorithm [82]. These initial object categories are used to generate the prototypes of the categories. SVM classifiers are then taught with the prototype categories. Final categorisation result is obtained by predicting a category of each object with the SVM classifier. It is obvious that this approach is computationally rather heavy. Pairwise image comparison in the first step can easily become a bottleneck if one needs to categorise tens or even hundreds of thousands of images. Learning can also become a problem if the number of categories increases dramatically because each category needs an own SVM classifier to be learned. Grauman and Darrel had only four categories in their experiments, which can be because of the high computational complexity.

One of the popular approaches to categorising images in an unsupervised manner is to use Latent Dirichlet Allocation (LDA) [6]. Sivic et al. [83] presented an unsupervised method utilising the LDA model. They improved the original LDA by introducing hierarchical LDA (hLDA). With the hierarchy, they were able to improve the categorisation performance, but the results were reported only for a small number of categories and it is not obvious if the approach generalises well.

Bart et al. [3] developed a method that builds a visual taxonomy (TAX) using a topic model similar to the LDA. Instead of having a single level, they build a hierarchy similar to the method introduced by Sivic et al. [83]. In the TAX model, the topics are codebook feature histograms. Categories are histograms of topics and the categories are organised in a hierarchy in the way that the root node contains all possible topics and leaf nodes are the specific cases. The TAX is learned using interference with Gibbs sampling. However, in their experiment they used only 13 categories, which gives an idea of the scalability and computational complexity of their method. They said that it took 24 hours to learn a taxonomy for 1300 images. Thus, it is not very practical for learning thousands of categories.

Kim et al. [48] introduced an UVOC method that is based on link analysis. The method finds linkages between the features by pairwise matching of the local features of the images. The number of matching local features for a pair of images defines the weight of the edge. This is used as an initial setup which is then refined by running the PageRank algorithm to search "hubs". These hubs are then used to refine the weights of the nodes. The final categories are found by using spectral clustering on the matrix that defines similarities between the images. Kim and Torralba [49] improved the method by using an iterative method to refine the links between the images. In addition, the Normalised Cuts segmentation [82] is used to find the initial regions that are iteratively refined. Also, instead of directly using local features, they use BoF histograms and Histogram of Oriented Gradients (HOG) [15]. Thus, for one image there is a set of segments described with these descriptions. This improvement makes it more scalable and they used hundreds of thousands of images in their experiments. However, the number of categories remained rather low (5 categories).

Tuytelaars et al. [92] made a comprehensive study about UVOC based on the BoF approach. They compared local feature detectors, normalisation methods, categorisation methods and different sizes of visual codebooks. They also introduced a new method for evaluating the performance of a UVOC which is also used in thesis in addition to the method introduced by Sivic et al. [83].

In this thesis, the BoF approach is used because in the supervised VOC it has shown superior performance [87] and it is scalable. Moreover, Tuytelaars et al. [92] used BoF in their UVOC experiments and showed that the baseline methods achieve state-of-the-art results. BoF contains some weaknesses, e.g. spatial information is not used in the basic method, but these limitations and weaknesses can be solved. This thesis revisits and revises the standard parts of the BoF approach.

1.2 Objectives

The goal of the thesis is to study UVOC, i.e. to study a method that categorises any given set of images into groups of instances from the same category. Instances of the same category do not need to be images of the same specific object, but they can also be images of visually similar objects i.e. from the same object category. The focus in the thesis is in the BoF approach to UVOC, studying the bottlenecks and properties of the approach in this context and propose a novel processing method which improves the performance.

The main research question is that is it possible to develop a method that can categorise any given set of images using only visual information in similar manner with people without any training data?

The second research question is that how spatial information can be used to improve the UVOC performance using the BoF approach? The BoF approach disregards spatial information, thus it could be worthwhile to study how the spatial information can be used to improve categorisation performance.

The third research question is that how visual saliency information can be used to improve UVOC performance?

1.3 Contributions

This thesis studies UVOC using the BoF approach. The BoF approach consists of many separate steps and each step contains many possible methods that can be used. In this thesis, different methods were experimentally evaluated in the supervised VOC task and the most suitable methods were selected to be used in the proposed UVOC approach. The main contributions of the thesis are the study of UVOC based on the BoF approach, careful selection of the methods used in each step and the improvements to the standard BoF approach that are applicable for UVOC. In addition to these contributions, a few other noteworthy contributions were made. A list of the contributions of this thesis is as follows:

- Better visual codebook using SOM

  The first contribution of the work is an improvement to the codebook generation step. The standard method for the codebook generation method, k-Means [14], is replaced with a Self-Organising Map (SOM) [53]. It is shown that by changing the codebook generation method, it is possible to achieve better categorisation performance. These results were published in [50].

- Performance evaluation as a function of the number of categories

  It is self-evident that the categorisation performance decreases when the number of classes/categories increases. This thesis and publications referred to also show that the performance collapses quickly in a typical Bag-of-Features approach. It is important to study the behaviour of an approach with different numbers of categories because it gives insight about the scalability and generalisation ability of the categorisation method. The rapid collapse of a simple Bag-of-Features approach was published in [50, 51].

- New image set: Randomised Caltech-101

  In this thesis, the quality of Caltech 101 [30] as the benchmarking image set was evaluated and a few weaknesses were found. Based on the findings, a new image set was generated using the images and contour information from Caltech 101 and random landscape images from Google image search. Then, the effect of the randomisation was evaluated quantitatively. This contribution was published in [52].

- New image set: Abstract images

  An Abstract image set was collected as a part of the co-operation project VisiQ between the Lappeenranta University of Technology (LUT) and Aalto University. The initial idea come from LUT, but most of the images were collected by Mari Laine-Hernandez from Aalto University and she also carried out all the subjective experiments including an eye-tracking experiment for saliency detection and a manual categorisation task, which was performed by human participants. She also prepared saliency maps from fixation data. The contribution of this thesis is the idea for the new kind of image set for UVOC benchmarks and saliency detection, the evaluation of existing and proposed saliency detection and UVOC methods using the new image set, categorisation trees, and similarity matrices. This contribution is not yet published.

- New saliency detection method and boosting visual object categorisation using saliency maps

  In this thesis, a new method for saliency detection was introduced. The method is very simple, but it reaches the state of the art in saliency detection performance. Visual object categorisation is boosted using saliency information to choose only the important local features among all detected features. This contribution is not yet published.

- Extension of the BoF method by using spatial scoring of local features

  Unsupervised categorisation performance using the standard Bag-of-Features approach is improved by introducing spatial information. The spatial local feature verification step improves the categorisation performance significantly. This contribution is not yet published.

1.4 Structure of the thesis

This thesis is organised as follows: The second chapter presents datasets and how the performance can be evaluated in supervised VOC and unsupervised VOC. These evaluation methods are used in the experiments conducted in the following chapters. The third chapter introduces the Bag-of-Features approach and its different steps in the supervised and unsupervised visual object categorisation. In the fourth chapter, a few related approaches are discussed such as how spatial information can be used with the Bag-of-Features approach. The fifth chapter investigates visual saliency detection and how it can be applied to visual object categorisation. The sixth chapter discusses the results of the thesis and future work. Finally, the seventh chapter summarises the major contributions and findings of the thesis.

Chapter II

Datasets and performance evaluation

In this chapter, a few of the most popular datasets for VOC are presented and different methods for evaluating the performance of the supervised and unsupervised VOC methods are discussed. The choice of the dataset that is used for the VOC research is important because different datasets have different properties, such as the number of categories and images, or the number of objects in one image. The most popular datasets for VOC are Caltech-101 [30] and its extended version Caltech-256 [39], LabelMe [81] and the annual Pascal VOC competition datasets [29, 23, 26, 27, 24, 25].

To evaluate the performance of a supervised or an unsupervised VOC method, one needs to have a performance evaluation method. For the supervised VOC, the performance evaluation can be computed directly from the number of correctly classified samples and the total number of samples, but for UVOC the performance evaluation is more difficult because the categorisation result cannot be directly compared with the ground truth labels as in the supervised case. This chapter discusses different benchmark datasets and performance evaluation methods for the supervised and especially for the unsupervised VOC.

2.1 Benchmark datasets

In this section, a few of the most popular datasets for VOC are presented which are the followings: Caltech-101 [30]; Caltech-256 [39]; the annual Pascal VOC challenge datasets [29, 23, 26, 27, 24, 25]. Caltech-256 and Pascal VOC datasets provide the most difficult challenge. However, Caltech-101 is still important for the basic research since Caltech-256 and Pascal VOC datasets include 3-D pose variations and Pascal datasets also contain multiple objects in a single image. The images in Caltech-101 are of moderately good quality, a wide range of object categories and the foregrounds annotated, and most importantly, its 3-D pose variation is controlled, i.e., the objects are captured from the same view point.

2.1.1 Caltech-101

The Caltech-101 image set by Fei-Fei et al. [30, 31] contains roughly 8,700 images from 101 object categories and 500 background images. Each image contains only a single object, and the objects are cropped to the centre and rotated so that each object is roughly in the same pose. Some of the object categories are overlapping (e.g. *cougar body* and *cougar face*, *crocodile* and *crocodile head*, and *faces* and *faces easy*), but still the data set offers a very diverse collection of images because of the relatively large number of object categories. This is one of the most popular data sets used in VOC evaluations. An image collage from Caltech-101 is shown in Fig. 2.1 where one image is shown from each object category. The figure shows the great variability between the object categories. Some of the images are artificial, for example the images of the stop sign, the stapler and the crab are drawings. However, not all of the stop sign category images are drawings which causes rather large intra-category variability. An example of intra-category variability is illustrated in Fig. 2.2, where a few images are shown from the *barrel*, the *chair* and the *stop sign* categories. Such a great amount of intra-category variability makes it difficult to learn an object category only from a few examples. Some of the categories contain drawings and photos from the object category which can cause difficulties for VOC because of larger variability within the object category. The problem of overlapping object categories can be seen also in the image collage. Images from *faces* and *faces easy* originate from the same image where the latter image is a cropped version of the first one.

Fei-Fei et al. [30] have set the standards for VOC research by using Caltech-101. However, the published performance improvements saturated quite rapidly, and the state-of-the-art supervised VOC methods reached 84.3% classification accuracy with 30 training images per class and 73.2% with 15 images per class [102] in 2009. Ponce et al. [77] identified significant weaknesses in Caltech-101: the images are not challenging enough since the objects are captured from similar view points causing small variation in poses and scales, and often the object backgrounds are undesirably similar. These issues are visible in the original images in Fig. 2.3 where all the faces are frontal, their poses and locations are very similar, and some similar background structures appear in every image. Ponce et al. proposed a new data set collected from Flickr images. The set was used in Pascal VOC 2005 and is continuously updated for the annual competition. In 2005, there were only four categories, but in the 2006 competition, the number was increased to 10 [29] and in 2007 the number of categories increased to 20 [23].

2.1.2 Randomised Caltech-101

The images in Caltech-101 [30] have many good properties, but there are also certain undesirable properties due to the selection process of the images. Specifically, i) the objects are mainly in a standard pose and scale in the middle of the images; ii) background variability is insufficient in certain categories making it a more characteristic feature than the object's visual appearance.

To solve these issues of Caltech-101, as one of the contributions of the thesis, a new image set was generated based on Caltech-101: Randomised Caltech 101 [52]. The randomised Caltech-101 circumvents the problems related to the pose, location, and scale of the

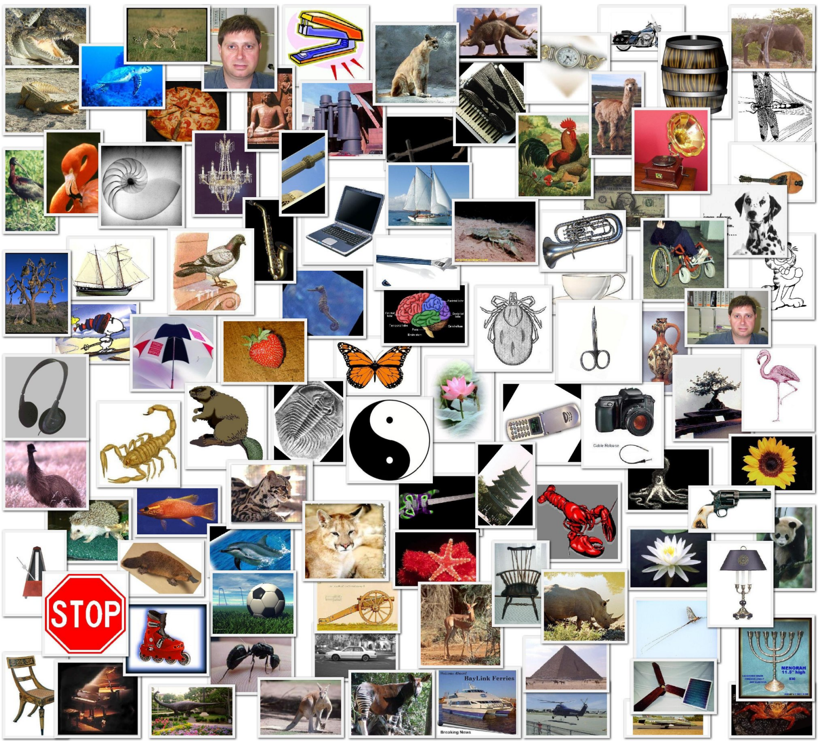


Figure 2.1: Collage of images from the Caltech-101 image set [30]. One image from each object category is shown.

object, and to the background. The remaining difference between the other available sets is the fact that the randomised data is still intrinsically 2-D whereas the others contain images in all 3-D poses. Genuine 3-D data is extremely difficult for computer vision methods, but it is questionable whether the problem is learnable just from the data, or should the 3-D pose information be provided as well. Since then, state-of-the-art results have been reported by Su et al. [88], but they used separate 3-D data in training. It can be agreed that genuine 3-D data is the ultimate challenge, but because the 2-D visual object categorisation is still an open problem, 2-D data sets, such as Caltech-101, are still important for method development, and therefore, making them more challenging is important. On the other hand, categorization can be performed, in principle, using 2-D methods which are trained with objects in different poses separately (car front, car rear, car side, etc.)

In the Randomised Caltech-101 [52] image set, the backgrounds are replaced with random landscape images from Google and the strong prior of the object placement and pose is reduced by random Euclidean transformations. The randomisation process is illustrated in Fig. 2.3. A collage of the image set is shown in Fig. 2.5.

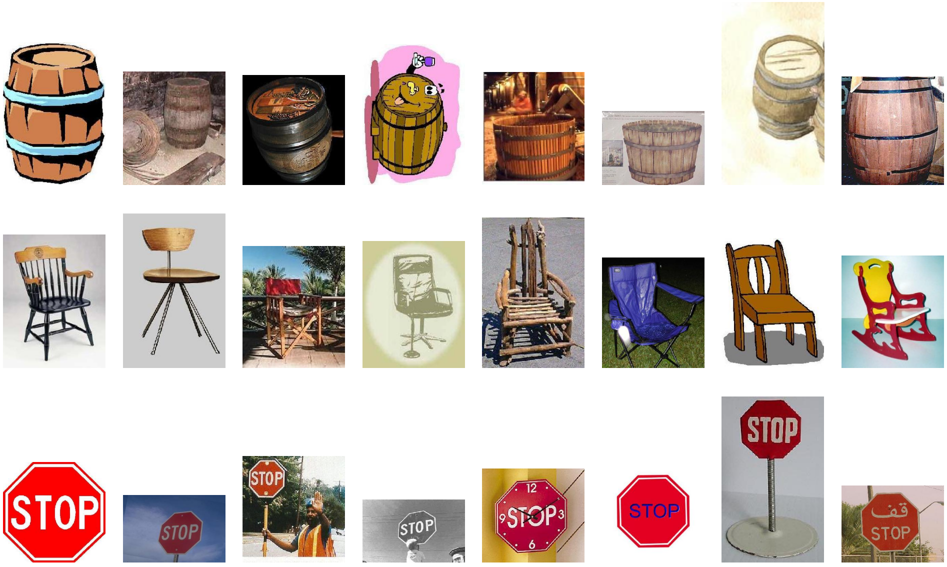


Figure 2.2: Examples from Caltech-101 [30] to illustrate intra-category variability of barrels, chairs and stop sign categories.

Randomisation process

Contours of the foreground objects have been annotated for all the images in Caltech-101 [30, 31]. Using the annotation data, the foreground regions were cropped, geometrically transformed, and drawn onto other backgrounds. In the randomisation process, random rotations of ±20° were applied. The range of angles was selected to limit the variations below the direction sensitivity of the human visual system [98]. Random translations were achieved by positioning the transformed regions randomly onto the random background images from Google. The scale was not explicitly changed, but the varying size of the random backgrounds implicitly changed the proportional object scale.

The minor pose and alignment variance of the original images is visible in the middle row in Fig. 2.4, where the selected categories are clearly recognisable from their average images. On the other hand, the average images become blurry when the averages have been computed after random rotations only. This can be seen clearly for the natural objects in the rightmost column of the figure while two simple human-made objects, the stop sign and ying-yang symbol, are still recognisable due to the rotation limits. It is evident that the randomised rotations and translations, and the implicit scale changes prevent the utilisation of the strong prior related to the object alignment and pose in the original Caltech-101 [30] images for VOC learning.

The importance of background randomisation is not evident from the average images in Fig. 2.4, but is quantitatively verified by the experiments in the experiments Sec. 3.5.1. Natural scenery and landscape images were gathered from the Internet using Google and the foreground objects were embedded onto these randomly selected backgrounds at random locations. It is noteworthy, that the images cannot be considered as "natural"

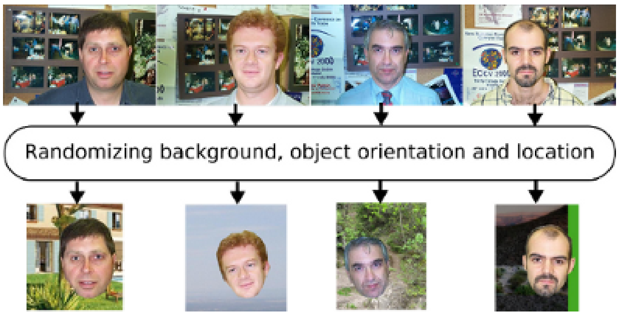


Figure 2.3: Randomising Caltech-101.

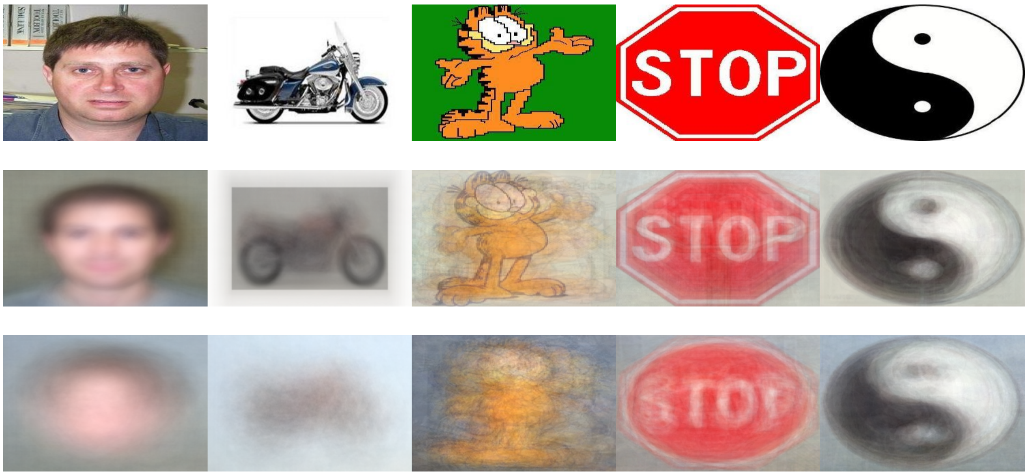


Figure 2.4: Average category images of Caltech-101 [30] and Randomised Caltech-101 [52]: (1st row) Examples of original Caltech-101 images; (2nd row) average category images of the original ones; (3rd row) the average of randomised images.

anymore because the objects do not appear in their typical scene. However, methods based purely on the object appearance and tolerating geometric variation should remain unaffected while methods which exploit the insufficient background variation in Caltech-101 [30] may severely fail.

Caltech-101 [30] can be claimed to be still useful for research since it provides good-quality category data with controlled 2-D variation. Thus the Caltech-101 and Randomised Caltech-101 [52] image sets are used as the main image sets in the thesis. In addition, Caltech-256 [39], which is introduced next, is used in a few experiments.

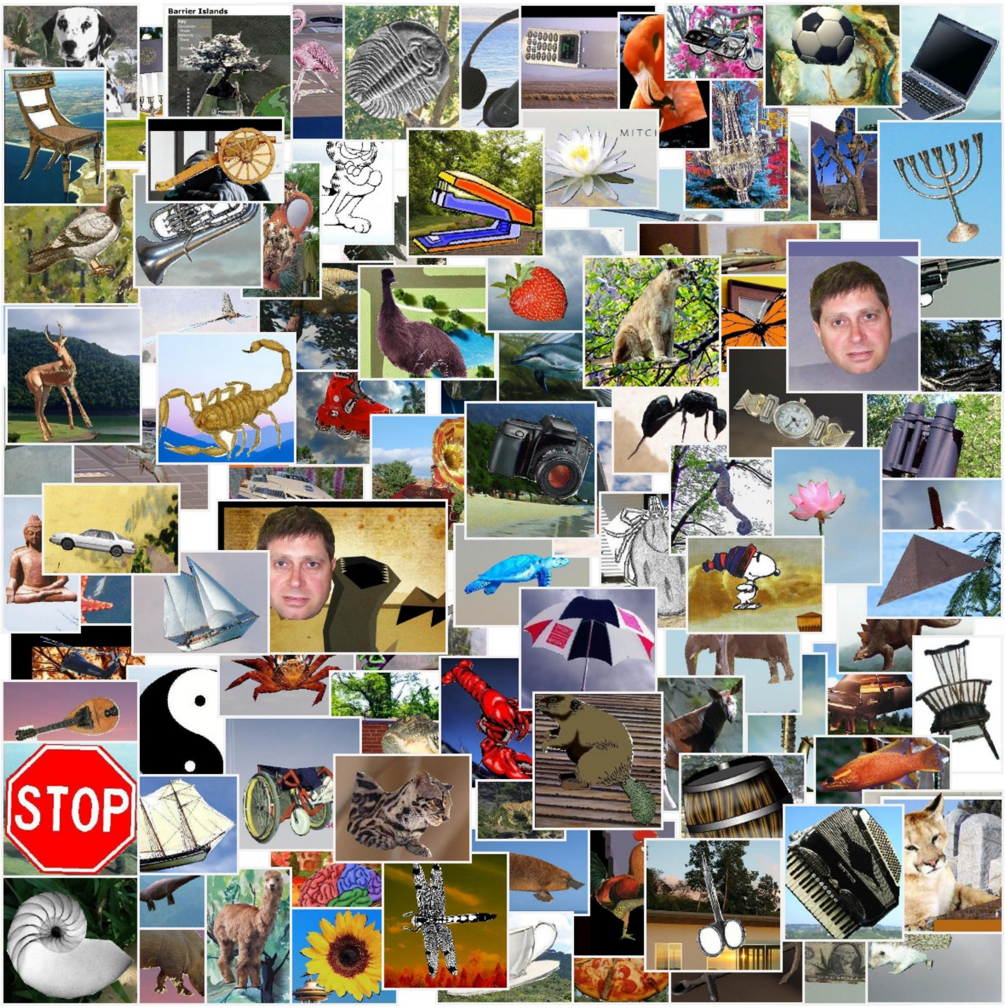
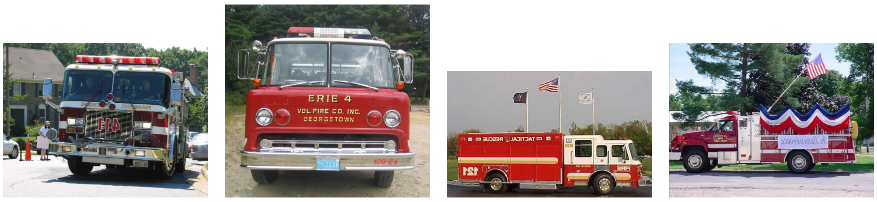


Figure 2.5: An image collage of the images from Randomised Caltech-101 [52] image set. One image from each object category is selected to the collage.

2.1.3 Caltech-256

Caltech-256 [39] can be considered as an extension of Caltech-101 [30] because Caltech-256 contains some of the Caltech-101 object categories and many new categories. Caltech-256 is improved by removing all the overlapping categories which were a problem in Caltech-101 and the number of images per category is increased significantly in Caltech-256. Moreover, the quality of the images is better because of higher resolution. However, in many object categories, the images are captured from various viewpoints which makes the objects visually more dissimilar and the categorisation task more difficult. Thus some of the predefined categories can actually have sub-categories, e.g. *fire-truck* could be divided in the *fire-truck side* and *fire-truck front* sub-categories. In the supervised learning, it is not a critical issue if the images from both sub-categories are included in the training set, but in unsupervised learning, however, it is rather likely that these two categories are separated because they do not share the same visual features, except their red colour (which is also shared with many other categories), and naturally the appearance of the fire-trucks is quite different depending on the viewing point. If one has seen only the front and side views of the fire-trucks and no images from an angle where both the front ant side of the fire-truck can be seen, it can be difficult to connect these images together. This issue is illustrated in Fig. 2.6, where four images of fire-trucks are shown from Caltech-256 [39]. Two of them are front-views and two of them are side-views.



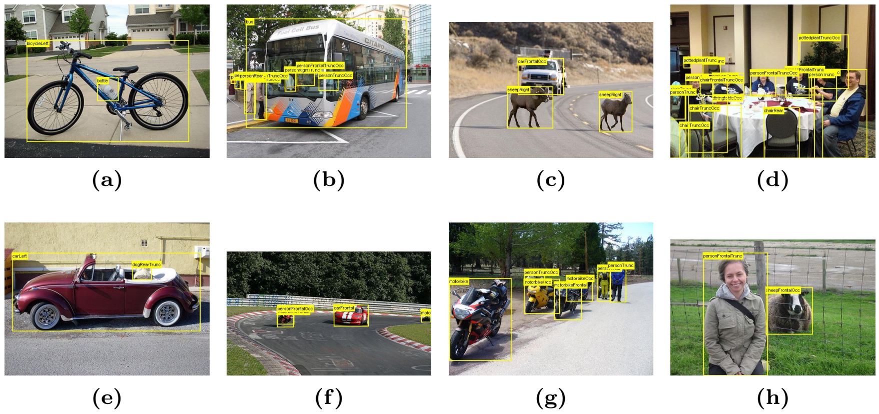
(a) (b) (c) (d)

Figure 2.6: Example images of fire-trucks from Caltech-256 dataset [39].

2.1.4 Pascal VOC competition image sets

Annual Pascal VOC competitions have encouraged research groups to exploit all possible ways to improve the performance of their method. The images in the annual Pascal VOC competitions [22, 29, 23, 26, 27, 24, 25] can contain objects from many different object categories in a single image as shown in Fig. 2.7.

In the classification challenge of the Pascal VOC competition, the task is to predict if an object from class X is in the image or not. By using unsupervised learning, one can assign each image only to one category (or if an image is first segmented then it would be possible to assign each segment into different category). Thus, the Pascal VOC datasets are not suitable for this thesis. Additionally, the number of categories is quite limited, even though the number of images is large.



(a) (b) (c) (d)

(e) (f) (g) (h)

Figure 2.7: Example images from Pascal VOC 2011 dataset [25]: (a) Bottle; (b) Bus; (c) Car; (d) Chair; (e) Dog; (f) Motorbike; (g) Person; (h) Sheep.

2.1.5 Abstract image set

As one of the contributions of the thesis, an Abstract image set was collected to acquire low-level information about how people recognise and categorise images. Since the images contain only abstract art, it is difficult to give unambiguous labels for every image. This makes labelling the image set into specific categories difficult, and thus, instead of labelling images into specific categories, a category tree was built, based on pairwise similarities that were captured from image stack assignments made by the participants.

The abstract images were downloaded from various sources of artistic images, *e.g.*, `digitalart.org`, `caedes.net` and `sxc.hu`. The images were selected from categories labelled as *abstract* and *surreal*. An initial set of 250 color images was selected based on visual quality and content. The images were supposed to be visually complex, with an abstract content, but also images with semi-representative content, such as 3-D modelled images, were included in order to make the test image collection varied. The final set of 100 images were selected from the initial set at random. The selected images are shown in Fig. 2.8.

Following this, the construction of image-wise ground truth for visual saliency and class hierarchy is explained. 24 university students without a background in art and with normal vision were selected for our saliency experiment, and 20 students for the categorisation experiment. The visual saliency experiment was a free-viewing experiment, where every image was shown for five seconds to each participant and an eye-tracker was used to record the eye movements. Attention maps were generated by following the approach by Judd et al. [46]. At first, fixation points were captured from the eye tracking data. These fixation points were used to generate a visual attention map for each image and for each participant separately. Next, the ground truth attention map for an image was constructed by summing up the visual attention maps of each participant for that image.

The category tree was built from the categorisation results made by human participants. The tree is illustrated in Fig. 2.9. In the categorisation experiment, 20 participants

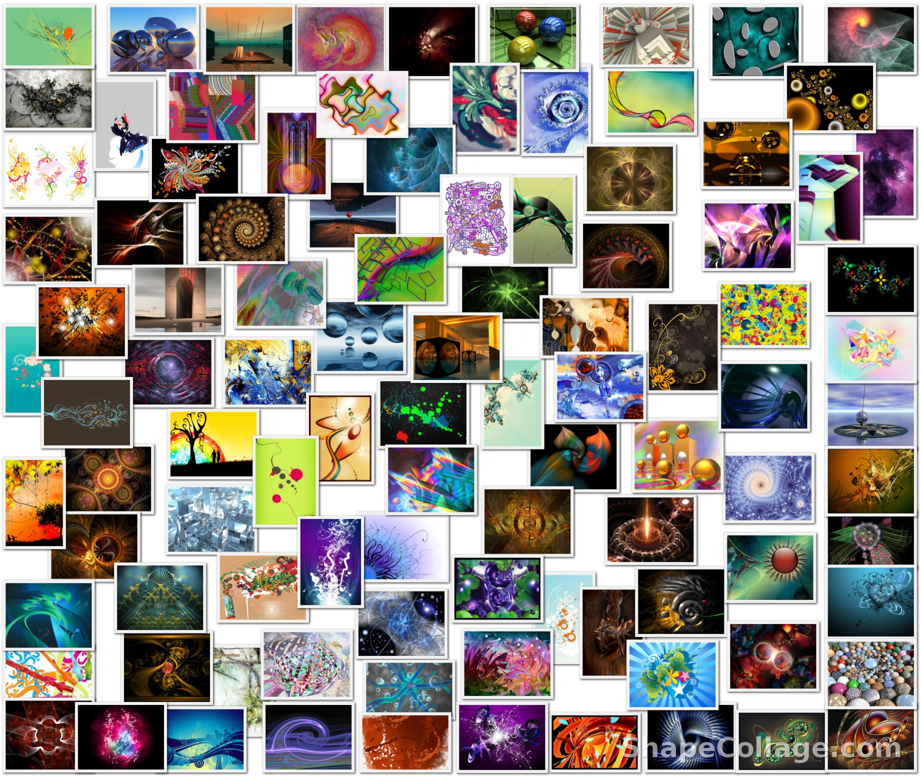


Figure 2.8: An Abstract image set.

were asked to categorise the images into stacks. The number of stacks and the stack assignments were decided by each participant. Co-occurrence of images in the same stacks was used to form the visual categorisation ground truth which represents the average participant opinion. The cluster hierarchy was built using agglomerative clustering with an average distance rule to connect image clusters together. A distance matrix that is used to build the hierarchy is complement of the similarity matrix and the similarity matrix was computed from image stack assignments that were performed by human subjects. The similarity between images i and j is the number of times i and j have been assigned into the same stack divided by the number of human subjects and thus the value is between 0 and 1.

2.2 Performance evaluation

To measure the performance of a supervised or an unsupervised VOC method, one needs to have a performance evaluation method. In this section, different methods are described for evaluating the performance of a supervised and an unsupervised VOC method. The methods described require an image set with the ground truth (correct labels) for evaluating the performance. For this purpose, the standard benchmark data sets described in the previous section can be used, but problems still remain for the evaluation of an UVOC

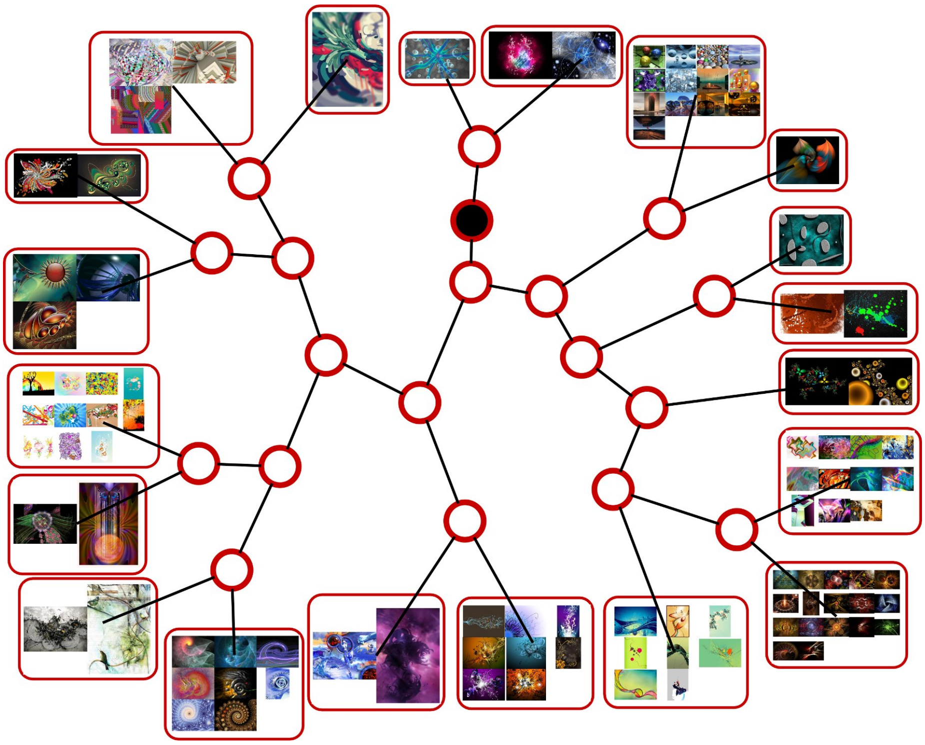


Figure 2.9: Ground truth category tree with 20 leave nodes.

method. For example, how to compare categorisation results with a different number of discovered categories? These issues have been discussed in the works by Tuytelaars et al. [92] and Sivic et al. [83]. They applied measures used to evaluate and compare clustering methods. In the following, these two works are reviewed and, in addition, an alternative method is introduced.

In this thesis, the following performance measure is used for the supervised VOC evaluation:

$$perf_{voc} = \frac{1}{N_c} \sum_{i=1}^{N_c} \left(\frac{|X_i \cap Y_i|}{|X_i|} \right) \quad , \tag{2.1}$$

where N_c is the number of ground truth classes, X_i is a list of images belonging to class i and Y_i is a list of images assigned (predicted) to class i and $|\cdot|$ denotes the number of images. Thus, the classification accuracy for class i becomes the number of images correctly classified to class i divided by the total number of images belonging to class i. Therefore, the final performance is the mean classification accuracy over the classes.

The performance evaluation method presented above cannot be used in the UVOC because the number of predicted categories can be different from the ground truth and also the order of predicted categories can be any. Thus, it is not possible to compute the performance UVOC using the same evaluation method. Next, a few methods for evaluating the performance of a UVOC method are presented.

For the UVOC method evaluation, Sivic et al. [83] proposed a performance evaluation method which takes a "categorisation tree" representing the class hierarchy as the input, and computes its performance to represent the true hierarchy. The evaluation protocol utilises the concept of a hierarchy, i.e., the categories near the root are more mixed than the leaf nodes, which should ideally represent the pure categories. This performance evaluation method cannot be compared with a typical average accuracy measure that is used in supervised learning because their method is more sensitive to errors. In this method, every mistake causes a double error: at first the image is assigned into a wrong node, and also the performance of the other node decreases because there is one image from an incorrect category. The performance of a single node, $P(t,i)$, is computed as

$$P(t,i) = \frac{|X_i \cap Y_t|}{|X_i \cup Y_t|} \ , \tag{2.2}$$

where X_i are the ground truth images for category i and Y_t are the images assigned to node t. Thus, the equation computes how many of the images are assigned from a same category to a specific node and divides it by the number of images assigned to the node + the number of images belonging to the category − the number of images assigned from the specific category to the node. The average performance, $perf_{uvoc}$, is computed as

$$perf_{uvoc} = \frac{1}{N_c} \sum_{i=1}^{N_c} \max_t P(t,i) \ , \tag{2.3}$$

where N_c is the number of categories. The method ultimately chooses nodes $P(t,i)$ that give the best categorisation performance per each object category, and then it computes the average over these nodes. The main drawback of this method is that it actually measures the hierarchical decomposition rather than the categorisation performance. It is not clear whether the hierarchy decomposition is relevant to the categorisation task, or whether it is a problem of its own. For example, if the objects of the same category are separated at the upper levels in the hierarchy, it is heavily penalised even though they would finally appear as two pure leaf nodes.

Tuytelaars et al. [92] adopted their evaluation strategies from the clustering evaluation literature, i.e. how well the produced clusters can map the data to their true labels. They also noticed two possible cases in the method evaluation: 1) the number of categories is enforced to correspond to the number of ground truth categories and 2) the number of produced categories does not correspond to the number of categories in the original data. For the first case, two simple measures can be used. The first one, "purity", is computed as follows:

$$purity(\boldsymbol{X} \mid \boldsymbol{Y}) = \sum_{y \in \boldsymbol{Y}} p(y) \ \max_{x \in \boldsymbol{X}} p(x \mid y) \ , \tag{2.4}$$

where $\boldsymbol{X}$ stands for the ground truth category labels and $\boldsymbol{Y}$ stands for the cluster labels. In practise, $p(x \mid y)$ is empirically computed from the ground truth label frequencies in each category. Purity measures how well a method separates the images of one category from all other categories. The second measure is the mutual information (information gain) which is used also in decision tree learning algorithms:

$$I(\boldsymbol{X} \mid \boldsymbol{Y}) = H(\boldsymbol{X}) - H(\boldsymbol{X} \mid \boldsymbol{Y}) \ , \tag{2.5}$$

which is based on the original entropy of an image set $H(\boldsymbol{X})$ and the entropy after the categorisation, the conditional entropy $H(\boldsymbol{X} \mid \boldsymbol{Y})$. The conditional entropy is computed as:

$$H(\boldsymbol{X} \mid \boldsymbol{Y}) = \sum_{y \in \boldsymbol{Y}} p(y) \sum_{x \in X} p(x \mid y) \log \frac{1}{p(x \mid y)} \ . \tag{2.6}$$

Conditional entropy measures how certain one can be that the image actually belongs to the cluster. However, since the term $H(\boldsymbol{X})$ is constant in (2.5), the conditional entropy in (2.6) can be directly used. When the number of clusters increases considerably, the conditional entropy and purity give ideal results. The main problem of these measures is that they can be used for the method comparison only when all methods return the same number of categories. Moreover, the values depend on the total number of images. The main drawback, however, is the limitation that neither of the methods estimate the categorisation accuracy well if the estimated number of categories is not the same as in the ground truth and especially if $|\boldsymbol{Y}| > |\boldsymbol{X}|$. In extreme cases, every image is its own category, and this produces the perfect performance values $purity = 1$ and $H(\boldsymbol{X} \mid \boldsymbol{Y}) = 0$. Tuytelaars et al. [92] circumvented this undesirable property by introducing an “oracle”, which means that they separated the training and test sets to discover the classes, and to test the formed classes with the test data. In this case, on the other hand, one could compute the number of correctly categorised images for each category and compute categorisation accuracy for each category and then take mean over the categories to obtain categorisation performance. This performance evaluation method could be more intuitive than the conditional entropy, but it needs to have separate training and testing set images. Thus, it is not used in this work.

2.3 Summary

Many datasets and benchmarks as well as methods for evaluating the performance of VOC and UVOC methods, have been introduced during the past few years. In this chapter, the most popular benchmarks were discussed and their characteristics were listed and compared with each other.

In this summary, all the presented image sets are summarised and briefly compared. The major differences between the image sets are the number of images in the image set, the number of categories, existence of 2-D and 3-D transformations and number of objects in each image. The differences between the image sets are summarised in Table 2.1.

The suitable benchmarks for the UVOC development are Caltech-101 [30], Randomised Caltech-101 [52] and Caltech-256 [39]. Pascal VOC benchmarks are not suitable because one image can contain objects from many categories, and thus, each image should be

Table 2.1: Comparison between the imagesets.

Name	Images	Categories	2-D t.	3-D t.	Objects/image
Caltech-101 [30]	8,677	101	-	-	1
r-Caltech-101 [52]	8,677	101	+	-	1
Caltech-256 [39]	29,782	256	+	+	1
Pascal VOC 2005 [22]	1,578	4	+	+	1-N
Pascal VOC 2006 [29]	2,618	10	+	+	1-N
Pascal VOC 2007 [23]	9,963	20	+	+	1-N
Pascal VOC 2008 [26]	4,340	20	+	+	1-N
Pascal VOC 2009 [27]	7,054	20	+	+	1-N
Pascal VOC 2010 [24]	10,103	20	+	+	1-N
Pascal VOC 2011 [25]	11,530	20	+	+	1-N

categorised in many categories. This is not possible without dividing the images into segments, and categorising each segment separately. Thus, Pascal VOC benchmark image sets are excluded from the thesis.

Moreover, a few methods for evaluating the performance of VOC and UVOC methods were also discussed. For the supervised VOC, only one method was introduced which is a simple average classification accuracy over the classes. However, for UVOC there are a few options: i) A method based on conditional entropy (2.6) introduced by Tuytelaars et al. [92] and ii) Mean performance (2.3) introduced by Sivic et al. [83]. Both of the methods have their strengths and weaknesses. Thus, as it is not obvious which method to use, both methods are used to evaluate the performance of the UVOC methods.

Chapter III

Bag-of-Features approach for unsupervised visual object categorisation

This chapter presents a method for Unsupervised Visual Object Categorisation (UVOC) using the Bag-of-Features (BoF) approach [84, 14]. In BoF, there are many steps and in each step, there are many alternative methods that can be used for the same task. Naturally, different methods have different properties. Thus this chapter gives an overview of different methods and the performances of different methods are compared with each other in the experiments section. Finally, a set of methods is chosen to be used in UVOC.

Csurka et al. [14] demonstrated how visual object categories can be learned using local features, clustering and supervised learning using the BoF approach. Their work has inspired many others to use BoF in their VOC [28, 36, 96] and UVOC [50, 92] studies. The BoF approach in UVOC is illustrated in Fig. 3.1, where given images are categorised using unsupervised learning. As in BoF, the first step for the supervised VOC is a local feature detection where important local features are detected. Subsequently, these detected local features are described by using a local feature descriptor. A codebook is constructed using cluster centroids produced by a clustering method or using SOM [53] node vectors as in our case. After this, the given image is described by matching extracted local features with the codebook and computing frequency of how many times each code has a match. The differences between the unsupervised and supervised (see Fig. 1.2) VOC based on BoF are that the final step is typically clustering instead of classification. Moreover, in the unsupervised VOC there is no training data that could be used to enhance the code selection in the codebook building as is introduced by Leibe et al. [58].

3.1 Local feature extraction

In this section, a few of the most popular local feature extraction methods are discussed and their characteristics are summarised at the end of the chapter. In the local feature extraction, local features are detected using a local feature detector and then described using a local feature descriptor. Thus, the result of local feature extraction is the spatial location of the detected region and description of the region. The local feature extraction is one of the key elements in visual object categorisation using the BoF approach because the descriptions of the detected regions are used to describe the appearance of the image. The selection of the local feature extractor has significant impact on categorisation accuracy [70, 68, 64].

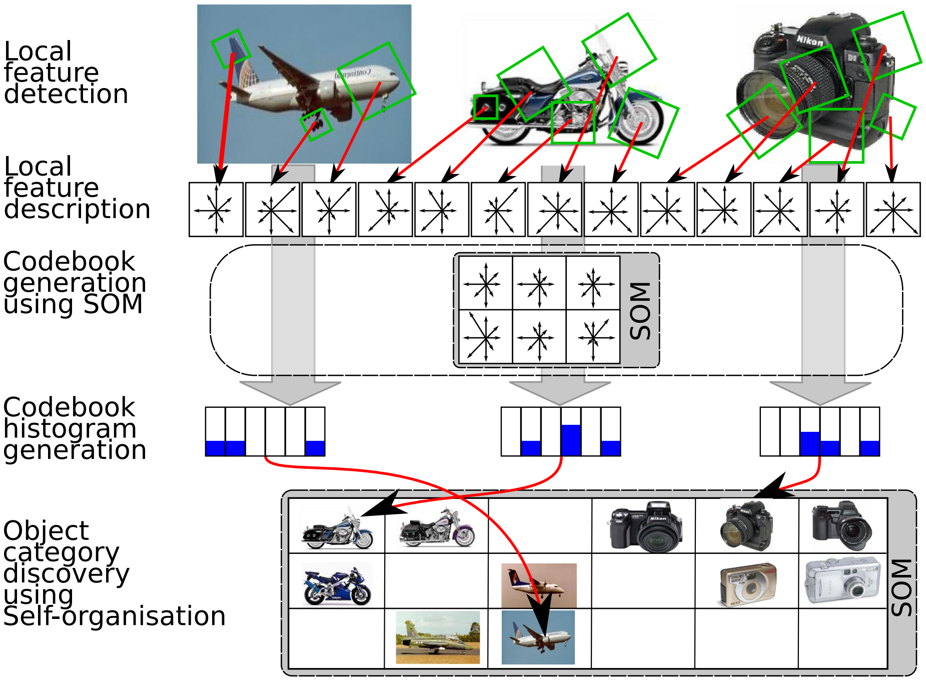

Local feature detection
Local feature description
Codebook generation using SOM
SOM
Codebook histogram generation
Object category discovery using Self-organisation
SOM

Figure 3.1: Bag-of-Features approach applied in UVOC. In the first row, detected local features are drawn with green rectangles. In the second row, detected local features are described by computing the gradients in 8 directions which are illustrated with arrows. In the third row, visual vocabulary is built by using a Self-Organising Map. In the fourth row, codebook histograms are shown. In the fifth row, images are categorised using Self-Organisation.

As mentioned earlier, local feature extraction consist of two steps: local feature detection (i.e. interest point detection) and local feature description (i.e. key-point description) [60]. In the first step, important regions are detected from an input image which are then described using a descriptor in the second step. The output of local feature extraction is a combination of spatial location information (x, y, scale (or scale-u, scale-v if an affine detector is used) and orientation) and region description of the appearance of the detected region [70].

A number of local feature detectors and descriptors have been proposed in the literature. A survey and comparison of different detectors can be found in the work by Mikolajczyk et al. [70] and for the descriptors in [68]. These comparisons, however, are based on the repeatability and matching performances over different views of the same scenes. Therefore, their applicability to VOC and UVOC is unclear. More explicit VOC evaluations have been carried out by Zhang et al. [103] and Mikolajczyk et al. [64]. Their main conclusions were that detector combinations performed better than any single detector, and that the extended versions of Scale Invariant Feature Transform (SIFT) [60] descriptor, the Gradient Location and Orientation Histogram (GLOH) [68], is slightly superior to others in VOC. The better performance using the detector combinations can also be explained by the increased number of detected features. The drawback of GLOH is that it requires training data to estimate eigenvectors for the required PCA dimensionality reduction step – proper selection of the PCA data can explain the slightly better performance compared with the original SIFT. Based on the above works, SIFT can be safely used as the descriptor, but it is justified to investigate which detector is the most suitable for UVOC.

3.1.1 Local feature detectors

In the local feature detection, a local feature detector detects regions from the image that are considered important and stable across the images of the same object on an object category. A good local feature detector should detect the same parts of the visual objects in different images. For example, if multiple images are taken of the side of a car, it should detect the same regions in every image, for example wheels, mirrors, etc. Detected regions should be invariant to scale, rotation and preferably to affine transformations so that they can be described accurately with a local feature descriptor. Otherwise, it is difficult to find similar regions from images. The best performing local feature detector detects a great number of regions to guarantee that the same regions can be found from the other images as well [70, 93]. The local feature detectors that are used in the thesis are presented next.

The SIFT local feature extractor method [60] contains a region detector and a region descriptor. Here, only the local feature detector is considered and the descriptor part is discussed in Sec. 3.1.2. The SIFT local feature detector is based on Difference-of-Gaussian (DoG) operator. Local features are detected from a scale-space that is built by subsequently smoothing and resampling the input image $\boldsymbol{I}$ with a Gaussian function $g(x, y, \sigma)$. The input image is smoothed with a Gaussian filter as follows:

$$\boldsymbol{G}(x, y, \sigma) = g(x, y, \sigma) * \boldsymbol{I}(x, y) \ , \tag{3.1}$$

where $\boldsymbol{G}$ is the smoothed image and $*$ is the convolution operation. Local features are searched from local extremes of Difference-of-Gaussian images $\boldsymbol{G}$, which is defined as subtraction of two images smoothed with different σ values as follows:

$$\Delta\boldsymbol{G} = \boldsymbol{G}(x, y, k\sigma) - \boldsymbol{G}(x, y, \sigma) \ , \tag{3.2}$$

where k is a factor > 1. This is repeated with many k values in order to obtain an octave of DoG images $\Delta\boldsymbol{G}$. Scale invariance is achieved by downsampling the input image and then computing another octave for the smaller image. Typically, the downsampled version is half of the size of the previous image. Thus, it takes less time to compute octaves for the smaller images. Locations of the local features are searched from each octave by selecting local extreme values from a $9 \times 8 \times 9$ neighbourhood. If the value of the pixel is the highest or the lowest in its neighbourhood, it will be selected as a candidate point. Next, candidate points, or actually regions, that do not have enough contrast will be removed. Orientation of the local feature is assigned based on the dominant gradient in the detected region. If the magnitude of the second most highest gradient is nearly equal ($>$80% [60]) then two local features with different orientations will be detected. This is done to guarantee that the region will be correctly detected even though it will also add a false match.

The Harris-Laplace detector [65, 70, 93] is based on the Harris corner detector [40] to detect spatial locations of the local features. The Harris corner detector is based on the *second order moment matrix.* The second order moment matrix for the neighbourhood of pixel $\boldsymbol{x} = (x, y)$ is defined as follows:

$$\boldsymbol{M}(\boldsymbol{x}, \sigma_I, \sigma_D) = \sigma_D^2\, g(\sigma_I) * \begin{bmatrix} \boldsymbol{I}_x^2(\boldsymbol{x}, \sigma_D) & \boldsymbol{I}_x(\boldsymbol{x}, \sigma_D)\boldsymbol{I}_y(\boldsymbol{x}, \sigma_D) \\ \boldsymbol{I}_x(\boldsymbol{x}, \sigma_D)\boldsymbol{I}_y(\boldsymbol{x}, \sigma_D) & \boldsymbol{I}_y^2(\boldsymbol{x}, \sigma_D) \end{bmatrix} \ , \tag{3.3}$$

where σ_I is a Gaussian kernels size (integration scale) and σ_D is a Gaussian kernel size (differentiation scale), The Gaussian $g(\sigma)$ and the image gradient $\boldsymbol{I}_x(\boldsymbol{x}, \sigma_D)$ are defined as follow:

$$\boldsymbol{I}_x(\boldsymbol{x}, \sigma_D) = \frac{\partial}{\partial x} g(\sigma_D) * \boldsymbol{I}(\boldsymbol{x}) \ , \tag{3.4}$$

$$g(\sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \ . \tag{3.5}$$

The pixel $\boldsymbol{x}$ is chosen as a local feature, i.e., corner, if the cornerness R is above a *threshold.* The cornerness is defined as:

$$R = det(\boldsymbol{M}) - \beta \ \ trace^2(\boldsymbol{M}) > threshold \ , \tag{3.6}$$

where β is a parameter that is given by the user, and *threshold* is a predefined threshold for accepting points that are defined to be corners [66, 93]. These corners are detected in many scales, by filtering the image with a Gaussian filter of various sizes. Then the scale of the regions is detected from Gaussian scale-space by maximising Laplacian-of-Gaussian over the scale space by finding local extreme values of the determinant of $\boldsymbol{M}$. The Gaussian scale-space is built by filtering the input image with a Gaussian kernel of different sizes. Orientation of the local feature is defined in the description phase, where gradients are computed from the (resampled) detected region and the dominative gradient is used to define the orientation of the detected feature.

The Harris-Affine detector is an affine invariant version of the Harris-Laplace detector. It uses Harris-Laplace to detect spatial locations and scales of the local feature and then parameters of the detected regions are refined using an iterative algorithm. In the iteration phase, shape of the detected region is transformed from ellipse to circular based on the second moment matrix, in the way that magnitudes of the moments are equalised. [66, 67]

The Hessian-Laplace detector [70] is similar to the Harris-Laplace detector, but instead of using Harris corner detector to detect spatial locations of the local features, a Hessian matrix is used which is defined as follows:

$$\boldsymbol{H}(\boldsymbol{x}, \sigma_D) = \begin{bmatrix} \boldsymbol{I}_{xx}(\boldsymbol{x}, \sigma_D) & \boldsymbol{I}_{xy}(\boldsymbol{x}, \sigma_D) \\ \boldsymbol{I}_{xy}(\boldsymbol{x}, \sigma_D) & \boldsymbol{I}_{yy}(\boldsymbol{x}, \sigma_D) \end{bmatrix} \ , \tag{3.7}$$

where $I_{xx}(\boldsymbol{x})$ is the second order partial derivative in the direction x, I_{yy} is the second order partial derivative for direction y and I_{xy} the second order partial derivative for x and y directions for a Gaussian with σ_D kernel size smoothed image $\boldsymbol{I}$. Local maximum of the determinant is used to locate blobs from the input images [70]. The scale of the detected region is defined, as in Harris-Laplace, by choosing the location and scale where the determinant of the Hessian-matrix gets local maximal values [67].

The Hessian-Affine detector [67, 70] is an extension to the Hessian-Laplace detector that is also invariant against affine transformations. The Hessian-Affine detector is similar to the Harris-Affine detector, but instead of the Harris corner detector, the Hessian matrix is used to define initial locations of the local features. However, Hessian-based detectors give strong responses to blobs and ridges in the image and Harris corner detectors give strong responses to a corner like shapes in the image.

Maximally Stable Extremal Regions (MSER) introduced by Matas et al. [63] differs significantly from the previously presented local feature detectors. Instead of finding regions where the change is maximal, such as a corner, MSER method finds regions where the change is minimal. Regions are found by thresholding the image with increasing threshold and using connected components analysis to connect pixels that are above the threshold. Detected regions are typically converted into ellipses before description, and thus, a significant amount of information is lost in respect to invariance against the rotation and scale of the region.

The SURF local feature extraction method [4] is basically a faster version of SIFT. It contains a local feature detector and descriptor. Here, only the detector part is considered and the descriptor is discussed in Sec. 3.1.2. SURF is a very fast local feature detector (and descriptor). The SURF detector uses an approximation of the determinant of the Hessian matrix, which is computed by using integral images to speed up the process. Then it seeks the maximum values for the determinant of the Hessian matrices. The smoothing Gaussians are also approximated by using box filters – this can be done in constant time without downsampling the original image by increasing the box filter size to acquire local features in many scales. Since the size of the filter is increased, there is no need for computing the integral images again. Orientations of the detected regions are computed using 2-D Haar wavelet filters on the x and y axes. The orientation is chosen from the most dominative response. [4]

Dense sampling extracts local features uniformly using a pre-defined spatial grid, scale and orientation. This is a very simple approach to the local feature detection, but in the comparisons, this approach has produced results superior to the local feature driven methods [26]. Since then, a hybrid method has also been proposed [91]. In the hybrid method, the initial locations of the local features are set by generating grids of different sizes to capture local features of different scales. Then, the location of each local feature is re-defined by choosing a more stable location from its neighbourhood. The topology is kept, by not letting the neighbourhoods overlay. Even though this seems to be a good idea, the results achieved in [91] do not promise any improvement in VOC.

Example images of detected regions are shown in Fig. 3.2. The performance of different local feature detectors is evaluated in Sec. 3.5.2. Properties of the discussed local feature detectors are summarised in Table 3.1.

3.1.2 Region descriptors

In this section, region descriptors are discussed. In the region description step, a detected region is described in a way that allows regions to be compared with each other and

Table 3.1: Invariance of local feature detector against different transformations.

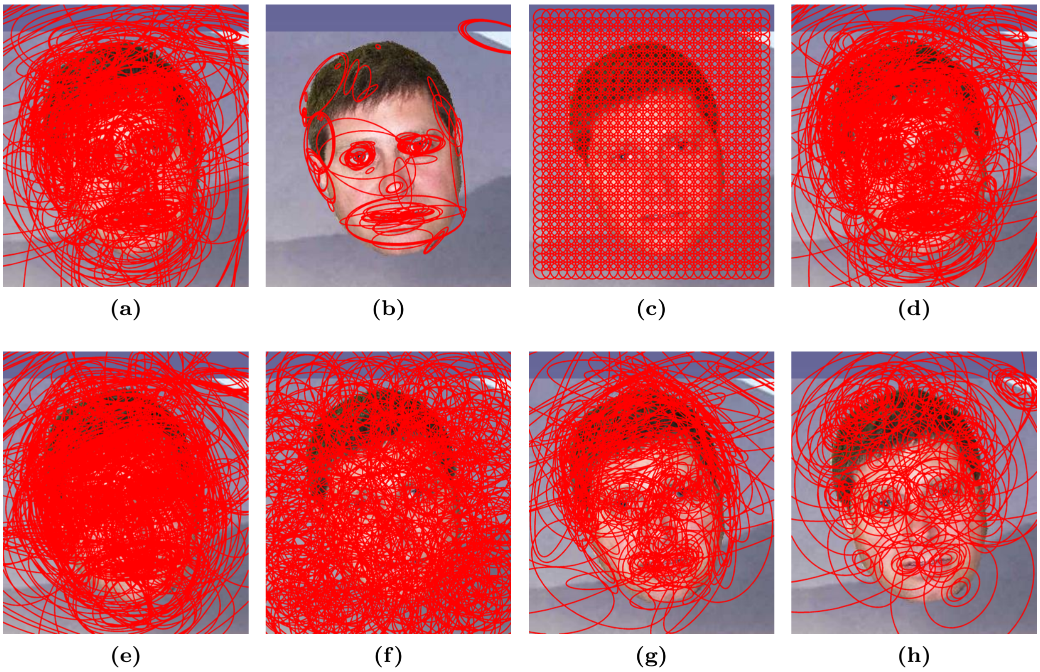
Detector	**Rotation inv.**	**Scale inv.**	**Affine inv.**
Dense sampling	-	-	-
Difference-of-Gaussian	+	+	-
Hessian-Laplace	+	+	-
Harris-Laplace	+	+	-
Hessian-Affine	+	+	+
Harris-Affine	+	+	+
SURF	+	+	-
MSER	+	+	+

similar regions can be found. Typically, a region descriptor is invariant against illumination [58, 60, 68, 95] and against rotation [60, 68]. There are many methods for describing the detected regions such as, Gradient Location Orientation Histogram (GLOH) [68], Histogram of Oriented Gradients (HOG) [15], Local Binary Pattern (LBP) [74], SIFT [60], Speeded Up Robust Features (SURF) [4], thus only a few of them are described here.

Scale Invariant Feature Transform (SIFT) [60] is a combination of an interest point detector and a descriptor. Here, only the descriptor part is discussed because the detector was discussed in the detector section. The SIFT descriptor is very widely used in many state-of-the-art studies in visual object recognition. In the original paper, and in most of the other studies as well, the region is sampled using a 4×4 grid and for every cell in the grid, a gradient is computed in eight directions. Magnitudes of these gradients are then used as a descriptor resulting a $4 \times 4 \times 8 = 128$ dimensional vector. The descriptor is also normalised to acquire better invariance against illumination changes.

The SURF descriptor is similar to the SIFT descriptor with only a few differences. The detected region is divided into 4×4 grid and each cell is sampled with 5×5 points, but instead of computing the gradients from the sampled points, 2-D Haar wavelets are used to produce the sums of dx, dy, $|dx|$ and $|dy|$, where dx is gradient in x direction, dy in y direction and $|dx|$ and $|dy|$ are their absolute values. Finally, the descriptor is obtained by concatenating the sums (dx, dy, $|dx|$ and $|dy|$) together to form a vector $\boldsymbol{d}$ and normalising it by dividing it with its length $|\boldsymbol{d}|$ as follows: $\hat{\boldsymbol{d}} = \boldsymbol{d}/|\boldsymbol{d}|$. The SURF descriptor is typically 64 dimensional, only a half of the dimensionality of the SIFT descriptor, however in the experiments made by Bay et al. it was shown that the SURF descriptor is at least as distinctive as the SIFT descriptor. [4]

GLOH developed by Mikolajczyk and Schmid [68] is an extension to the SIFT descriptor. The descriptor is computed in 17 locations in log polar coordinates and gradients are computed in 16 directions which forms a vector with $17 \times 16 = 272$ dimensions. The length of the vector is reduced to 128 using PCA whose largest eigenvectors have been estimated beforehand from 47,000 image patches.


(a) (b) (c) (d)

(e) (f) (g) (h)

Figure 3.2: Detected interest regions by using several methods and their different implementations: (a) Harris-Affine (FS); (b) MSER (FS); (c) Dense sampling; (d) Harris-Laplace (FS); (e) Hessian-Affine (FS); (f) DoG, i.e. SIFT (LV); (g) Harris-Laplace (LV); (h) Hessian-Laplace, basically Hessian-Affine, (LV). FS: implementation from the Feature Space web-site [69]; LV: implementation from the Lip-Vireo web-site [104].

3.1.3 Colour information

All the detectors and descriptors discussed earlier use only grey-level information. Van de Sande et al. [95] have studied different approaches to the use of colour information in the SIFT descriptor. Their final result is that colour information can improve category recognition up to 8%, but the usefulness of colour information is category dependent, and thus, it is problematic in UVOC, because the object categories are unknown. The Colour-SIFT descriptor also makes descriptors more distinctive (the original SIFT descriptor is 128 × 1, whereas the Colour-SIFT descriptor is 128 × 3) which increases the needed computation. Van de Sande et al. used the Harris-Laplace detector to choose the regions for Colour-SIFT descriptor extraction.

3.1.4 Local feature filtering

One of the problems in the BoF approach is that in the feature generation step a large number of false matches are made, especially if the size of the codebook is small. When the size of the codebook is small, two very dissimilar local features can be matched to the same code. It is also common that a region detector detects most of the local features from the background (see Fig. 3.2), and thus, a codebook histogram represents mostly the background of the image instead of the object itself.

One approach to solving this problem is to filter out all local features that come from the background. If it is assumed that the background varies more than the foreground, then one can try to find a set of common local features and use only these because the common features should be extracted from the foreground and infrequent local features can be assumed as being extracted from the background.

Feature matching

To filter out uncommon features, we need to have a threshold for local feature matching that we use to define whether two local features match with each other or not. In the specific object detection method introduced by Lowe [60], a match of two local features was accepted if the distance between the features was less than 1.5 times the distance between the second best match. Lowe also claimed that it is not possible to set a single threshold. However, we need to have a single global threshold that we can use to match any pair of local features. In our case, the threshold does not need to be perfect and we are doing object category detection instead of specific object detection as Lowe did. In our case, we are satisfied if we can filter out many local features from the background and keep most of the local features detected from the foreground. The problem of selecting the optimal threshold is adhered to in the experiment presented in Sec. 3.5.3.

The algorithm (see Algorithm 3.1) to find a common set of local features is defined as follows: i) Extract local features $\boldsymbol{D} = \{\boldsymbol{d}_1, \ldots, \boldsymbol{d}_{numOfFeatures}\}$; ii) Acquire a list of common local features, $\boldsymbol{D_c}$, by comparing all the local features in the training set against each other. To accept a match, the distance between the local feature and the common set of local features can be at most $maxDist$; If the local feature does not match with a current set of common local features, it will be added to the list of common local features and counter, $\boldsymbol{hitCounter}$, for the new common local feature is initialised and the number of common local features, n, will be increased by one. If the local feature matches the one of the common local features, the counter of the matching common local feature will be increased by one. iii) Ascertain the number of hits for each common local feature; iv) Choose the common local features which have more than $minHits$ matches; Sample images about filtering are shown in Fig. 3.3.

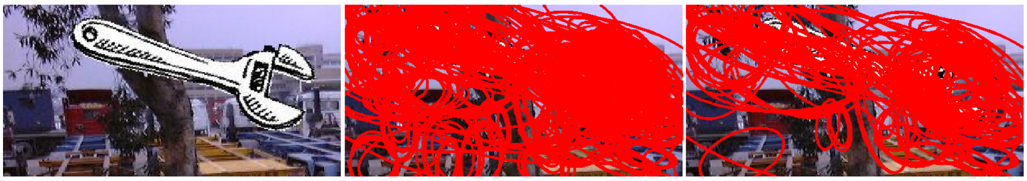


Figure 3.3: Local feature filtering sample outputs using Randomised Caltech-101: Left: Original image (wrench); Middle: All the extracted Hessian-Affine local features; Right: Local features after filtering uncommon features out.

Fig. 3.3 shows that many local features are kept, even though $minHits$ is large. This is explained by the fact that the $maxDist$ is quite high, and thus, many of the local

Algorithm 3.1 Find a set of common local features

```
Require: $\boldsymbol{D} = \{\boldsymbol{d}_1, \ldots, \boldsymbol{d}_{numOfFeatures}\}$, $maxDist$, $minHits$
  $\boldsymbol{commonLf}_1 \leftarrow \boldsymbol{d}_1$
  $\boldsymbol{hitCounter}_1 \leftarrow 0$
  $n \leftarrow 1$
  // Find a list of unique local feature descriptors
  for all $\boldsymbol{d}_{1,\ldots,numOfFeatures} \in \boldsymbol{D}$ indexed with $i$ do
    $[distance, bm] = findClosestFeature(\boldsymbol{commonLf}, \boldsymbol{d}_i)$
    if $distance \leqslant maxDist$ then
      $\boldsymbol{hitCounter}_{bm} \leftarrow \boldsymbol{hitCounter}_{bm} + 1$
    else
      $n \leftarrow n + 1$
      $\boldsymbol{commonLf}_n \leftarrow \boldsymbol{d}_i$
      $\boldsymbol{hitCounter}_n \leftarrow 0$
    end if
  end for
  // Filter out local feature descriptors that are not enough popular
  for $i = 1$ to $n$ do
    if $\boldsymbol{hitCounter}_i < minHits$ then
      delete($\boldsymbol{commonLf}_i$)
    end if
  end for
  return $\boldsymbol{commonLf}$
```

features are matched with a local feature from the common local feature set. In many cases, many local features are filtered out that are detected from the background while most of the local features from the foreground are kept. In the experiments section, an experiment is made to verify if this visual interpretation is valid.

3.2 Codebook generation

In the codebook generation step, extracted local features are used to form a codebook which is used to generate BoF histograms to describe the input images. In the original BoF study by Sivic et al. [84] the k-Means clustering algorithm was used to cluster extracted local features and the cluster centroids were used as the codebook. In this section, a few alternative approaches for the codebook generation are discussed.

Even though the k-Means is one of the most popular methods for generating codebooks it is also known to have some weaknesses: for example, the cluster centroids are typically found around high densities in data, and therefore, the input space is not evenly covered. Jurie and Triggs [47] have developed a clustering method which is more robust than the k-Means. Their method avoids setting all cluster centroids into high density areas. Their algorithm first chooses N samples randomly and then computes maximal density of the samples using a mean-shift estimator. Then it assigns a cluster centroid to the maximal density and eliminates all samples that are within a certain radius from the cluster centre. Then the algorithm repeats these steps with the remaining samples as long as there are too many samples left or the number of clusters is too low. When all the clusters are found, Jurie and Triggs had an additional step to find the topological order of the codes. Interestingly, this “topology preserving” enforcement is very similar to the main characteristic of self-organisation. The problem in this method is that one needs to choose a radius.

One of the problems in codebook generation is the difficulty of setting the size of the codebook. Nistér and Stewénius [73] approach to this problem was to use a hierarchical codebook. The codebook was built using hierarchical k-Means, i.e., local feature descriptors were clustered recursively into smaller and smaller clusters. This process defines the hierarchy of the codebook. By using a hierarchical codebook, Nistér and Stewénius were able to generate a large codebook efficiently.

Problem-specific clustering approaches have been developed as well. Leibe et al. [58] developed a method that can learn a model of a visual object class from a set of images from a particular class. However, in the UVOC problem, there are no training or validation sets with manually labelled ground truths which conversely prevents using the most effective enhancements in the codebook generation that are used in the supervised VOC.

One problem-specific enhancement outside clustering is to utilise the spatial information in the codebook generation or probing. For example, Lazebnik et al. [57] reported a method which uses a spatial pyramid to organise descriptors based on their appearance and location.

One family of algorithms for codebook generation are the ones typically used for data visualisation and exploration, such as the Multi-Dimensional Scaling (MDS) [7], Kohonen’s Self-Organising Map (SOM) [53], Isomap [90], and locally linear embedding (LLE) [79]. These methods have similar properties, and therefore, in the thesis the one that can find a topological grouping of data points effectively is selected: the self-organising map and its public implementation, the SOM Toolbox [1]. The self-organising map has been successful compared with the k-Means algorithm in the experiments [50].

3.3 Feature generation and normalisation

In the Bag-of-Features approach, images are described by matching extracted local features with the codebook in the feature generation step that is illustrated in Fig. 3.4. The process of describing the images with codebook histograms in the BoF approach [14] can be described as follows: Let $\boldsymbol{D}$ be a set of local feature descriptors which are detected from an image using a local feature detector such as the Hessian-Affine [66] and described using a local feature descriptor such as SIFT [60], and let $\boldsymbol{CB}$ be a codebook which contains N_{cb} codes. In practice, codes in the $\boldsymbol{CB}$ are clusters’ centroids. Let N_{lf} be the number of local feature descriptors extracted from the image. After this, a BoF histogram $\boldsymbol{f}$ is generated according to the Bag-of-Features approach which is defined in Algorithm 3.2. The $Dist$ function calculates the Euclidean distance between two vectors. The smaller the distance, the greater the similarity is between the two vectors. Hence, a code that minimises the distance from a descriptor is chosen as the best match which has an index of bm.

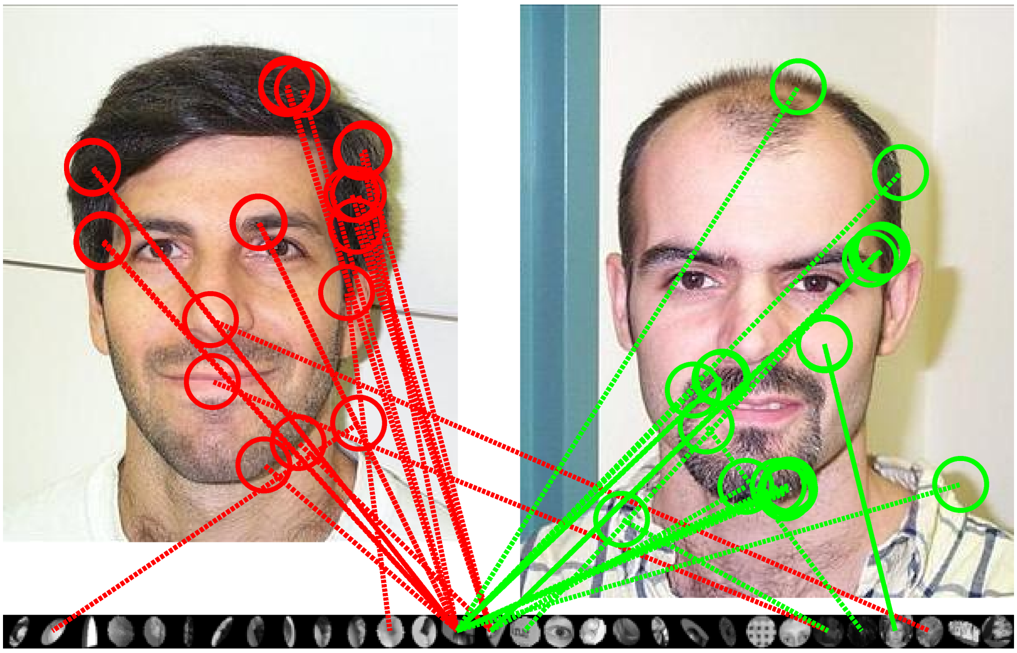


Figure 3.4: Feature generation step of the Bag-of-Features approach illustrated. Circles represent the locations of the detected local features (the scale and orientation is disregarded to make it visually more applicable). The visual codebook is shown at the bottom and the local features are connected to the matching codes with lines.

3.3.1 Normalisation methods

Tuytelaars et al. [92] made a comprehensive study of the effects of normalisation methods in UVOC. Their conclusion was that the L2-norm normalisation produces the best performance followed by the binarised-BoF. Their results show that the normalisation has a significant impact on the categorisation performance, thus it is also an important topic for discussion in this thesis.

In the L1-norm normalisation the feature vector (e.g., BoF histogram) $\boldsymbol{f}$ is divided by its L1-norm (i.e. Manhattan distance):

$$\hat{\boldsymbol{f}} = \frac{\boldsymbol{f}}{|\boldsymbol{f}|} = \frac{\boldsymbol{f}}{\sum_{i=1}^{N_d} |f_i|}, \tag{3.8}$$

where N_d is the number of the dimensions in the feature vector and i is a running index ($i = 1, \ldots, N_d$). In the L1-normalisation, the BoF histogram is divided by the sum of all bins thus the result will be a vector of values between 0 and 1. In the L2-norm normalisation, the feature vector $\boldsymbol{f}$ is divided by its L2-norm (i.e. Euclidean distance) to make it an unit length vector:

Algorithm 3.2 Codebook histogram generation using the Bag-of-Features approach

```
Require: $\boldsymbol{D}$, $\boldsymbol{CB}$
  $N_{cb} \leftarrow length(\boldsymbol{CB})$
  $\boldsymbol{f}_{1,\ldots,N_{cb}} \leftarrow 0$
  $N_{lf} \leftarrow length(\boldsymbol{D})$
  for all $\boldsymbol{d}_i = \boldsymbol{d}_1, \ldots, \boldsymbol{d}_{N_{lf}} \in \boldsymbol{D}$ do
    $bm \leftarrow \arg\min_j \; Dist(\boldsymbol{d}_i, \boldsymbol{CB}_j)$
    $\boldsymbol{f}_{bm} \leftarrow \boldsymbol{f}_{bm} + 1$
  end for
  return $\boldsymbol{f}$
```

$$\hat{\boldsymbol{f}} = \frac{\boldsymbol{f}}{||\boldsymbol{f}||} = \frac{\boldsymbol{f}}{\sqrt{\sum_{i=1}^{N_d} f_i^2}}. \tag{3.9}$$

In the code-wise normalisation, all bins of a certain code are normalised. In the binarised-BoF, the median of occurrences of each code is computed, and all bins below the median are set to zero, and all above to one. By binarising the BoF histograms, the BoF histograms should be more stable because small differences diminish in the normalisation. [92]

In the Term Frequency - Inverse Document Frequency (TF-IDF) normalisation [45], the number of occurrences of a code in an image (Term Frequency) is divided by the number of images containing the code (Inverse Document Frequency). The idea in TF-IDF normalisation is to give more weight to codes that are popular only in a subset of images. TF-IDF has been used successfully in large scale CBIR by Philbin et al. [76].

Dimensionality reduction

The dimensionality of the codebook feature histograms can be decreased by using the traditional method called Principal Component Analysis (PCA) [44]. In the PCA-BoF, the histogram dimensionality is reduced, for example to 20, by the PCA. In the dimensionality reduction using PCA, information is lost, but PCA preserves most of the variation which in the case of BoF histograms means that codebook codes that have more variation gain more weight. [92]

Soft-assignment and accurate matching

Gemert et al. [96] have developed a method based on the k-Means. They replaced the simple learning rule which assigns a sample to the closest cluster, with uncertainty, plausibility and distance values. These values are used in the codebook histogram generation. For example, if a data point is in the middle of two clusters, it will be assigned with the uncertainty of 50% to both clusters.

The idea of soft-assignment is that each local feature is being matched to many codes in the codebook, and thus the local feature can be estimated more accurately by weighting different codes based on distances between the local feature and the codebook codes. [96]

3.4 Image categorisation

In the final step of the BoF approach, the input images are categorised. In the unsupervised visual object categorisation, the codebook feature histograms (i.e. images) can be categorised by using any clustering method. One of the most popular methods is the k-Means clustering which is very simple, and thus, it can be used as a baseline method as in [92]. The goal of the thesis is to improve UVOC using BoF and the image categorisation step is as important as the other steps in the categorisation process. Thus, in this section other related categorisation methods are described and their performance in a typical UVOC task is evaluated.

3.4.1 k-Means clustering

The k-Means clustering algorithm has been used in many applications. One reason can be its simplicity, which is the reason why it is also used in the thesis as the baseline method for image categorisation. k-Means has been used earlier for UVOC by Tuytelaars et al. [92]. In their experiments, the k-Means clustering performed very well compared to the other methods.

The k-Means clustering algorithm consists of two phases: a cluster assignment phase and a cluster updating phase. Initial cluster locations are usually chosen randomly. Then, each data point is assigned to its closest cluster (the cluster assignment phase). Next, cluster centroids are updated by computing the mean of data points belonging to a specific cluster (the cluster updating phase). This is repeated as long as the cluster centroids are changing or the maximum number of iterations is reached. More formal presentation of the algorithm is given in Algorithm 3.3. In the beginning of the k-Means, the cluster centroids $\boldsymbol{CB}$ are set by randomly choosing k data points from $\boldsymbol{D}$, i.e., $\boldsymbol{CB} = \boldsymbol{D}_{\boldsymbol{idx}_{1,\ldots,N_{cb}}}$ where $\boldsymbol{idx} = randperm(length(\boldsymbol{D}))$. Then, each data point $\boldsymbol{d}_i \in \boldsymbol{D}$ is assigned to its closest cluster $\boldsymbol{clusters}_i$. Next, cluster centroids $\boldsymbol{CB}$ are updated by computing the mean of the members in each cluster.

3.4.2 Self-Organizing Map

One possible method of categorising images is to use Self-Organizing Map (SOM) [53]. In SOM, nodes on the SOM are organised so that similar nodes are closer to each other and dissimilar are further apart. The SOM algorithm is very simple and it is shown in Algorithm 3.4. At first, it can be initialised randomly as in k-Means or by using some heuristics to obtain better initialisation, e.g., by computing the principal components and using them to give initial weights for the SOM nodes. After the initialisation, for each input sample $\boldsymbol{d}_1, \ldots, \boldsymbol{d}_{length(D)} \in \boldsymbol{D}$, the closest node, bm, (the Best Matching Unit (BMU)) from the codebook $\boldsymbol{CB}$ is searched and the weight of the BMU is changed so that it is moved towards the given data point. BMU bm is defined as follows:

$$\|\boldsymbol{d} - \boldsymbol{CB}_{bm}\| = \min_i \{\|\boldsymbol{d} - \boldsymbol{CB}_i\|\} \ . \tag{3.10}$$

To maintain the topology, also BMU's neighbours (in the topology) are updated in such a way that the weights of the nodes that are closer in a topology to the BMU are changed more than the weights of the nodes that are further away. Neighbouring nodes for the

Algorithm 3.3 k-Means clustering algorithm

```
Require: $\boldsymbol{D}$ // Dataset e.g. BoF codebook histograms
Require: $N_{cb}$ // Number of clusters
Require: $maxIter$ // Maximum number of iterations
  $\boldsymbol{CB}_0 \leftarrow zeros(N_{cb}, getDimensions(\boldsymbol{D}))$
  $\boldsymbol{idx} \leftarrow randperm(length(\boldsymbol{D}))$
  $\boldsymbol{CB}_1 \leftarrow \boldsymbol{D}_{\boldsymbol{idx}_{1,\dots,N_{cb}}}$
  $t = 1$
  // Repeat clustering while clusters are changing
  while $\boldsymbol{CB}_{t-1} \neq \boldsymbol{CB}_t$ and $t \leqslant maxIter$ do
    $\boldsymbol{CB}_t \leftarrow \boldsymbol{CB}_{t-1}$
    // Cluster assignment phase
    for $i = 1$ to $length(\boldsymbol{D})$ do
      $\boldsymbol{clusters}_i \leftarrow \underset{c \in 1,\dots,N_{cb}}{\arg\min}\ Dist(\boldsymbol{d}_i, \boldsymbol{CB}_{t,c})$
    end for
    // Cluster updating phase
    for $c = 1$ to $N_{cb}$ do
      $\boldsymbol{CB}_{t+1,c} \leftarrow \underset{i}{mean}(\boldsymbol{D}_i | \boldsymbol{clusters}_i = c)$
    end for
    $t = t + 1$
  end while
  return $\boldsymbol{CB}_t$ // Codebook i.e. cluster centroids
```

BMU bm are search by finding nodes that are connected to the best match in the topology. The neighbourhood function ω for the BMU $\boldsymbol{CB}_{bm}$ is defined by the spatial location of the BMU $\boldsymbol{x}_{bm}$ and spatial location of the neighbouring node $\boldsymbol{x}_c$ as follows:

$$\omega(\boldsymbol{x}_{bm}, \boldsymbol{x}_c, \alpha_t) = \alpha_t \exp\left(\frac{-\|\boldsymbol{x}_{bm} - \boldsymbol{x}_c\|^2}{2\tau^2(t)}\right) \quad , \tag{3.11}$$

where α_t is learning factor at time step t, $\boldsymbol{x}_{bm}$ and $\boldsymbol{x}_c$ are coordinates of the BMU bm and the neighbouring node c and τ is for adjusting the learning rate. The function ω is greater than zero for all the nodes within the neighbourhood and zero for all other nodes. Data points can be given at once (batch mode) or separately. The output should be quite similar, but the batch mode is faster. [53]

In Fig. 3.5, Caltech-101 images have been categorised using a 20 × 15 SOM. The SOM is trained with 30 × 101 images, i.e., codebook histograms of 3030 images are given to the SOM for training. For each node, the best matching image, (i.e., minimal distance from the codebook feature histogram to the SOM node that has not been used yet) is selected and shown in the figure. We can see in the figure how the SOM is able to find groups of similar images. For example, in the bottom left corner, there are cartoon characters (Garfield and Snoopy) and other drawn images. In the left top corner, there are pictures of animals without backgrounds. In the centre of the result, there are many images of faces and also accordions, and in between the accordions and faces, there is an image with both.

Algorithm 3.4 Self-organizing Map [53]

```
Require: $\boldsymbol{D}$ // Input data, e.g. BoF codebook histograms
Require: $N_{cb}$ // Number of nodes in the map, i.e. number of clusters
Require: $l$ // Learning factor $(0 > l < 1)$.
Require: $\epsilon$ // Threshold to stop the learning process
  $t \leftarrow 1$
  $\alpha_t \leftarrow 1$
  $\boldsymbol{CB} \leftarrow rand(N_{cb}, getDimensions(\boldsymbol{D}))$
  $\delta \leftarrow inf$
  while $\delta \geqslant \epsilon$ do
    $\boldsymbol{CB}' \leftarrow \boldsymbol{CB}$
    for all $\boldsymbol{d}_1, \ldots, \boldsymbol{d}_{length(D)} \in \boldsymbol{D}$ do
      $bm \leftarrow \underset{c \in 1,\ldots,N_{cb}}{\arg\min}\ Dist(\boldsymbol{d}_i, \boldsymbol{CB}_c)$
      // Update weight of the best matching unit and its neighbours
      for $c = 1$ to $N_{cb}$ do
        $\boldsymbol{CB}_c \leftarrow \boldsymbol{CB}_c + \omega(bm, c, t, \alpha_t)\,(\boldsymbol{d}_i - \boldsymbol{CB}_c)$
      end for
    end for
    $\alpha_{t+1} \leftarrow \alpha_t * l$
    $\delta \leftarrow \sum_{c=1}^{N_{cb}} \sqrt{(\boldsymbol{CB}_c - \boldsymbol{CB}'_c)^2}$
    $t \leftarrow t + 1$
  end while
  return $\boldsymbol{CB}$
```

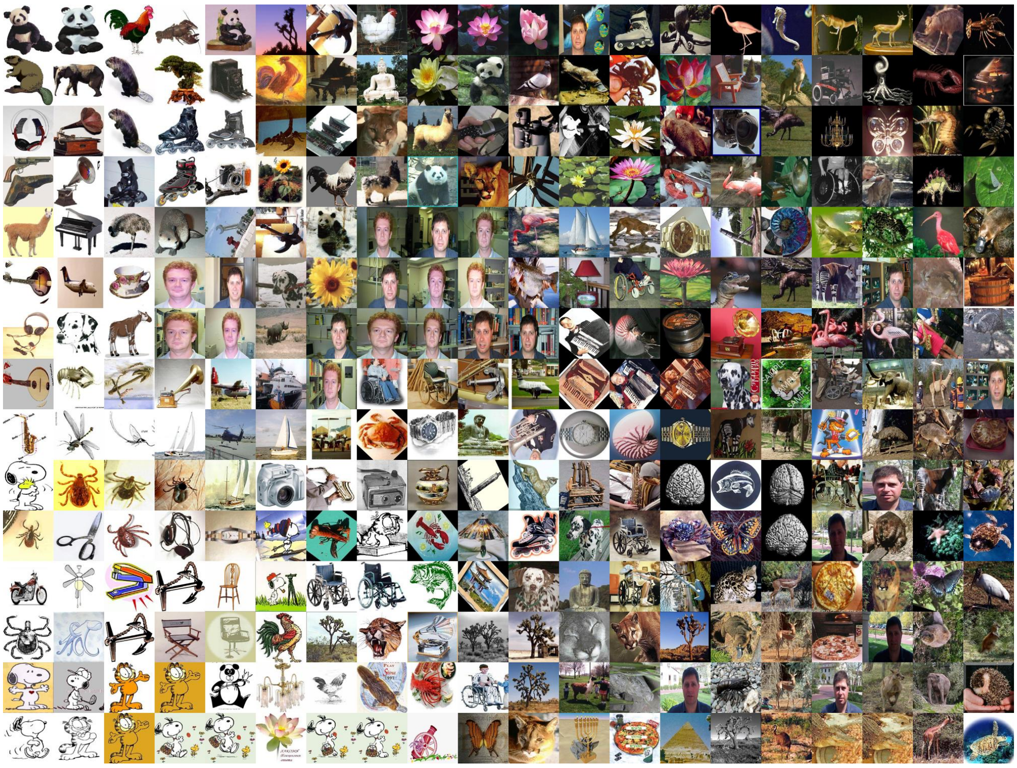


Figure 3.5: Result of the image categorisation using SOM on Caltech-101 images. Only one image is shown for each node.

3.4.3 Neural Gas

Neural Gas [62] is similar to the SOM, but in Neural Gas nodes are not organised in a topology. Instead of forcing the nodes in a predefined topology, the algorithm learns a structured manifold in the feature space which is defined based on the distances of the nodes in the input space. The learning algorithm in Neural Gas is presented in Algorithm 3.5. As in the SOM algorithm, the codebook $\boldsymbol{CB}$ can be initialised with random weights. Then, it is trained by choosing randomly one sample $\boldsymbol{d}_r$ at time t. The training step is repeated t_{max} times. Each time, N_k closest nodes are searched for the randomly chosen training sample by computing distances from the training sample $\boldsymbol{d}_r$ to all nodes in the codebook $\boldsymbol{CB}$. These best matches are founded by sorting the distances $\boldsymbol{scores}$ into ascending order and using indexes $\boldsymbol{idx}_{1,\ldots,N_k}$ of the sorted list denote the N_k best matching units. Then, weights of the best matches are updated based on the distance from the training sample $\boldsymbol{d}_r$ to the best matching unit CB_{idx_i}, neighbourhood factor λ_t, and the learning factor α_t. In each learning iteration, or epoch, weights of the N_k best matching nodes are updated. The algorithm takes samples $\boldsymbol{D}$, initial and final learning factors α_0 and α_{final}, initial and final neighbourhood factors λ_0 and λ_{final}. The output of the Neural Gas clustering is a codebook, i.e. weights of the nodes $\boldsymbol{CB}$.

In Fig. 3.6, Iris dataset [35] is clustered using the three previously presented clustering

Algorithm 3.5 Neural Gas learning algorithm [62, 55]

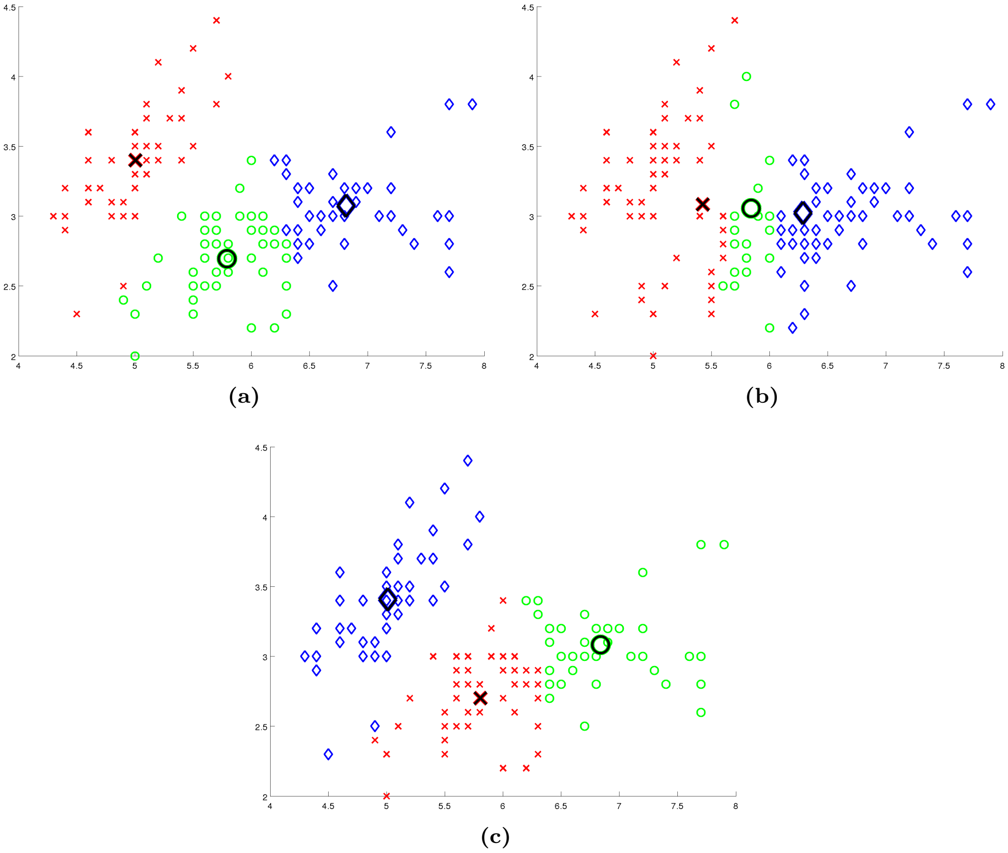
```
Require: $\boldsymbol{D}$, $\alpha_0$, $\alpha_{final}$, $\lambda_0$, $\lambda_{final}$, $t_{max}$, $N_k$ $N_{cb}$
  $\boldsymbol{CB} \leftarrow rand(N_{cb}, getDimensions(\boldsymbol{D}))$
  for $t = 0$ to $t_{max}$ do
    $r \leftarrow rand(1)$ // Select random training sample
    $\lambda_t \leftarrow \lambda_0(\lambda_{final}/\lambda_0)^{t/t_{max}}$ // Update size of the neighbourhood
    $\alpha_t \leftarrow \alpha_0(\alpha_{final}/\alpha_0)^{t/t_{max}}$ // Update learning rate
    // Distance from training sample to every codebook node
    $\boldsymbol{scores} \leftarrow Dist(\boldsymbol{d}_r, \boldsymbol{CB})$
    // Sort distances into ascending order and get order ($\boldsymbol{idx}$)
    $[\boldsymbol{scores}, \boldsymbol{idx}] \leftarrow sort(\boldsymbol{scores})$
    for $i = 0$ to $N_k$ do
      $\boldsymbol{CB}_{idx_i} \leftarrow \boldsymbol{CB}_{idx_i} + \alpha_t e^{-N_k/\lambda_t}(\boldsymbol{d}_r - \boldsymbol{CB}_{idx_i})$
    end for
  end for
  return $\boldsymbol{CB}$
```

methods. Neural Gas and k-Means produce very similar results, as we can see that the cluster centroids are almost in identical places, even though the ordering is different. SOM assigned nodes are close to the centre of the whole data set. In addition, cluster assignments differ from Neural Gas and k-Means. The cluster in the middle is rather small in comparison with the other clusters. Obviously, the parameters of the SOM are not optimal, the size of the SOM should be larger or the shape of the map should be planar, toroidal or cylinder so the SOM nodes would cover the data more evenly. Here, SOM was trained using SOM Toolbox [1] with default parameters.

3.4.4 Hierarchical clustering

One of the problems with the previously introduced clustering methods is that the number of clusters or nodes must be defined beforehand. When one is using a benchmark image set, it is not difficult to fix this value, but then the method is not fully unsupervised and in real life, it is not always possible to fix or know the number of categories. This supports the use of hierarchical clustering instead of “flat” clustering. However, it does not solve the problem of selection the optimal number of categories.

In the hierarchical clustering, one can start clustering from the bottom by connecting clusters together, or from the top by dividing each cluster into parts. In the bottom-up approach, all the data points are assigned to their own cluster in the beginning. Then, two clusters are merged together on the second level to form a new cluster. These two clusters that are going to be merged are selected based on distances and there is many options to compute the distance such as, single link, complete link, average link distance, median link, centroid and Ward’s method [43, 72]. In the top-down hierarchical clustering, all the images are in a single cluster in the beginning. Then, the cluster is divided into partitions for example using k-Means [43].



(a) (b) (c)

Figure 3.6: Examples of clustering results using (a) k-Means, (b) SOM [53] and (c) Neural Gas [62] using Iris data set [35] with two first dimensions. Data points from different categories are marked with different colours and symbols and nodes/cluster centroids are marked with larger symbols.

3.5 Experiments

In this section, the performance of the BoF approach is evaluated in the supervised and unsupervised VOC using the original Caltech-101 dataset [30], Randomised Caltech-101 dataset [52], and the extended version of Caltech-101, called the Caltech-256 dataset [39].

3.5.1 Experiment 1: Randomising Caltech 101 images

In the thesis, a new image set based on Caltech-101 [30] was introduced. This new image set is called Randomised Caltech 101 and the effects of the randomisation process are evaluated in the following experiment. In this experiment, the effect of the background, the object orientation and the spatial location is studied in several experiments using supervised VOC with 1-NN classifier. According to the standard VOC evaluation procedure, categorisation performance is utilised as the quantitative evaluation measure. The performance is computed as the average classification accuracy over classes (2.1) as it was presented by Lazebnik et al. [57]. The performance values are computed as a function of the number of categories. The asymptotic VOC behaviour is important since the methods should ultimately cope with thousands or even hundreds of thousands of categories.

The experimental procedure is randomised itself: for each number of categories, 10 independent iterations were performed by first selecting random categories and 30 random training images for each category. 20 images, or what was omitted from the training process, were used in testing. The experiment was repeated with 5, 10, 20, 50, and 101 object categories.

There were six data configurations for which the BoF method was tested and the codebook size optimised: i) the original Caltech-101 data (*Caltech 101 (full)*), ii) only the Caltech-101 foreground objects (*Caltech 101 (Fg)*), iii) only the Caltech-101 backgrounds (*Caltech 101 (Bg)*), iv) the full randomised images according to Section 2.1.2 (*r-Caltech 101 (Full)*), v) Foregrounds from Randomised Caltech-101 (*r-Caltech 101 (Fg)*) and vi) Backgrounds from Randomised Caltech-101 (*r-Caltech 101*). Classification results using Bag-of-Features with the Hessian-Affine detector and the SIFT descriptor and a 200 × 1 codes SOM codebook with the 1-NN classifier are shown in Fig. 3.7.

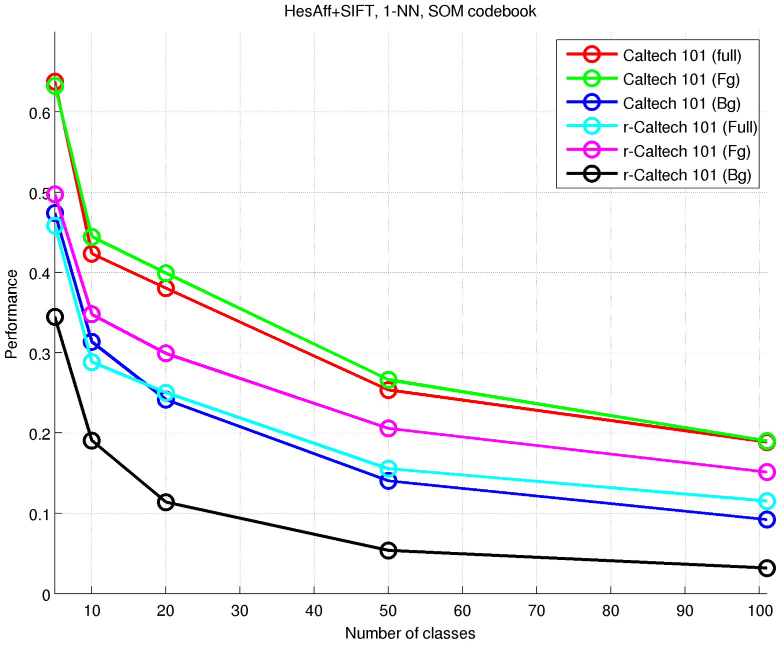


Figure 3.7: Performance of the Bag-of-Features approach using Hessian-Affine detector, SIFT descriptor and 1-NN with various modified Caltech-101 image sets.

The best performance in Fig. 3.7 was achieved using the foregrounds only from the original Caltech-101 (the green curve). The background clutter had virtually no effect on the performance with Caltech-101 since the performance with the full images was almost the same (the red curve). Most of the local features are extracted from the object (see Fig. 3.2e) and thus ground truth foreground segmentation has no significant effect. The rotation and scaling of the foregrounds affected the detected features which is evident from the results for the Randomised Caltech-101 foregrounds (the magenta curve) which is the third best, but clearly outperformed by the original foregrounds and full images. In Randomised Caltech-101, the background clutter had the expected result as it significantly reduced the performance compared with the Randomised Caltech-101 foregrounds. Interestingly, the Caltech-101 background only (the blue curve) achieved almost the same performance as the full Randomised Caltech-101 images (the cyan curve). The worst performance was achieved with the Randomised Caltech-101 backgrounds only (the black curve). It is noteworthy that the worst result does not correspond to random chance which can be explained by the fact that since the features in the foreground were just omitted, the total number of detected features correlates with the object sizes, and therefore, provides a cue of the class.

As a summary, the Randomised Caltech-101 data set provides a more challenging test benchmark for the VOC methods, since the background clutter and invariance have a drastic effect on the performance. The Randomised Caltech-101 does not provide natural data, but it should be used with Caltech-101 to represent how well a method can tolerate geometric transformations and background clutter.

3.5.2 Experiment 2: Local feature detector experiment

This experiment focuses on the comparison of different region detectors and different implementations. Detected local features were described with SIFT region descriptor and codebooks were built by using SOM. The size of the SOM is chosen for each detector separately, in the way that the performance is the highest on average. Classification is made using 1-NN classification rule. In this experiment, Randomised Caltech-101 dataset was used and the evaluation protocol was the same as in the previous experiment: 10 independent trials with 30 randomly chosen training images and 20 randomly chosen test images. This was repeated for 5, 10, 20, 50 and 101 classes as in the previous experiment. The performance of VOC with different local feature detectors is shown in Fig. 3.8.

The Hessian-Affine detector (the Feature Space implementation) seems to provide the best results followed by dense sampling, and Harris-Laplace (the Feature Space implementation). Vireo implementations do not perform as well as the Feature Space which might be due to different default parameters. In this experiment, the default parameters were used. This experiment verifies that Mikolajczyk's Hessian-Affine detector outperforms the other detectors in the 1-NN based VOC task.

3.5.3 Experiment 3: Finding an optimal threshold for accepting a local feature match

To define an optimal threshold for accepting matches, 30 images were chosen from 101 Randomised Caltech-101 object categories. Then distances between every local feature

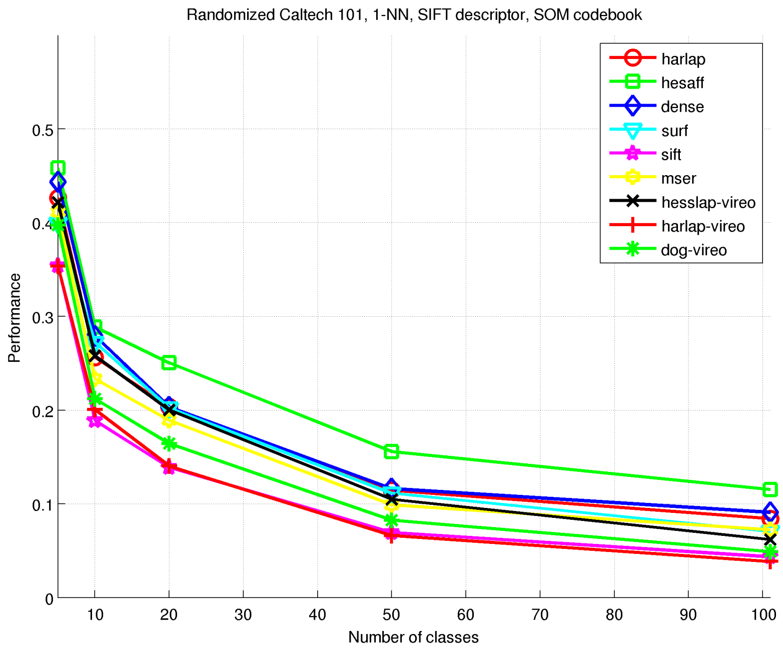


Figure 3.8: Performance curves of VOC method using different region detectors, and Randomised Caltech 101 image set, SIFT descriptor, SOM codebooks and 1-NN classification rule.

pair within an object category was computed. Finally, distributions of foreground-foreground (FGFG) and foreground-background (FGBG) matches were computed and are shown in Fig. 3.9.

Fig. 3.9 shows that the matching FGFG features have generally shorter Euclidean distance (smaller error) than FGBG matches. The result makes it possible to define a threshold that can be used to filter out matches that are more likely a FGBG (false) than a FGFG (correct) match. When the difference between the correct and false matches is maximised, more foreground features should be accepted, and thus, the codebook histograms should describe the foreground more accurately and categorisation performance should be increased. According to the difference (FGFG-FGBG), the matching threshold is set to $maxDist = 1.5 \times 10^5$ with the Hessian-Affine detector and the SIFT descriptor.

3.5.4 Experiment 4: Local feature filtering using sets of common features

In this experiment, the Randomised Caltech-101 was used because the image backgrounds vary more than in the original Caltech-101 image set. Thus, it should be more suitable for the experiment, where a common set of local features is searched with Algorithm 3.1 and used for generating the codebook and codebook histograms. Maximal distance for accepting local feature matches was set to $maxDist = 1.5 \times 10^5$ based on the previous experiment. However, $minHits$ is still unknown, it is optimised experimentally by using

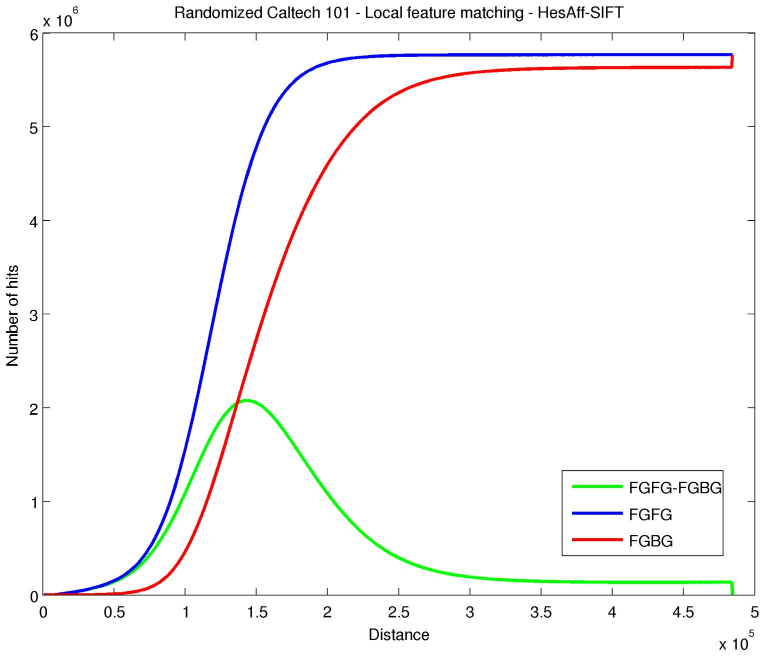


Figure 3.9: Cumulative sums of matching local features. The blue curve corresponds to the number of matching FGFG features below a threshold, and the red curve corresponds to the FGBG matches. The green curve is the difference between FGFG and FGBG matches.

$minHits = 0, 1, 5, 10, 20, 50, 100$. When $minHits = 0$, it means that no local features are filtered, and thus, it can be used as the baseline. In this experiment, the 1-NN classifier is used to avoid all the necessary parameter tuning. The results of the experiment are shown in Fig. 3.10.

In Fig. 3.10, we can see that the classification performance is improved only a very small amount by filtering out local features that are not frequent. Example images of filtered local features in Fig. 3.3 show that some of the local features are filtered out from the background, but most of the local features are preserved even though the minimum number of hits is large. Because the filtering affects the extracted local features only by a small amount the effect on the VOC performance is also insignificant.

3.5.5 Experiment 5: Codebook generation experiment

In this experiment, Caltech-101 dataset was used. The testing procedure is the same as in the previous experiments: 30 images are randomly chosen from each class for training and 20 images are randomly chosen for testing (if there was 20 images left after the training, otherwise the rest of the images were used). This was repeated 10 times for 5, 10, 20, 50 and 101 classes. In this experiment, SIFT (DoG) detector was used with

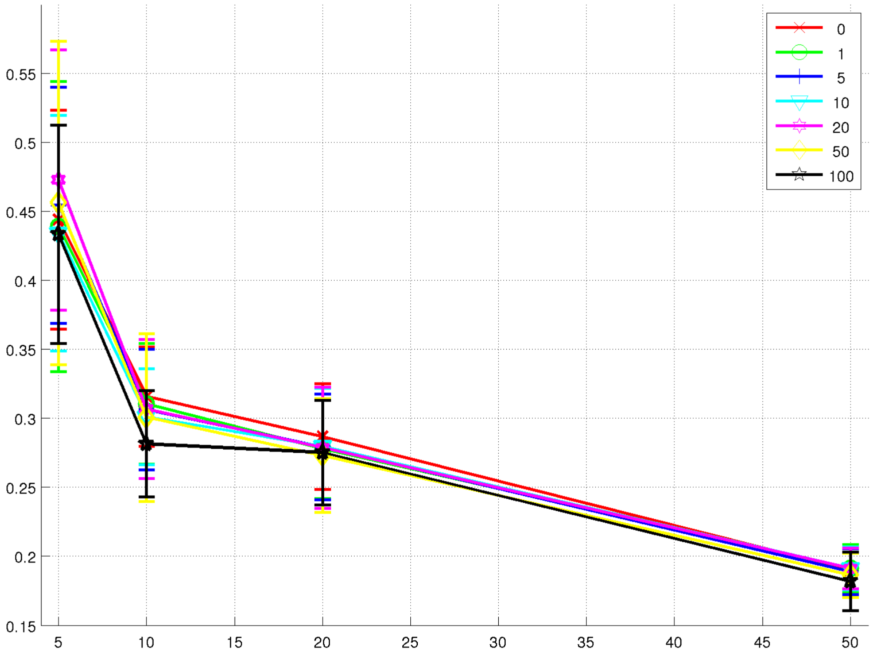


Figure 3.10: Results of the local feature filtering experiment using a list of common features and 1-NN classifier.

SIFT descriptor. Visual codebooks were generated using k-Means and SOM and the images were classified using 1-NN classification rule. The comparison between k-Means and SOM generated codebooks of various size are shown in Fig. 3.11.

Fig. 3.11 shows that the supervised VOC approach using SOM can predict object categories more accurately in most of the cases. This is especially true when the size of the visual codebook is small, the BoF approach using SOM performs better. However, when the size of the visual codebook is increased significantly, the performance drops when SOM is used, but increases slightly when k-Means is used. The cause of this effect could be that the amount of training data (the number of local features) for SOM is not enough if the size of the SOM is large.

3.5.6 Experiment 6: Unsupervised visual object categorisation using Caltech-256

In this experiment, the presented UVOC approach based on the BoF approach was compared to that of [92], which represents the current state-of-the-art in UVOC. The results are reported for the same 20 categories of Caltech-256. According to the results of the experiment 2, the Hessian-Affine local feature detector was used with SIFT descriptor to extract local features. Codebooks were generated using 1-dimensional SOM, i.e. codebooks were 50 × 1, 100 × 1, 200 × 1, 500 × 1, 1000 × 1, 2000 × 1, 5000 × 1, and 10000 × 1. For unsupervised categorisation, the previously used 1-NN classifier is replaced with a

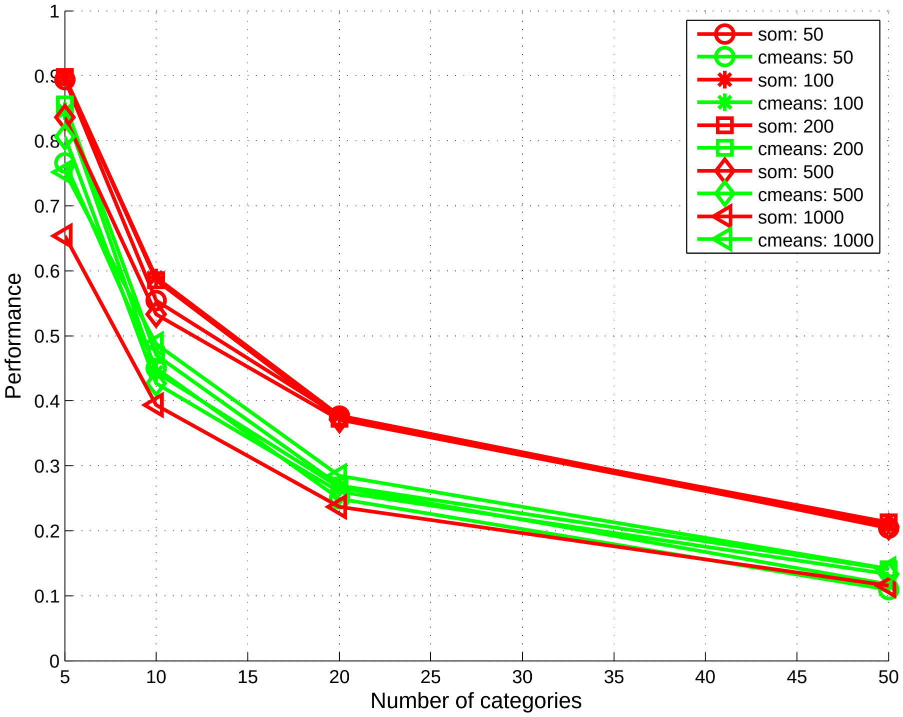
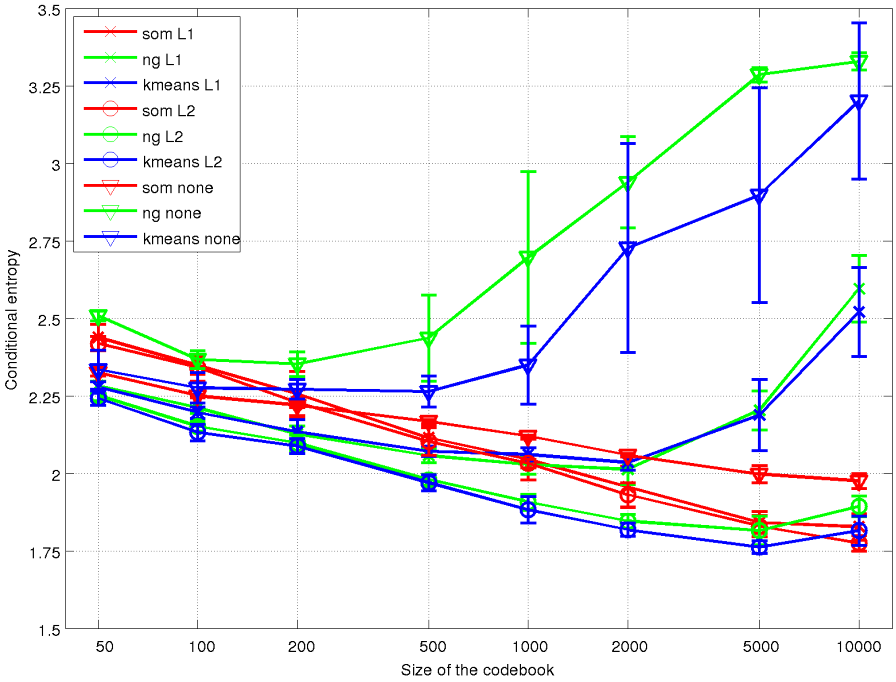


Figure 3.11: Codebook generation method comparison using Caltech 101 image set.

20 × 1 unit category book generated by the SOM algorithm. For comparison, the same experiments are also conducted with the Neural Gas [62] with 20 nodes and k-Means algorithms with 20 clusters. The performance is reported by computing the conditional entropy defined in Eq. (2.6). In this experiment, the effect of the normalisation is also investigated using L1-normalisation Eq. (3.8) and L2-normalisation Eq. (3.9). These results were compared with results achieved without using any normalisation.

In Fig. 3.12, conditional entropy graphs are shown for the different methods and sizes of the codebook. In the Tuytelaars et al. [92] protocol, the size of the category book was fixed to the number of categories. The different colours denote the different methods and the markers denote the different normalisation methods.

Two important findings can be made based on Fig. 3.12. First, the large codebooks provide better results. Second, the k-Means and Neural Gas algorithms are very sensitive to the data normalisation, whereas SOM is not. Moreover, the SOM performance steadily increases, and with the largest codebook, it outperforms the second best approach, k-Means with the L2-norm normalisation. In the original work by [92], the best performance, the conditional entropy value of 1.78 (+/- 0.03), was achieved with dense sampling, binarised features, and a k-Means category book. A comparable performance of 1.78 (+/-0.02) was achieved with the proposed method.



(a)

Figure 3.12: Performances in the UVOC experiment with the 20 Caltech-256 classes [92]. The red line stands for SOM categorisation, the green for Neural Gas and the blue for k-Means. The cross denotes L1-normalisation, the circle L2-norm normalisation and the triangle denotes that codebook histograms are not normalised. Performance measured using the condiotional entropy [92].

As a summary, we can conclude that this experiment verified our previous results indicating that the SOM algorithm is a competitive alternative to clustering methods, such as the k-Means algorithm. It also has some advantageous properties, such as its stable performance with various normalisation methods.

3.5.7 Experiment 7: Unsupervised object discovery from Randomised Caltech-101

This experiment is the most challenging UVOC experiment with images from the Randomised Caltech-101 data set. For each iteration, 30 images are randomly selected from each category, and following the previous experiment, the size of the category book is fixed to the true number of categories. The other parameters are selected based on the previous experiments: the Hessian-Affine detector, SIFT descriptors, and the L2-norm normalisation of the histogram features.

Fig. 3.13 presents the results of this experiment. Performance of the proposed method is reported using both performance measures, the conditional entropy Eq. (2.6) by Tuytelaars et al. [92] and Sivic performance [83] Eq. (2.2). Note that for the conditional entropy, the smaller values are better, and for the classification accuracy, the greater values are better. By comparing the results in Figs. 3.13a and 3.13b, it is obvious that the both performance measures provide the same information: the performance steadily degrades as the number of categories increases, and on average, the codebook size 1000 provides the best performance (insignificant difference to others). Fig. 3.13b shows that the performance degrades as the number of categories increases, as is expected, but compared to pure chance, the performance improves (approx. 2 times better for 20 categories and 5 times better for 100 categories).

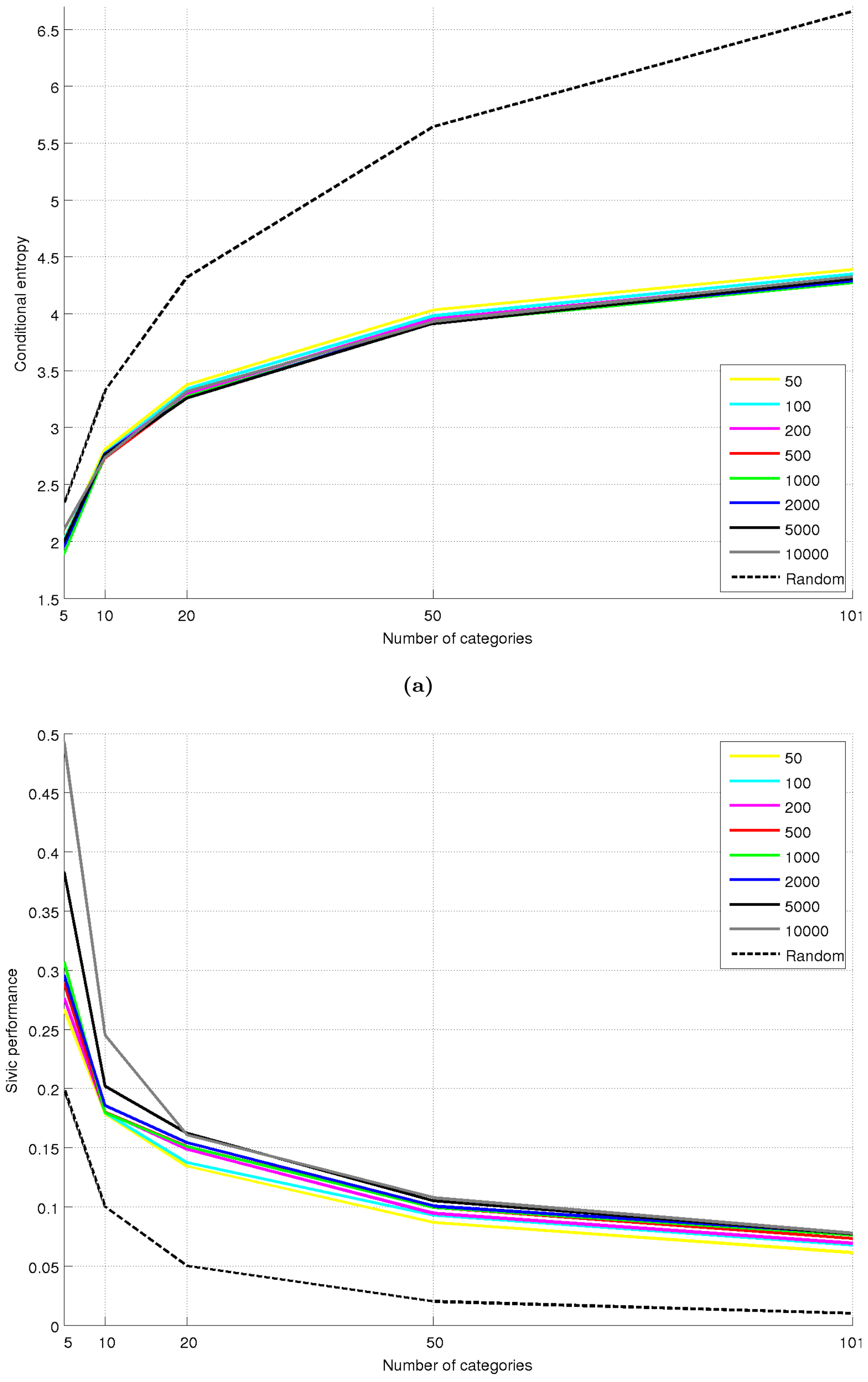
3.6 Summary

In this chapter, an UVOC approach using the BoF approach was presented, alternative methods for different steps were discussed, and in the experiments, seven experiments were carried out.

In the first experiment, the importance of the background, the spatial location of the object and the orientation of the orientation in the supervised VOC was tested. The result of the experiment was that the background is very important in the Caltech-101 dataset [30] which verifies the results by Ponce et al. [77]. Moreover, the orientation and spatial location of the objects decreases the performance of the VOC. In the second experiment, different local feature detectors were compared with each other in the supervised VOC task. The result of the experiment, was that the Hessian-Affine local feature detector performs the best.

In the third experiment, a threshold for accepting local feature matches was searched and founded by making pairwise comparisons from foreground local features to foregrounds (FGFG) and to backgrounds (FGBG). The threshold was chosen by selecting the value that has high $FGFG - FGBG$, i.e. it should accept most of the local features detected from the foreground while filtering out local features extracted from the background. In the fourth experiment, a new method for filtering out local features that are extracted from the background was tested. Even though the method seems to be able to filter some local features that are coming from background while preserving most of the local features extracted from the foreground, the new method did not have significant impact in the supervised VOC performance. In the fifth experiment, k-Means and SOM clustering methods were compared in the codebook generation task. The evaluation was performed by evaluating the performance of the supervised VOC when using these two clustering methods. The result of the experiment was that the SOM algorithm can generate visual codebooks that are at least as good as codebooks generated by k-Means clustering and when the size of the codebook is small, then SOM can generate better codebooks for the supervised VOC with the dataset and the local feature extraction method that were used.

In the sixth experiment, the 1-NN classifier was replaced with unsupervised clustering. Three alternative clustering methods: k-Means, SOM and Neural Gas, were tested. The result was that the clustering methods perform quite evenly at their best. However, SOM seemed to be much more stable when compared with different normalisation methods. In the seventh experiment, the performance of the UVOC using the standard BoF approach to describe images was tested with various sizes of codebooks generated SOM. A conclusion of the experiment was that the performance decreases rapidly as in the supervised VOC in experiments 1 and 2. The size of the codebook affects to the performance, especially when the number of categories is small (5 or 10). However, with larger number of categories the difference between the performance with larger and smaller codebooks decreases. In the last experiment, the performance was measured with conditional entropy Eq. (2.6) by Tuytelaars et al. [92] and Sivic's performance measure Eq. (2.2) [83]. Results with both performance evaluation methods are nearly identical. Small differences are due to the fact that in Sivic's performance evaluation method, each category contributes equally to the final performance evaluation, whereas in conditional entropy, each sample contributes equally to the final performance evaluation.



(a)

(b)

Figure 3.13: Results for the Randomised Caltech-101 UVOC experiment: (a) conditional entropy; (b) Sivic performance [83].

Chapter IV

Utilising spatial information with Bag-of-Features

In this chapter, spatial information is used to improve the categorisation performance of the BoF UVOC approach described in Chapter 3. The standard BoF approach disregards all the spatial information. Therefore, the local features can be in any spatial configuration and the BoF histogram remains the same, even though the appearance of the image changes. One can think about the spatial configuration as an analogy to the order of the words in a paragraph. The order of the words is very important to the interpretation of the text. Similarly the spatial configuration of the local features is important for VOC. In this chapter, different approaches to utilise the spatial information in the BoF approach are presented.

4.1 Bag-of-Bag-of-Features

The first approach to add spatial information to the UVOC is to use a "Bag-of-Bag-of-Features" (BoBoF) approach which is based on the BoF approach [84, 14]. In the BoBoF approach, an input image is divided into parts using a grid or segmentation. Then each segment (or grid cell) is described using the BoF approach. A secondary codebook, the BoBoF codebook, is built by clustering the BoF histograms in similar manner with BoF codebook generation in the BoF approach where the BoF codebook is built using local features. Next, each input image is described by BoBoF histograms which are generated by matching the BoF histograms to the BoBoF codebook and computing the histogram. The idea in the BoBoF approach is that each BoBoF code is constructed from local features that are nearby each other in the original image. When local features that are nearby each other are described with a single code, it adds a spatial constrain in the image description although the BoF approach disregards all the spatial information. The algorithm to describe the images with BoBoF approach is presented in Algorithm 4.1. The algorithm takes a list of training images $\boldsymbol{I}_{1,2,\ldots,N_{train}}$, BoF codebook $\boldsymbol{CB}$, a list of descriptors $\boldsymbol{D}_{1,2,\ldots,N_{train}}$ as inputs, and spatial locations of the extracted local features $\boldsymbol{L}_{1,2,\ldots,N_{train}}$.

Algorithm 4.1 Image categorisation using the BoBoF approach.

```
Require: $\boldsymbol{I}$, $\boldsymbol{CB}$, $\boldsymbol{D}$, $\boldsymbol{L}$, $N_{bobsize}$
  $N_{train} \leftarrow length(\boldsymbol{I})$
  for $i = 1, \ldots, N_{train}$ do
    $\boldsymbol{imgS}_i \leftarrow imgSegmentation(\boldsymbol{I}_i)$
  end for
  for $i = 1, \ldots, N_{train}$ do
    for $j = 1, \ldots, length(\boldsymbol{imgS}_i)$ do
      $\boldsymbol{idx} \leftarrow \boldsymbol{L}_i \subset \boldsymbol{imgS}_{i,j}$ // Local features from image $i$ belonging to segment $j$
      $\boldsymbol{f}_{i,j} \leftarrow generateCodebookHist(\boldsymbol{D}_{i,\boldsymbol{idx}}, \boldsymbol{CB})$ // See Algorithm 3.2.
    end for
  end for
  // Build a codebook as in the BoF approach using codebook histograms.
  $\boldsymbol{CB}_{bobof} \leftarrow generateCodebook(\boldsymbol{f}, N_{bobsize})$
  for $i = 1, \ldots, N_{train}$ do
    $\boldsymbol{f}_i^{bobof} \leftarrow generateCodebookHist(\boldsymbol{CB}_{bobof}, \boldsymbol{f}_i)$
  end for
  return $\boldsymbol{f}^{bobof}$
```

4.1.1 Grid approach

One straightforward approach to add spatial information to the BoF approach is to divide an input image into parts using a grid and run the BoBoF algorithm introduced in Algorithm 4.1. This approach is similar to the approach of Lazebnik et al. [57] where they used a spatial pyramid approach where an image is divided into 1×1, 2×2 and 4×4 grids and computed local feature histograms for each cell using the standard BoF approach. Then they concatenated all the feature histograms together and used Support Vector Machine (SVM) to learn the object categories. In their case, the final description of the image becomes $(1^2 + 2^2 + 4^2) \times numberOfCodes = 21 \times numberOfCodes$, whereas in the case of BoBoF the length of the image description is defined by the size of the BoBoF codebook, thus making BoBoF description of the image significantly more compact.

4.1.2 Segmentation approach

Instead of dividing an image into parts using a grid, a segmentation method can be used. If the segmentation method can successfully separate different objects in the scene, object categorisation performance could be increased. However, the segmentation and categorisation problem is a chicken-egg problem, i.e., in order to get good segmentation one should recognise objects in the scene and on the other hand, to recognise objects in the scene, one should segment the objects. Thus, the segmentation problem is difficult.

One of the most popular segmentation methods is the Normalised Cuts segmentation [82]. Originally Normalised Cuts was used to find clusters from data sets, but Shi and Malik developed a method that used Normalised Cuts to find segments from images. In Normalised Cuts, data points are clustered using a pairwise similarity matrix of the data points which are used as weights for the edges that connect the data points. The Normalised Cuts algorithm tries to find optimal cuts by searching for edges that can be cut while maintaining high similarity between the nodes that are connected and cutting the edges that connect nodes with smaller similarities. The problem of finding the optimal cut is NP-hard, thus the solution must be approximated which is done by using spectral clustering. The Normalised Cuts algorithm has been successfully applied in [41, 80], and thus, it is chosen also for this study. One of the advantages of Normalised Cuts is that the user can choose how many segments the method should find.

4.2 Spatial matching of local features

An alternative approach to using spatial information with the BoF approach is to spatially match (i.e. verify) local features. Instead of matching local features using only distances between the descriptors of the local features, it is possible to use spatial information of the local features to make sure that the local feature matches are correct. This method has been used successfully in specific object detection [11, 76] and homography estimation [86]. However for object category detection the spatial matching of local features is more complicated since the visual appearance can vary more, i.e. the appearance of the local patches can vary and the spatial locations of the object parts can be different. Lankinen and Kamarainen [56] proposed a new method for automatic landmark assignment which finds parts of the visual objects that are visually similar and in the same spatial configuration. Their method is capable of finding stable landmarks from a set of images belonging to the same category, even though the visual appearance of the objects varies to some extent.

4.2.1 Spatial matching approach

The spatial matching algorithm used in the thesis is based on the unsupervised landmark alignment algorithm introduced by Lankinen and Kamarainen [56]. However, instead of finding landmarks for a set of images from a specific category, the task here is to compute the distance between a pair of images based on the descriptors of spatially matching local features. Thus, their algorithm is used to find landmarks for a pair of images and compute the fitness of the landmarks. This information is used to define a distance between a pair of images.

However, the spatial matching step is computationally expensive, and thus, it is not possible to match all images against each other, especially when there are hundreds or thousands of images. One must choose candidate images carefully. Fortunately, we have the UVOC method based on BoF which can be used to find a list of candidate images for every given image. Using the BoF histograms of the images, it is trivial to find a sorted list of the most similar candidate images. A list of the N_{cand} best matching images can be given to the unsupervised landmark alignment algorithm by Lankinen and Kamarainen [56] to spatially match the local features of the given image and candidate images, and to compute the fitness of the matching local features, which is used to define the distance between a pair of images. This approach is similar to the approach by Philbin et al. [76] and Chum et al. [12, 13], but instead of detecting specific object, the objective here is to detect the category of the object.

Selection of candidate images

Since, the spatial matching using RANdom SAmple Consensus (RANSAC) [34] for homography estimation is computationally intensive and it can be performed only to a subset of image pairs, this thesis uses a similar approach to Chum et al. [13], using BoF codebook histograms to obtain a list of similar images. The first step in the candidate image search is to run the BoF approach to describe all the given images with BoF histograms as is illustrated in Fig. 3.1 and described in Chapter 3. Next, a list of similar images is generated for each input image by computing Euclidean distances between the BoF codebook histograms and sorting the distances and images into ascending order. Then, N_{cand} most similar images are given to the spatial matching algorithm where the local features of the given image and N_{cand} most similar images are compared pairwise. Chum et al. [13] did not use a hard threshold to limit the number of candidate images as is done here, instead they used an iterative method to select a cut for each query. In their method, the cut was made after 20 images in a row were predicted to be negative match. This approach was not studied in the thesis, but it is likely that it does not work, because it is possible that the 20 first images are from a wrong category, i.e., false matches.

Spatial matching

The spatial matching algorithm for VOC introduced by Lankinen and Kamarainen [56] finds stable landmarks from a set of images by finding a transformation matrix (i.e. fundamental matrix) between a pair of images using local features and RANSAC [34]. In the transformation matrix estimation using RANSAC, at first, two local features are chosen randomly from the given image. Next, correspondences for the randomly chosen local features are chosen randomly from the N_m best matches in the candidate image. Then the transformation $\boldsymbol{T}$ matrix is computed based on the spatial locations of the correspondences. The transformation matrix $\boldsymbol{T}$ is used to transform the spatial location L of the local feature from the candidate image to the given image. Next, the local features that spatially match after the transformation are then matched using the descriptor part of the local features. If the local feature matches spatially and the distance between descriptors is “small”, then the match is accepted.

The spatial matching algorithm returns the number of matching local features and distances between the matching local features which are used for defining the distance between the pair of images. The distance between the image pair is evaluated by computing a distance between spatially matching local features, choosing the N_{lm} best matches and computing $fScore$ the sum of the distances of the local feature descriptors of the best matches. In the case of supervised learning, $fScore$ can be used for deciding the class of the given image by choosing an image from the training set with the smallest distance to the unknown image and using its class information to predict the class of the unknown image. In unsupervised categorisation, $fScore$ can be used to find a sorted list of similar images to every given image. The problem of finding the optimal size of the candidate list N_{lm} is studied experimentally in Sec. 4.3.4.

4.2.2 Unsupervised spatial verification and categorisation

In UVOC, it is not possible to match the images with candidate images with known labels. Thus, one needs to solve an image categorisation problem utilising spatial matching information $fScores$ without using labelling information. In the spatial matching, images are compared pairwise resulting in a matrix of pairwise distances. By using the pairwise distances, the images are sorted in ascending order. A list of candidate images contains false matches, and thus, only a subset of the pairwise image distances are important. Therefore, only N_{cand} smallest pairwise distances are kept for each input image. The value of N_{cand} is experimentally explored in the experiment in Sec. 4.3.3. Next, a similarity matrix is constructed by setting the similarity value of image pair i and j as $S(i, j) = N_{cand}/rank(i, j)$, where N_{cand} is the number of images in the list of candidate images after the cut and $rank(i, j)$ is the index of image j in the list of similar images for the image i. The similarity matrix might not be symmetric because the spatial matching phase does not produce symmetric results. To fix the issue, the similarity matrix is made symmetric by refining each similarity value by $S'(i, j) = \max(S(i, j), S(j, i))$. This guarantees that the similarity matrix is symmetric. The final clustering result is computed by using the Normalised Cuts algorithm [82].

4.3 Experiments and results

In the following experiments, the methods using spatial information presented earlier are evaluated in supervised and unsupervised VOC. For the evaluation, Caltech-101 [30] and Randomised Caltech-101 [52] datasets were used. For the supervised VOC, the performance evaluation method defined in Eq. (2.1) was used. For the unsupervised VOC, the method introduced by Tuytelaars et al. [92] (see Eq. (2.6)) and by Sivic et al. [83] (see Eq. (2.2)) were both used and the results are compared. In the supervised VOC, 30 images were chosen randomly for each class for training and 20 images for testing. In the unsupervised VOC, 30 images were chosen from each category with no separate testing set. The Hessian-Affine local feature detector was used for detecting local features and SIFT descriptor for describing the detected local features. The local feature codebook was generated using SOM and the size of the SOM is 200 × 1 if nothing else is mentioned. BoF and BoBoF histograms were both normalised using L2-normalisation defined in Eq. (3.9). For the supervised VOC, 1-NN classification rule is used to predict the class.

4.3.1 Experiment 8: Grid approach

The performance of BoBoF using the grid approach was evaluated using the Randomised Caltech-101 image set. Since it is not obvious how to choose the optimal grid size, different grid sizes were used and the optimal size was chosen for each case. In addition, the size of the BoBoF codebook was optimised experimentally by choosing the optimal size from 10–500. The idea of using grids in BoBoF is similar to the method introduced by Lazebnik et al. [57], but here only one level "hierarchy" is used. In this thesis, the BoBoF histograms are generated from the new BoBoF codebook instead of concatenating several Bag-of-Features histograms together as in [57]. Lazebnik et al. also used SVM to learn object categories, but since the heavy supervised learning is not the topic of this thesis, 1-NN is used.

Results of the experiment are shown in Fig. 4.1. The red solid line with circles denotes the performance with the Bag-of-Features approach presented earlier and the blue line denotes the optimal performance achieved using the BoBoF approach with the optimal size of a BoBoF codebook and the optimal grid size. Fig. 4.1 shows that grid segmentation with a Bag-of-Bag-of-Features codebook does not improve classification accuracy. The BoBoF method using grid segmentation actually performs surprisingly poorly compared with the standard method. Lazebnik et al. [57] were able to improve the classification accuracy significantly using a spatial grid pyramid, but in this experiment, the outcome is the opposite. The reason behind this could be the fact that the size of the BoBoF codebook needs to be larger to separate the images from different categories, but the number of segments is small (1–100) compared to the codebook size (10–500). Thus, the final BoBoF codebook histograms are sparse and obviously more random than in the case of the standard BoF method.

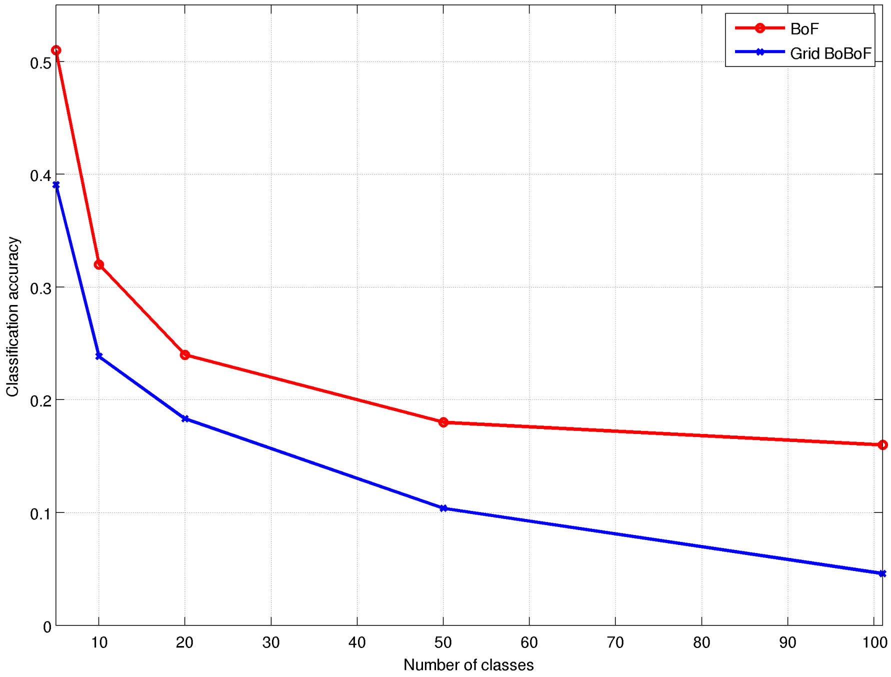


Figure 4.1: Bag-of-Bag-of-Features experiment using the Randomised Caltech 101 image set. Segmentation using various sizes of grids (1×1 – 10×10) and Bag-of-Bag-of-Features codebooks 10–500. The red line is the performance with BoF and the blue line is the performance with BoBoF using grid segmentation.

4.3.2 Experiment 9: Normalised Cuts segmentation

In this experiment, the input images are divided into segments using the Normalised Cuts segmentation algorithm [82]. Then all the segments are described separately using the BoF approach presented in Chapter 3 with BoF codebook histograms, which are used to generate the BoBoF codebook. BoF and BoBoF histograms are normalised using the L2-normalisation defined in Eq. (3.9). Randomised Caltech-101 dataset [52] is used as the image set, and in each experiment, the image set is divided into 30 randomly chosen training images and 20 randomly chosen test images. 1-NN is used as the classification rule. The results are shown in Fig. 4.2.

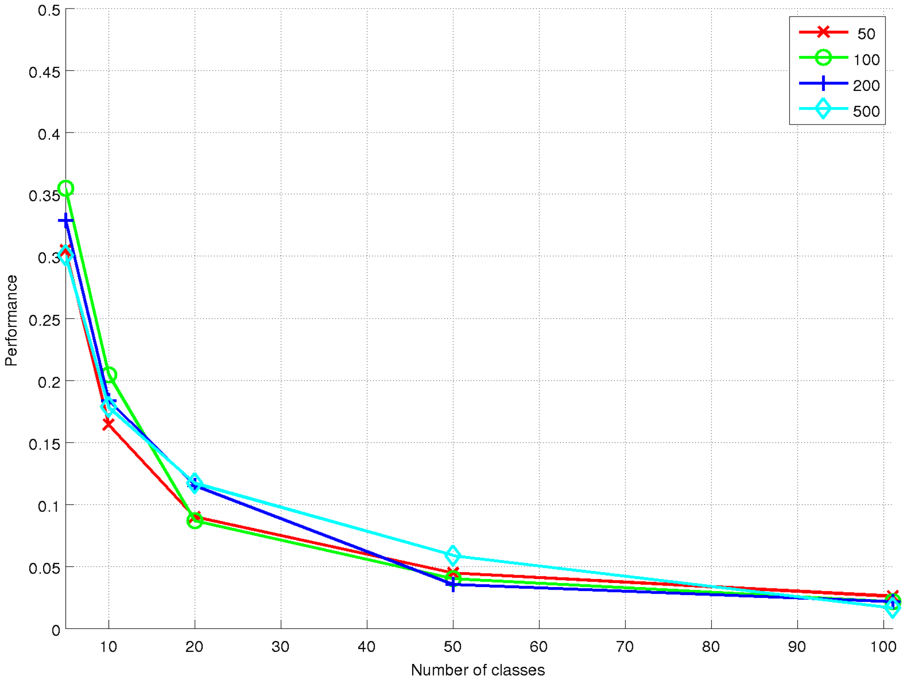


Figure 4.2: 1-NN classification performance using Normalised Cuts with 10 segments with various sizes of Bag-of-Bag-of-Features codebooks denoted with different colours.

Fig. 4.2 shows that the performance is not improved by the usage of BoBoF codebook and Normalised Cuts. The classification performance is significantly lower than that which was achieved in the previous experiment with the BoF approach for which results are shown in Figs. 3.7 and 3.8.

Based on the two previous experiments using the BoBoF approach, it can be concluded that the BoBoF approach is not very suitable for VOC because the BoBoF codebook histograms tend to be very sparse because of the low number of segments and large BoBoF codebook. The size of the BoBoF codebook can be set to small, but then its discrimination power suffers. The number of segments can also be made larger, but then each of the segments captures less local features, and thus, their description suffers, which causes inaccurate descriptions of the segments. However, segmentation has been used successfully in VOC where the goal was to find similar segments from an image set [49].

4.3.3 Experiment 10: Finding value for the number of candidate images for spatial verification

Spatial matching using RANSAC and homography estimation as was presented in Sec. 4.2.1 is computationally expensive. Thus, it is necessary to limit the number of images compared pairwisely using spatial matching even though it can lead to suboptimal solutions. In this experiment, the size of the list of candidate images for the spatial matching is defined, i.e., value for N_{cand} of Sec. 4.2.1, by carrying out an experiment. In the experiment, Randomised Caltech-101 image set was used. 30 images from 101 categories were chosen randomly and then the bag-of-feature approach was performed to obtain BoF histograms for every image. Then, a graph was plotted that illustrate how many of the best matching images based on BoF histograms must be listed in the list of candidate images to be enough confident that there is at least one image from the same category as the given image. The results of the experiment are shown in Fig. 4.3. From the figure, we can see that the confidence of having at least one image from the correct category increases rapidly when the number of candidate images is low (horizontal axes), but after 100 candidate images the level of confidence (vertical axes) begins to saturate. Thus, the size of the candidate set N_{cand} is set to 100, i.e. 100 best matching images based on BoF histograms are chosen for the spatial matching step. According to the figure, for more than 80% of images at least one correct match is within 100 best candidates from the BoF method. According to Fig. 4.3b, only the first 80 images are more likely to be from the correct category compared to the standard BoF approach and the spatial matching begins to saturate around 30 candidate images. Thus, only the 30 best matching images based on spatial distances are kept in the list of similar images in UVOC using spatial information.

4.3.4 Experiment 11: Supervised visual object categorisation

There is no simple method of choosing the optimal value for the number of matching features N_{lm} that is used to define how many of the best matching features are used to compute the sum of feature distances, i.e. $fScore$, which is used to define the distance between the pair of images. Thus, an experiment was conducted to find the optimal value experimentally. Results of the experiment are shown in Fig. 4.4.

From Fig. 4.4, it can be seen that spatial matching can improve the performance significantly and the most suitable N_{lm} is between 4 and 6, thus we chose $N_{lm} = 5$ for the experiment to produce Fig 4.5. This figure shows that the spatial local feature verification can improve the classification accuracy significantly. Spatial verification improves classification accuracy especially when the number of classes increases. Moreover, the performance is improved with both the Caltech-101 [30] and Randomised Caltech-101 image sets.

4.3.5 Experiment 12: Unsupervised visual object categorisation using Randomised Caltech-101

In this experiment, the performance of UVOC was measured using the BoF approach and the spatial matching approaches with Randomised Caltech-101 dataset. In UVOC, label information cannot be exploited in the categorisation process, thus the 1-NN classification rule must be replaced with an unsupervised categorisation method. The final categorisation results were obtained as was presented in Sec. 4.2.2. The standard BoF method was used to generate a sorted list of candidate images for each given image. Then the spatial matching method was used to match all the given images with their $N_{cand} = 100$ best matching images, based on the Euclidean distances of BoF histograms. For each image pair, $fScore$ for $N_{lm} = 5$ was computed (cumulated sum of distances of 5 best spatially matching SIFT descriptors). $fScore$s of pairs of image were used to build a similarity matrix that was given to the Normalised Cuts algorithm [82] to obtain image categories. The test was repeated 10 times with 5, 10, 20, 50, and 101 object categories. In each test, 30 images were chosen randomly from each category. Fig. 4.6 shows that the spatial matching improves the categorisation performance. One can also notice a difference in the BoF performance compared to Fig. 3.13. The reason for the difference is that, in this experiment, clusters were formed using Normalised Cuts as was presented in Sec. 4.2.2 instead of clustering the BoF histograms directly using SOM as in Sec. 3.5.7. The UVOC approach using Normalised Cuts seems to also improve the performance slightly.

4.3.6 Experiment 13: Unsupervised visual object categorisation using Caltech-256

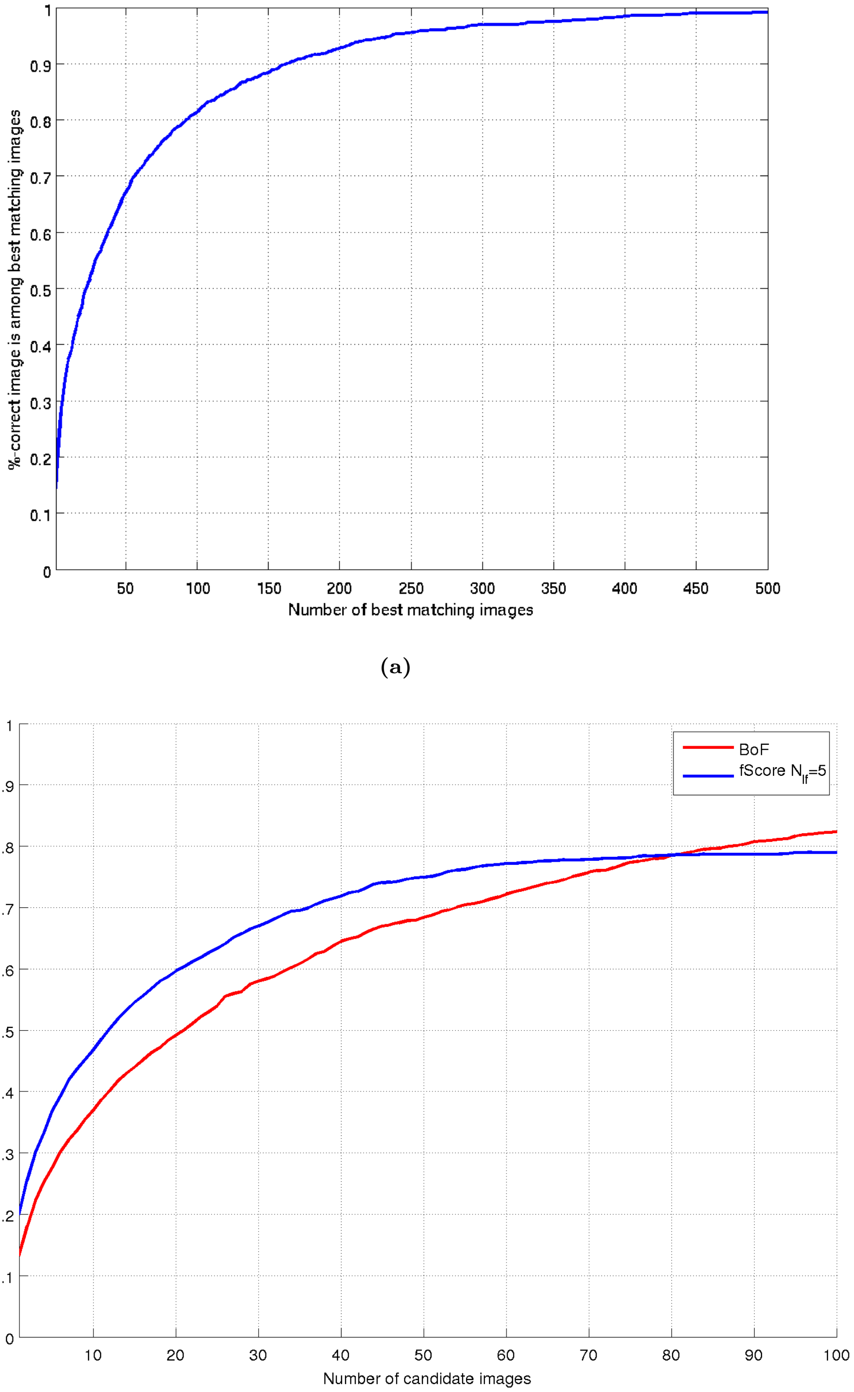
In this experiment, the previous experiment is repeated using Caltech-256 [39] and choosing the same 20 categories as in the experiment presented in Sec. 3.5.6 and in [92]. 30 images were chosen randomly from each category as in the previous experiment. This experiment cannot be compared directly with the experiment presented in Sec. 3.5.6 because in that experiment, all the images were used in the categorisation. Thus, the performance using the BoF approach is not the same. In this experiment, the focus is in the comparison between the BoF approach discussed in Chapter 3 and the spatial matching approach discussed in this chapter.

Input images were described in the same way as in the previous experiments. The Hessian-Laplace detector and SIFT descriptor were used to extract local features. A visual codebook was built using a SOM of size 200 × 1 and codebook histograms were normalised using L2-normalisation. In the spatial matching, as a distance between the pair of images a sum of the $N_{lm} = 5$ best matching local features was used as in the previous experiment. Results of the experiment are shown in Fig. 4.7.

Fig. 4.7 shows that the performance can be improved significantly using the spatial matching. Both performance evaluation methods (i.e. Sivic performance Eq. (2.2) and Conditional entropy Eq. (2.6)) show that the spatial matching improves the results significantly. We cannot compare these results directly with the results obtained earlier because the number of input images are different in the experiments.

4.4 Summary

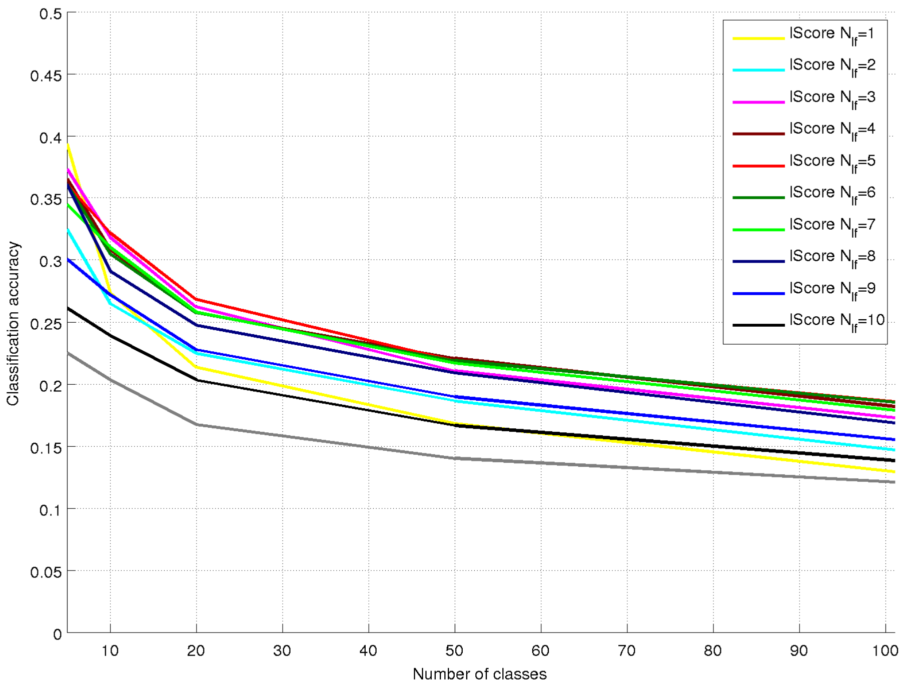
Different approaches were presented for the use of spatial information in the BoF approach. BoBoF methods using the grid approach and segmentation approaches did not improve the performance of the VOC. However, spatial verification of the detected local features improved categorisation performance significantly. In the local feature spatial verification, local features were also matched spatially and then the sum of distances between the descriptors of the best matching local features was used as a measure of how well the two images matched each other. In the supervised VOC using the 1-NN classifier, it is easy to use pairwise image distances, i.e., sum of the distances between the descriptors of the best matching local features, but in UVOC, it is not as straightforward. In this work, pairwise distances were used to rank images, which were then used to compute similarities between the images. This was given to Normalised Cuts [82] to form the final clusters. As we can see from the results shown in Figs. 4.6 and 4.7, the performance of the UVOC is improved significantly by using the spatial matching.



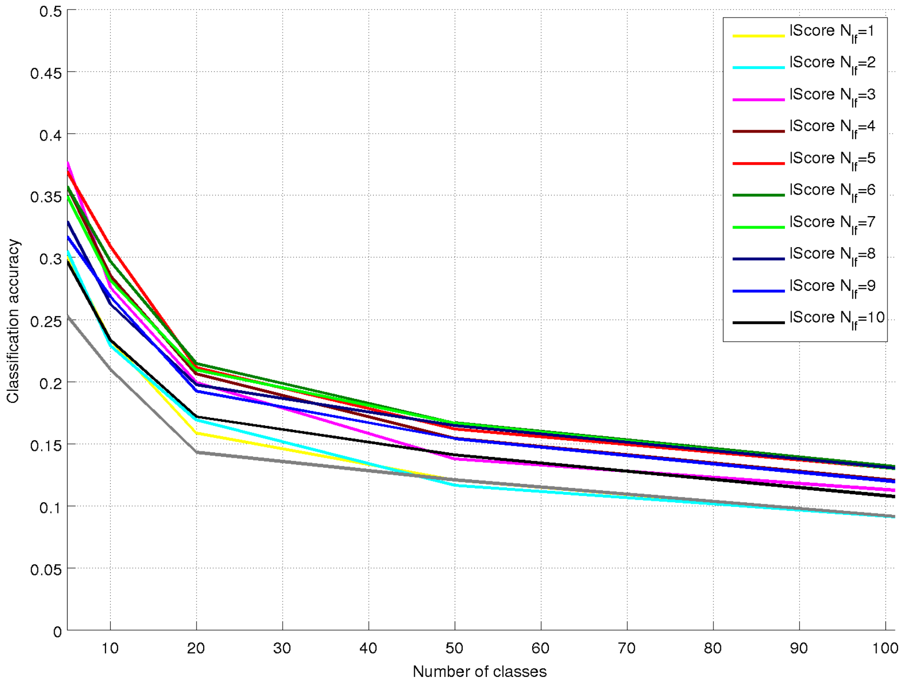
(a)

(b)

Figure 4.3: Candidate image search based on Bag-of-Features using Randomised Caltech-101 with 101 categories and 30 images from each category: (a) Finding optimal number of candidates for spatial matching; (b) Probability of having an image from the same category using BoF and spatial matching.



(a)



(b)

Figure 4.4: Supervised VOC experiment using spatial local feature matching. Performance using only BoF is shown with yellow and spatial matching results are shown with different colours. Two different datasets were used: (a) Caltech-101 dataset [30]; (b) Randomised Caltech-101 dataset [52].

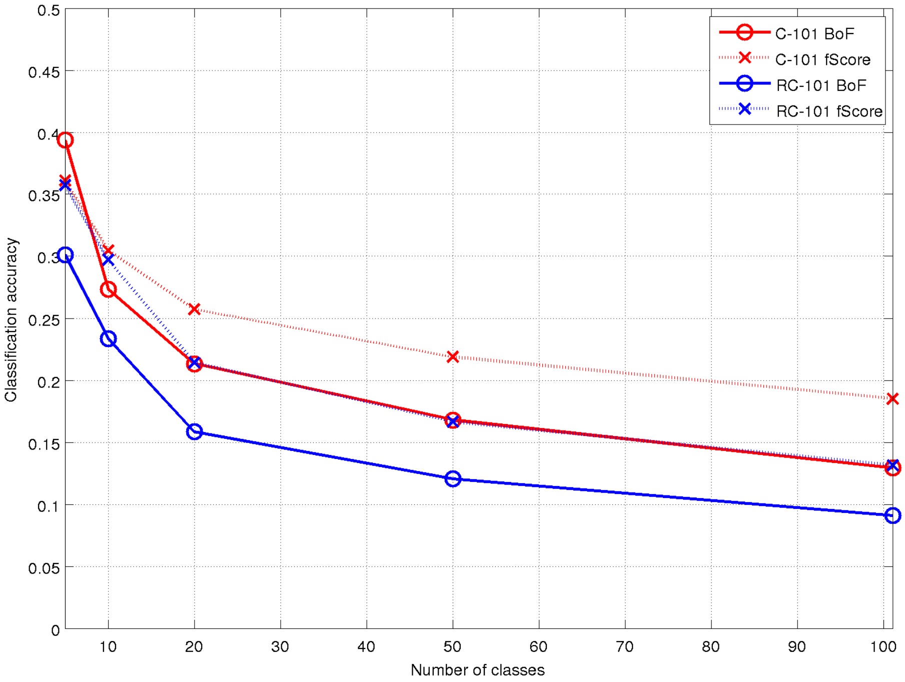
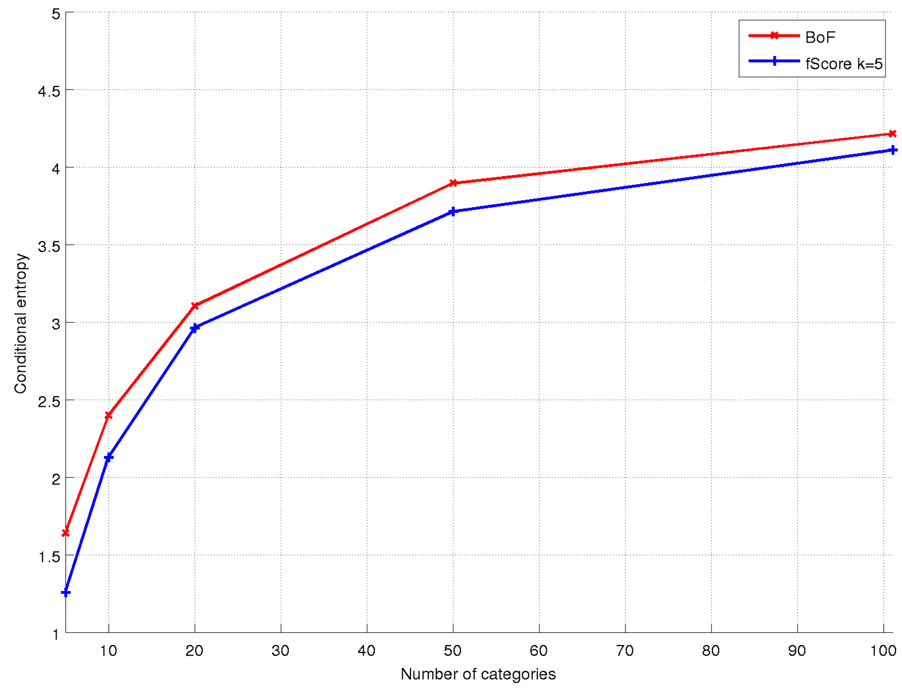
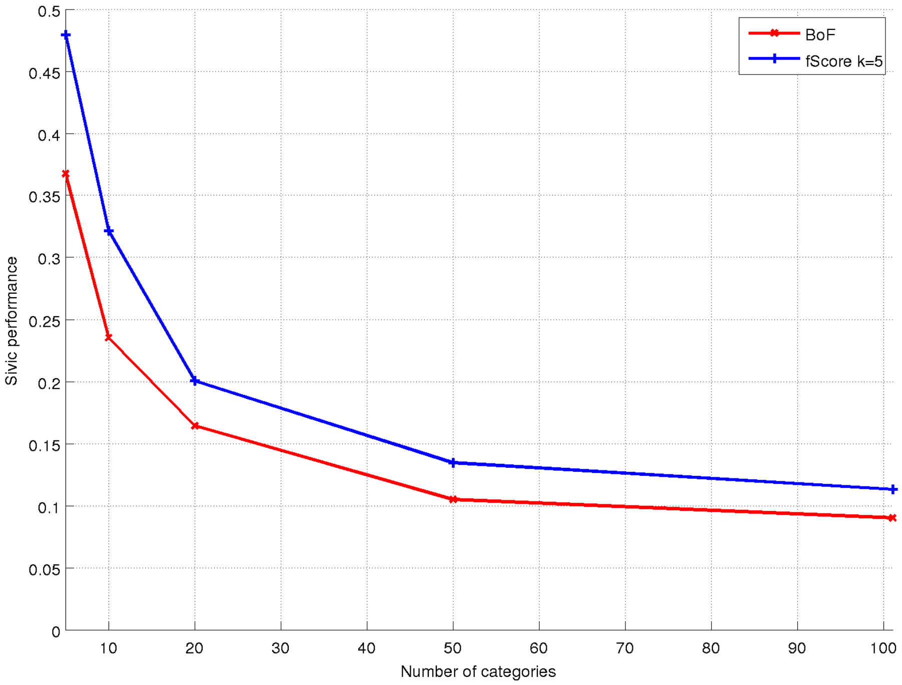


Figure 4.5: Supervised VOC experiment using spatial local feature matching. The blue line denotes classification accuracy using the Caltech-101 image set and the red line denotes results with the Randomised Caltech-101 image set. The solid line is the result using only BoF codebook histograms and dashed line is the performance using spatial verification of landmarks using the distance of 5 best landmarks.



(a)



(b)

Figure 4.6: Results of the UVOC experiment using the Randomised Caltech-101 image set. (a) Conditional entropy Eq. (2.6); (b) Sivic performance Eq. (2.2).

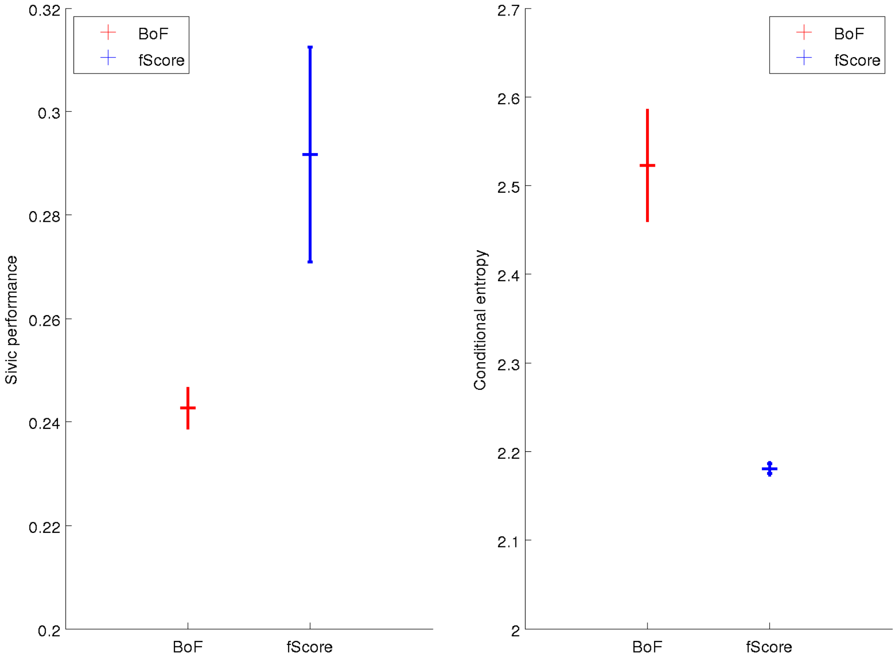


Figure 4.7: Results of the UVOC experiment using the Caltech-256 image set. Performance of the UVOC categorisation using the BoF approach and spatial matching approaches are evaluated using two methods: Left: Sivic performance Eq. (2.2); Right: Conditional entropy Eq. (2.6). Mean performances and standard deviations are shown in the plots.

Chapter V

Visual saliency information in object categorisation

The way people perceive visual information has evolved during thousands of years of evolution. People can recognise thousands of objects quickly and accurately [5]. People perceive tremendous amounts of information through their visual system. However, only a fraction of the information is important. Thus, during the evolution, the vision system has evolved so that the focus can be changed quickly to detect important things. This "pop up" effect is called saliency [42]. The motivation to study the saliency detection in this thesis is that it could be used to improve the VOC categorisation performance by detecting the foreground and using local features that are extracted from the foreground.

Visual saliency detection has been one of the active research fields in computer vision. It has been used in a variety of applications such as automatic image cropping [10], thumbnail generation [61], image collage generation [37], image segmentation [94], image segmentation for VOC [99, 9], VOC [18], and automatic calibration for gaze tracking [89]. Thus, saliency detection has received a great amount of attention from computer vision research in the last few years [19, 46, 100, 101].

In this chapter, a few saliency detectors are introduced together with a new saliency detector based on local features. Performance comparisons between the saliency detectors are made using a data set by Judd et al. [46] and a new Abstract image set introduced in Sec. 2.1.5. The performance of the saliency detector is evaluated using the procedure introduced by Judd et al. [46] which uses a recall curve to evaluate the performance of saliency detectors. Finally, in the last two experiments saliency information is used in the supervised and unsupervised VOC to guide a segmentation method into detecting the foregrounds of the images and discarding all the local features detected from the background.

5.1 Saliency detection methods in literature

In saliency detection, the problem is to predict the saliency of each pixel, i.e. define how much each pixel attracts attention from human observers. These predicted saliency maps are compared with ground truth saliency maps, e.g. by computing a recall curve. This is computed by choosing $p_x\%$ (*Percent Salient*) of the most salient pixels from the predicted saliency map. Then the recall can be computed by computing the number of fixation points inside the thresholded area and then dividing their number with the total number of fixation points. This is repeated with $p_x = 1\%$, 3%, 5%, 10%, 15%, 20%, 25%, and 30% in order to obtain the recall curve. Ground truth saliency maps are obtained by using human participants and an eye-tracker. From eye-tracking data, the fixations are detected by locations where the eye stops for a short moment. Many methods have been introduced in the literature for saliency detection; a few of them are discussed next.

The first detector is the standard Itti&Koch saliency detector [42]. The Itti&Koch detector uses a bottom-up approach to detect saliency. At the lowest level, it uses low-level features such as intensity, contrast, colour opponency, orientation and motion information and stereo disparity if they are available. These low-level features are given to an artificial neural network that combines these features together to detect the visual saliency of an image. The Itti&Koch saliency detector is compared with the state of the detector by Judd et al. [46]. The Learning Predictor (LP) method by Judd et al. combines low level features (*e.g.* Itti&Koch) with mid-level features (*e.g.* horizon) with high-level features (*e.g.* Viola&Jones face detector [97], Felzenszwalb person and car detector [32]). In addition to these features, Judd et al. also founded that central bias is very important in saliency detection. They measured performance of their saliency detection method with, without central bias and using only central bias and found that LP without central bias performs slightly worse than central bias only and LP with central bias performs better than central bias only. For fusing these features together, they are using a support vector machine. Because of high number of features and powerful supervised machine learning it achieves the state of the art results [46].

One of the methods that are relevant to this thesis is published by Cheng et al. [10]. It introduces a foreground/background segmentation method based on saliency detection. For the saliency detection, it uses global contrast information. Cheng et al. introduce a method called Histogram-based Contrast (HC) which computes saliency using a colour histogram of the image in the L*a*b colour space. To compute the saliency value for a pixel, the method computes colour contrast between all other pixels and sums them together. This is computationally heavy, and thus, it can easily be made faster by realising that all the pixels with the same value are given the same saliency value. Thus, saliency can be computed for each different pixel value once and those that are computed can be used globally since the method does not use spatial information. Cheng et al. improved this method by also taking spatial relationships into account. The improved method is called Region-based Contrast (RC) and it uses the segmentation method by Felzenszwalb et al. [33] and then computes saliency values inside segments in a similar manner as HC. To emphasise the spatial relationship between the pixels, they also used the spatial distances of the pixels in the saliency estimation. Cheng et al. also introduced a Region-based Contrast Cut (RCC) segmentation method that uses saliency information from RC to initialise the GrabCut segmentation method [78]. The GrabCut algorithm seeks segments iteratively, but in the original paper, the object in the foreground had to be marked manually with a bounding box. Cheng et al. use saliency information to choose the foreground (more salient pixels) and background (less salient) automatically. The segmentation application using saliency information makes the method particularly interesting for this work.

Examples of the predicted saliency maps using the methods presented earlier and the ground truth with the original image are shown in Fig. 5.1. The saliency detectors presented in this section are evaluated using the data set introduced by Judd et al. [46] and the Abstract image set presented in this thesis, in Sec. 5.4.2.

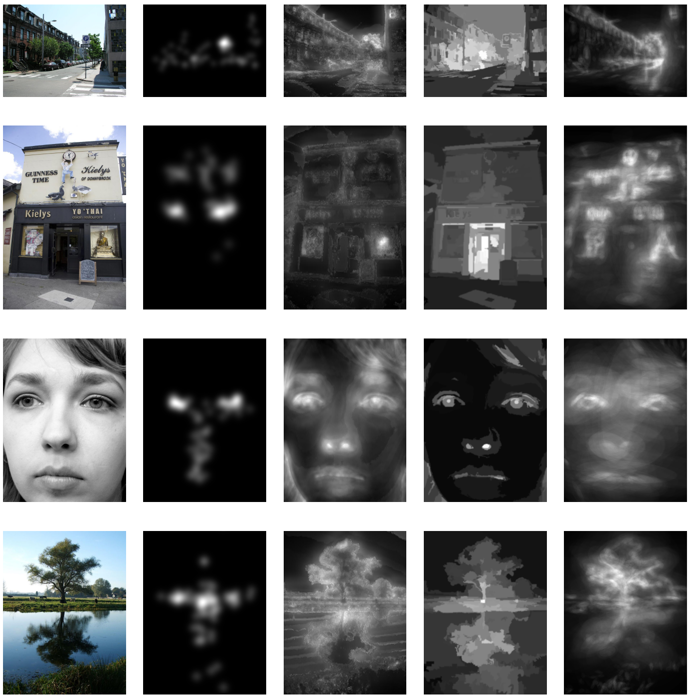


Figure 5.1: Outputs of the saliency maps using local feature detectors. From the left: Original image, ground truth, Learning Predictor (LP) by Judd et al. [46], Region-based Contrast (RC) by Cheng et al. [10] and Hessian-Affine (HA) presented in the thesis. The data set is introduced by Judd et al. [46].

5.2 Saliency detection using local feature detectors

In this section, a comparison of local feature detectors is made from the point of view of which one of the local feature detectors captures the most of the salient ground truth region. Different local feature detectors were used for detecting regions and then the detected regions were converted into saliency maps as presented in Algorithm 5.1. The algorithm takes the regions of the detected local features $\boldsymbol{L}$, an input image $\boldsymbol{I}$ as inputs. At first, the saliency $\boldsymbol{A}$ of each pixel is set to zero. Then, the saliency values of the pixels, $\boldsymbol{x} = (x, y)$, belonging to the region i of detected local features, are increased by one. This is repeated for every detected local feature. Finally, the saliency map is normalised by dividing the saliency values by their sum. A few examples of predicted saliency maps are shown in Fig. 5.2.

Algorithm 5.1 Local feature regions to saliency maps

```
Require: $\boldsymbol{L}$, $\boldsymbol{I}$
  $\boldsymbol{A}_{1,\dots,width(\boldsymbol{I}),\ 1,\dots,height(\boldsymbol{I})} \leftarrow 0$ // $I$nitialise saliency map with zeroes
  for $i = 1, \dots, numberOfFeatures(\boldsymbol{L})$ do
    // $S$elect all the indexes of pixels belonging region $\boldsymbol{L_i}$
    // $a$nd store them in $\boldsymbol{x}_1, \dots, \boldsymbol{x}_n$ where $\boldsymbol{x}_j = (x, y)$
    $\boldsymbol{x} \leftarrow getPixelsOfRegion(\boldsymbol{L_i})$
    for $j = 1, \dots, length(\boldsymbol{x})$ do
      $\boldsymbol{A}_{x_j,y_j} \leftarrow \boldsymbol{A}_{x_j,y_j} + 1$
    end for
  end for
  $\boldsymbol{A} \leftarrow \boldsymbol{A} / \left(\sum_{i=1}^{length(\boldsymbol{A})} \boldsymbol{A_i}\right)$
  return $\boldsymbol{A}$
```

Fig. 5.2 shows how the saliency maps generated using different local feature detectors differ from each other. The Harris-corner (Eq. (3.3)) detector based detectors, the Harris-Laplace and the Harris-Affine detect the highest number of local features, and thus, the saliency maps are covered with salient pixels. Additionally, the Hessian-matrix (Eq. (3.7)) based local feature detectors detect a large number of features. However, saliency maps generated from Hessian-Laplace and Hessian-Affine local features seem to also cover more non-salient areas (i.e. black pixels). In the saliency prediction experiment, the performance of each saliency predictor is evaluated using a recall curve. Experiment and results are in Sec. 5.4.1.

5.2.1 Predicting the saliency of local features

Whether it is possible to learn a model of a salient local feature was then studied as this could be used to select only important ones among hundreds or even thousands of local features. Such a model could be used directly in VOC to choose only the important, i.e. salient, local features to improve the categorisation performance.

In Fig. 5.2, it is shown that local feature detectors can capture local features from salient regions in the image. To acquire more benefit, one can try to learn a model of a salient local feature. To learn which of the local features are salient, three different approaches were tested: i) the codebook based approach; ii) the regression model Artificial Neural Network (ANN); iii) and the nearest-neighbour method using a kd-tree. These three methods for local feature saliency prediction are evaluated in the experiments section. Saliency value for a local feature is computed from the ground truth saliency map. At first, the detected region of a local feature is projected onto the ground truth saliency map and then the mean saliency value of the detected region is computed. The mean saliency value of the region is used as the saliency of the detected region (i.e., the region of the detected local feature). An example of a saliency of ten detected local features is shown in Fig. 5.3. It shows a ground truth saliency map where the brightness defines the saliency of the pixel. Detected region are marked with yellow ellipses and saliency value of the detected regions is marked above each region using red.

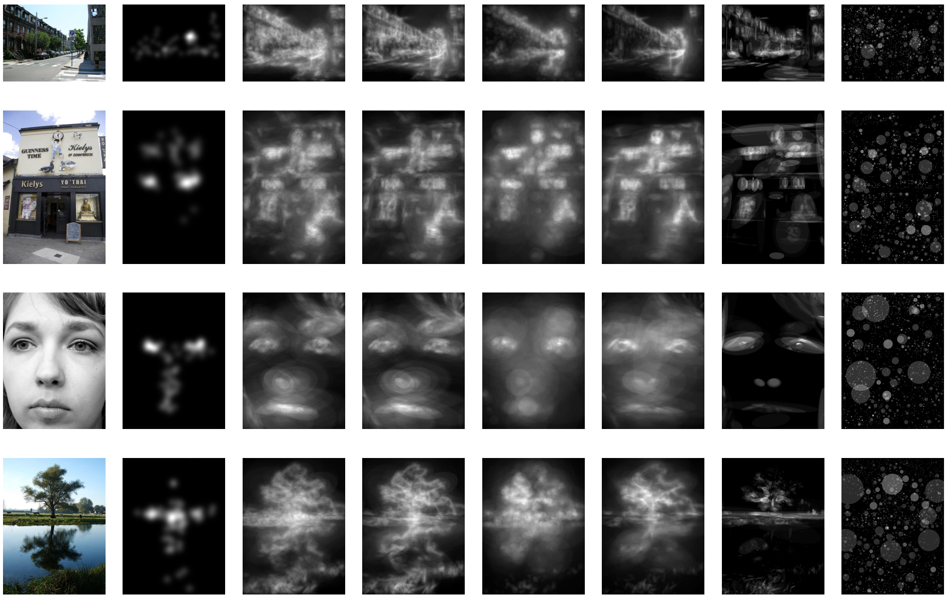


Figure 5.2: Outputs of the saliency maps using local feature detectors. From the left: Original image, ground truth, Harris-Laplace, Harris-Affine, Hessian-Laplace, Hessian-Affine, MSER and SIFT. The data set is introduced by Judd et al. [46].

5.3 Improving object category detection using salient regions

Figs. 5.1 and 5.2 show that it is possible to detect salient regions, e.g. objects, in the images, and thus, it could be possible to benefit from saliency maps. Judd et al. [46] used object detectors to improve saliency detector performance. In this section, the goal is the opposite, the saliency information is used to improve the VOC performance. However, both approaches assume that the object of interest is salient.

Saliency information can be used with segmentation to choose an important region from an image to be used in categorisation. If the important region can be detected successfully, the image can be described more accurately because the codebook histogram would contain only hits from the foreground (see, e.g., Fig. 3.7), and thus, the categorisation performance should be better. Cheng et al. [10] have developed a method that uses saliency information to detect important area from an image. Here, segmentation results are used to detect the foreground from the images. Example results of the detected salient segments are shown in Fig. 5.4. The figure shows how the RCC segmentation method [10] can detect foregrounds from the Randomised Caltech-101 images even though some of them have challenging backgrounds. The foregrounds, especially of the first three images, are detected very well. The RCC segmentation fails on the car side image (Fig. 5.4j) because the saliency detector detects that the mountain is the most salient region as the mountain differs the most from its surroundings. According to the color contrast differences, it differs the most from the rest of the image, and thus,

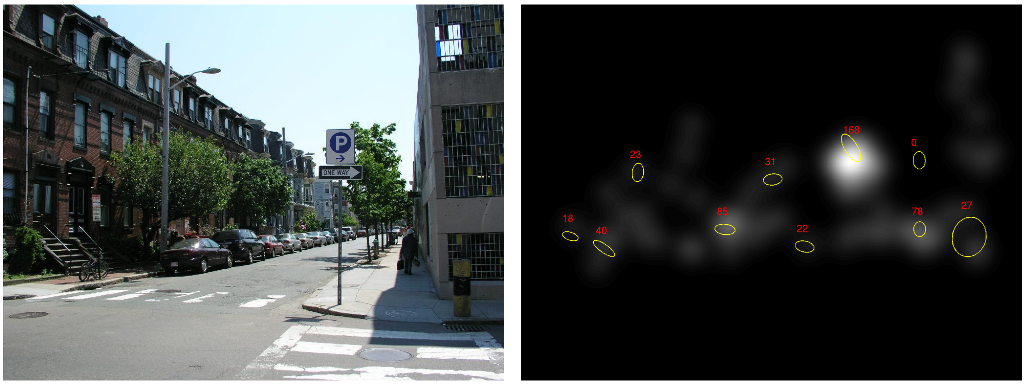


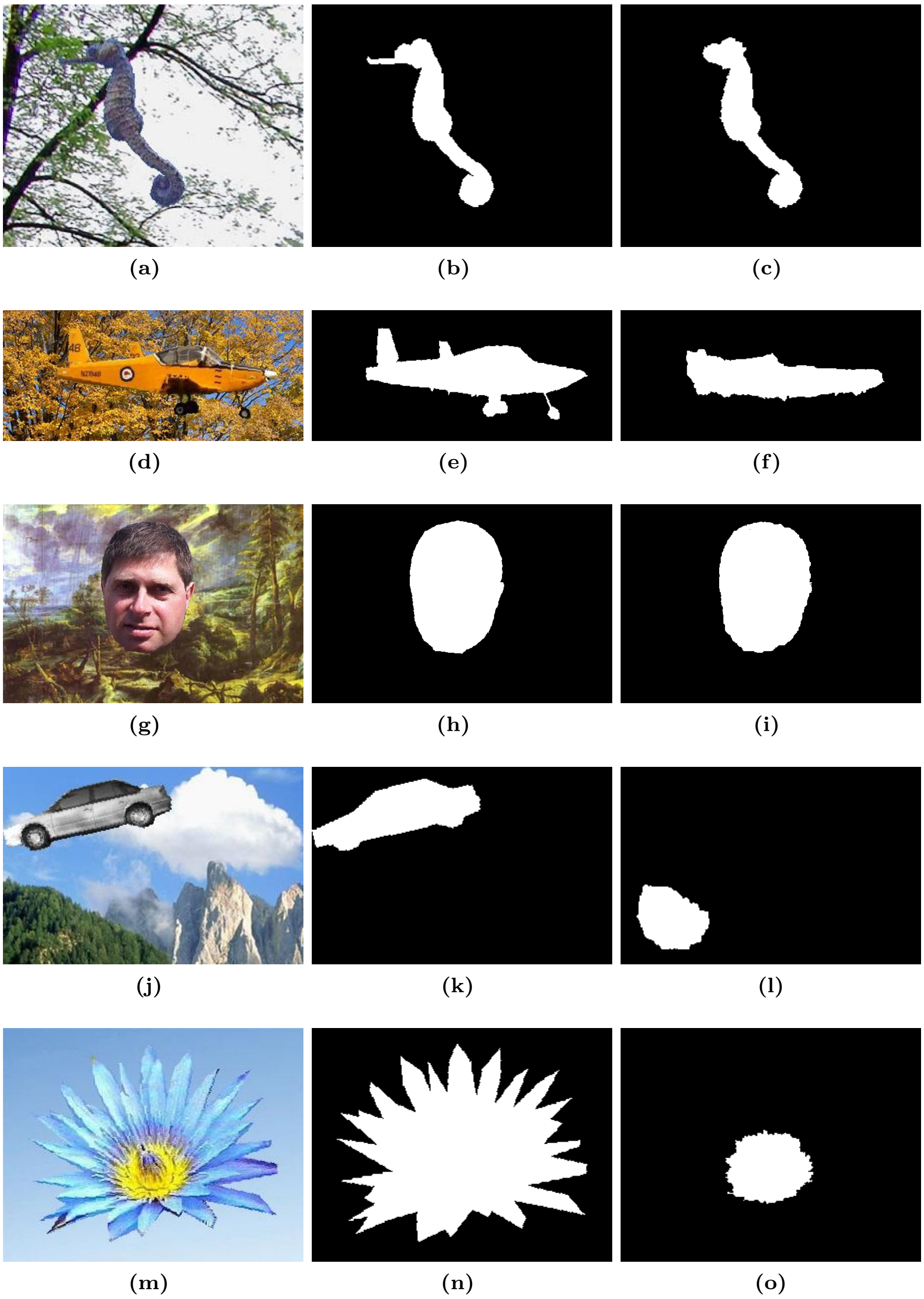
Figure 5.3: Saliency values for detected Hessian-Laplace local features. Ten local features are shown with yellow ellipses and saliency of the region is shown with a red number above the region. The intensity displays the ground saliency value in the image on the right.

it is incorrectly detected. There is a similar problem with the lotus image in Fig. 5.4m. The inner part of the lotus is labelled as salient, but the leaves of the flower do not differ significantly from the background, and thus, they are considered as non-salient. When this is given to the GrabCut segmentation algorithm, it segments the inner part of the flower and it is used as the predicted foreground. In this experiment, the data set is very challenging because it is artificial and the backgrounds can also contain salient objects. However, RCC is able to find the foreground in many cases, and thus, it can be used to detect foregrounds for VOC. An experiment using RCC predicted foregrounds is introduced in the following section.

5.4 Experiments and results

In this section, the performances of different saliency detectors are compared with each other, and then, saliency detection is used to detect the foreground from the images to filter out local features that come from the background. The effect of this local feature filtering approach is studied in the last two experiments.

In the saliency detection experiments, a recall curve is used to compare the performance of different saliency detectors. The evaluation method follows the same procedure used by Judd et al. [46].



(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

(j) (k) (l)

(m) (n) (o)

Figure 5.4: Example outputs of Region-based Contrast (RCC) segmentation by Cheng et al. [10]. Images are from Randomised Caltech-101 [52]: The left column shows the original images; The middle column shows the ground truth foregrounds; The right column shows the RCC predicted foregrounds.

5.4.1 Experiment 14: Comparison of local feature detectors in saliency prediction

In this experiment, saliency detectors based on local feature detectors, which were presented in Sec. 5.2, are compared quantitatively using the dataset presented by Judd et al. [46]. In addition to the presented saliency detectors, we have an inter-subject which tells how well people can predict saliency maps, i.e. how consistent are the saliency maps generated from different subjects. We also have the central bias saliency detector where the saliency of the pixels are defined as the inverse of the distance from the centre of the image.

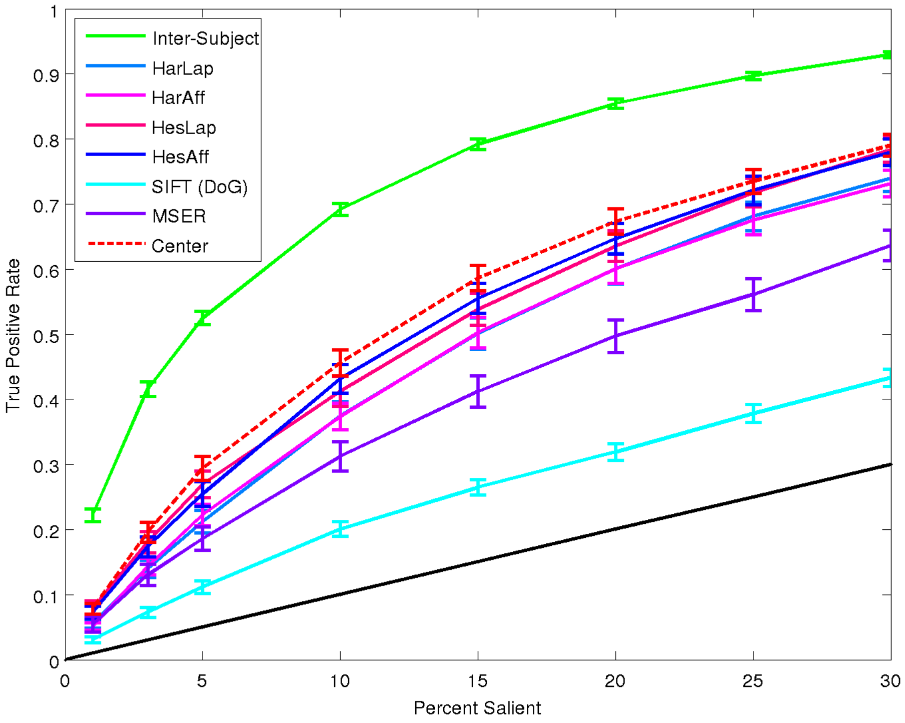


Figure 5.5: Recall curves for saliency detection using different local feature detectors and the natural images dataset introduced by Judd et al. [46].

Fig. 5.5 shows that the Hessian-Laplace and Hessian-Affine local feature detectors perform the best. The Harris-Affine and Harris-Laplace local feature detectors perform slightly worse than Hessian-matrix based detectors. The difference between the two Hessian-matrix based detectors is small and Harris-corner based detectors perform equally. This is not very surprising because the saliency maps shown in Fig. 5.2 are visually very similar. MSER and Difference-of-Gaussian detector used in SIFT do not perform as well. However, all the detectors are far from the inter-subject performance and even behind the centre biased saliency detector.

5.4.2 Experiment 15: State-of-the-art saliency detection

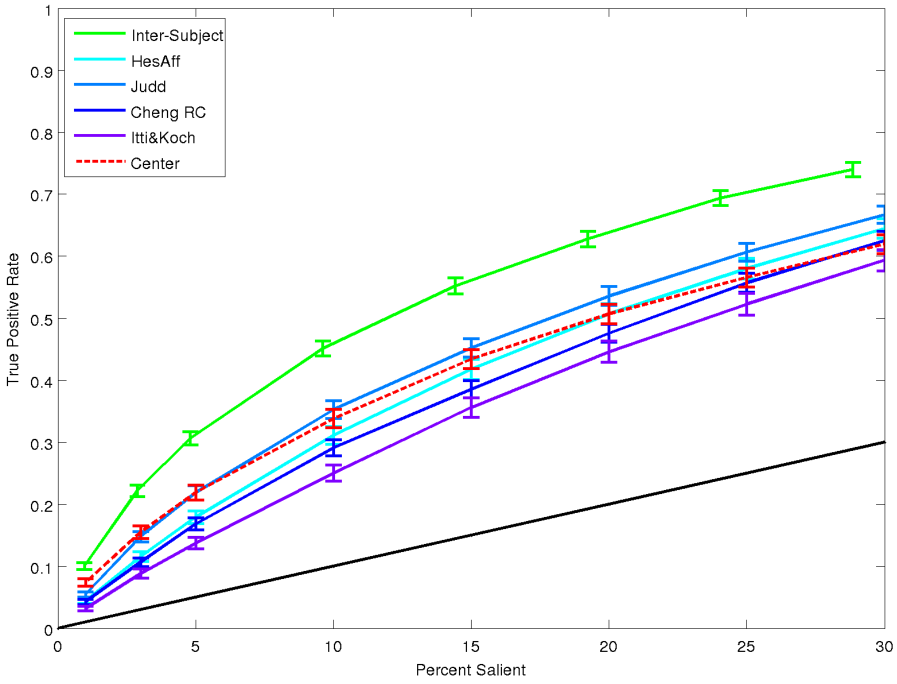
In this experiment, the performance of state-of-the-art saliency detectors presented in Sec. 5.1 is evaluated using two image sets: the Judd et al. image set [46] and the Abstract image set presented in Sec. 2.1.5. The saliency detector by Judd et al. is trained using 903 training images from their image set, the rest of the saliency detectors do not need training data. In addition to the current state-of-the-art detectors, Hessian-Affine local feature detector based saliency detector that was introduced in this thesis was also compared to the other detectors.

Results of the saliency detection experiments are shown in Fig. 5.6. The state-of-the-art saliency detector by Judd et al. [46] seems to perform very well with both of the image sets. The Hessian-Affine based local feature saliency detector had the second best performance. The Region-based Contrast (RC) method by Cheng et al. [10] performed slightly better than the Itti&Koch detector [42] with the Abstract image set, but with the image set by Judd et al., the RC and Itti&Koch detectors performed equally well. However, the RC saliency detector's performance was worse than the Hessian-Affine saliency detector with both image sets. Although, the order of the best performing saliency detectors is very similar with both image sets, one can notice that the performance is worse for all the saliency detectors with the Abstract images.

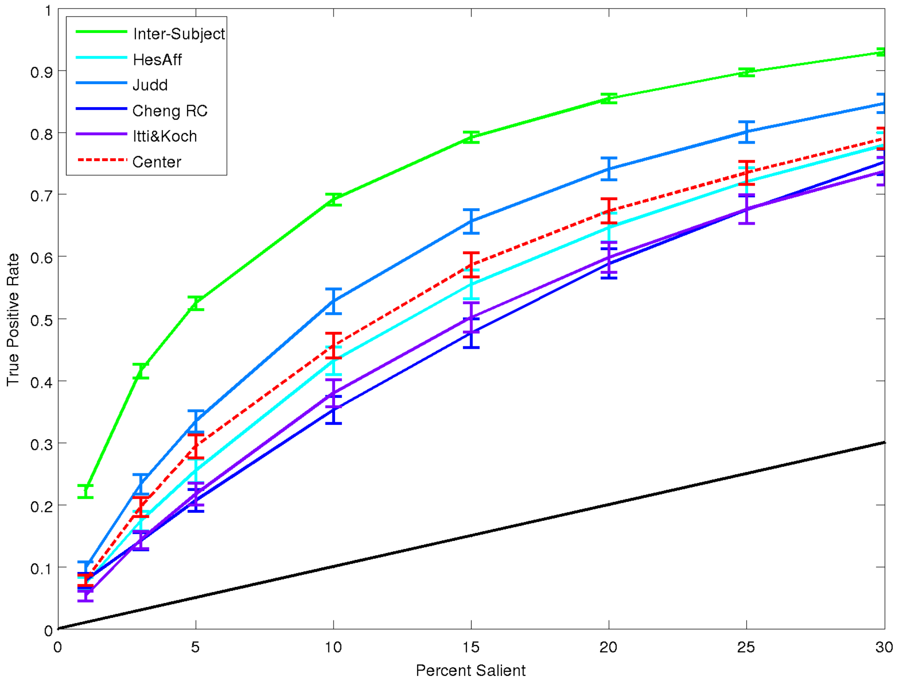
5.4.3 Experiment 16: Salient local feature prediction

The ability to predict saliency of the local features is tested using the data set introduced by Judd et al. [46]. Their experiment procedure is followed by choosing the same 903 images for training and 100 images for testing. The training data was collected by extracting local features using the Hessian-Affine local feature detector and the SIFT local feature descriptor. Next, the saliency of each local feature was computed as described in Sec. 5.2.1. Thus, the training data was a set of local feature descriptors and saliency value of each local feature.

To learn to predict which of the local features are salient, three different approaches were tested in the experiments: i) Codebook based saliency prediction; ii) Regression modelling using Artificial Neural Network (ANN); iii) Nearest neighbour method using a kd-tree. In the codebook approach, a BoF codebook was built and then saliency of each local feature was defined by computing the mean of ground truth saliency of the training set local features. In the ANN approach, a multi-layer feed-forward network was trained using the Scaled Conjugate Gradient algorithm [71]. Only a subset of the training data was used because of memory capacity limitations. The training data was divided into tree parts: training (70%), validation (15%) and testing (15%). In the nearest neighbour method, the saliency of the test set local features were simply predicted by choosing the saliency of the most similar local feature in the training set. The results for the experiments are shown in Fig. 5.7. The horizontal axis determines the ground truth saliency value for a local feature and the vertical axis determines the predicted the saliency value of the local feature. Therefore, if the saliency is predicted correctly, both should be plotted on the black line an equal distance away from both axes. However, the results was the same for all three approaches: all methods failed to predict saliency of the testing set local feature.

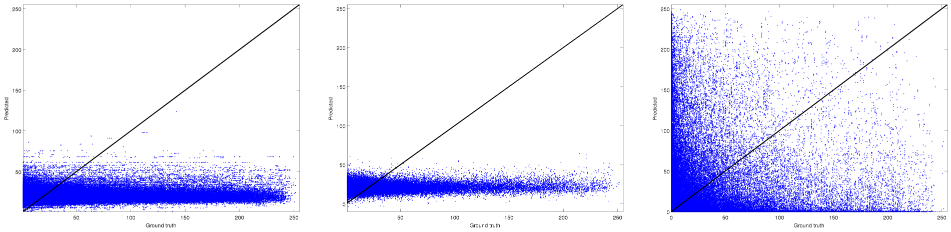


(a)



(b)

Figure 5.6: Recall curves for the saliency detection experiments: (a) Abstract image set; (b) Judd et al. image set [46].



(a) **(b)** **(c)**

Figure 5.7: Saliency prediction using a) Codebook; b) Artificial Neural Network; c) Nearest neighbour using a kd-tree. The horizontal axis is the ground truth saliency value and the vertical axis is the predicted value. The black line denotes the correct prediction.

The reason for this behaviour can be found in Fig. 5.8, where the mean and the standard deviation of each saliency value of a matching code is plotted in the codebook approach. We can see in the figure that the deviation is very high. For a group of similar local features, i.e. local features which match the same code, saliency value varies considerably. Thus it is difficult to learn the saliency of different codes as could be seen in Fig. 5.7a.

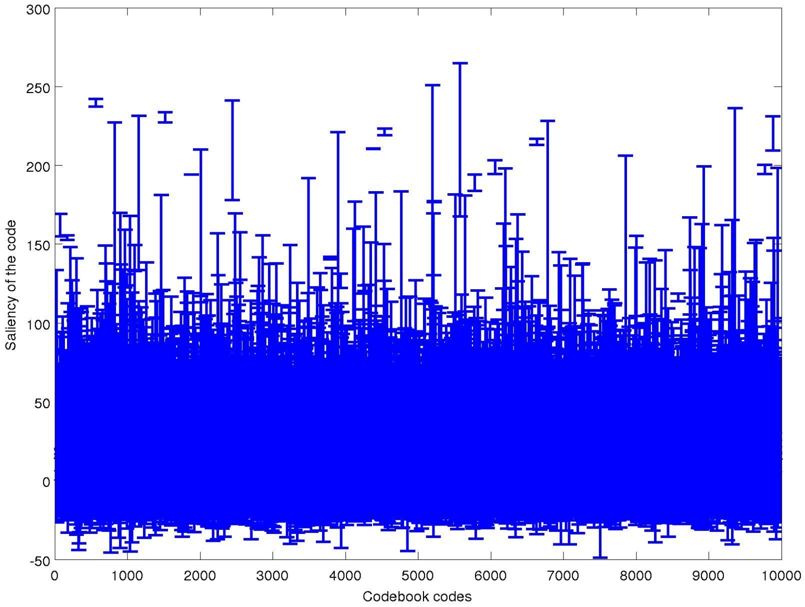


Figure 5.8: Mean and standard deviation for the codebook code saliency.

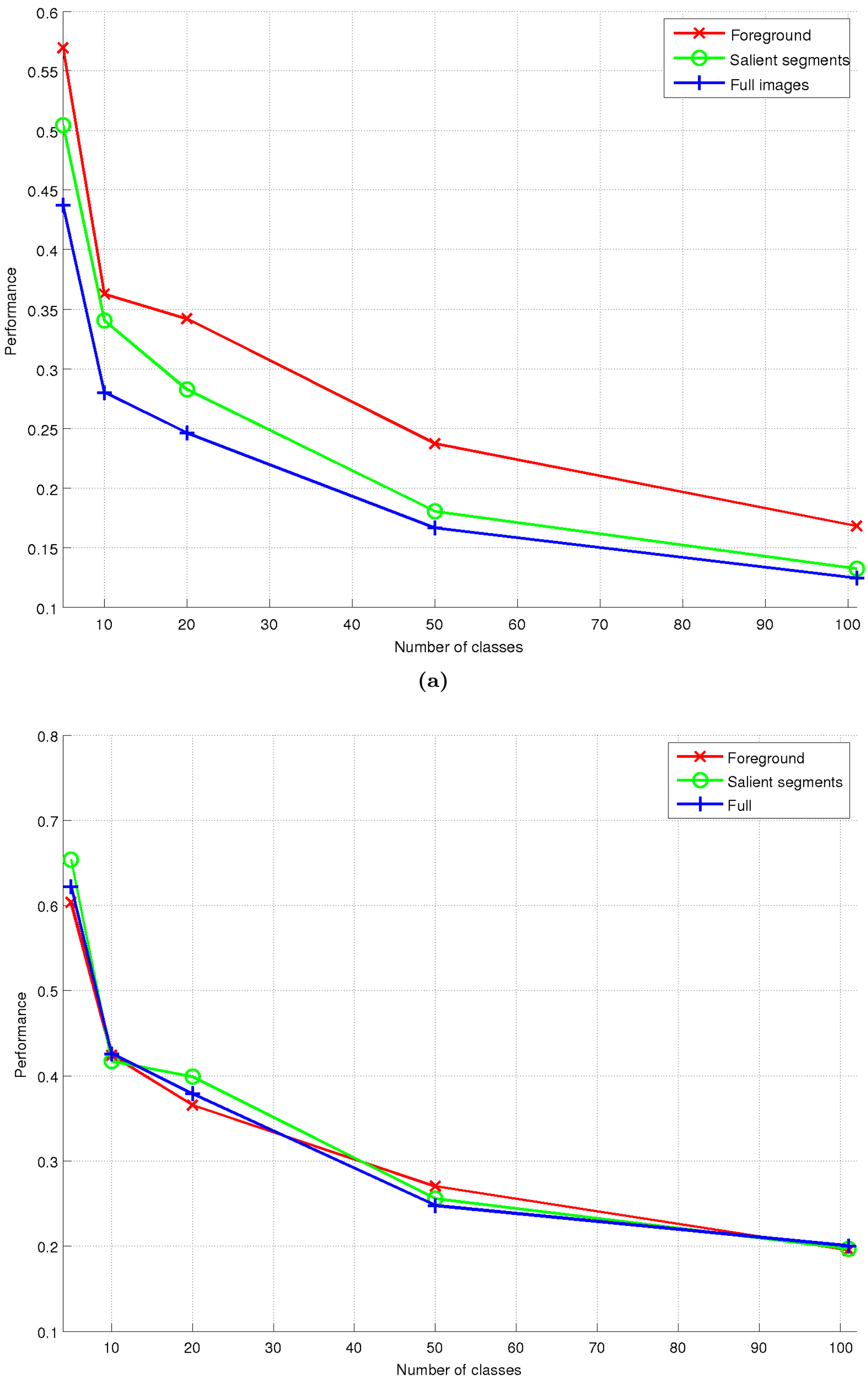
5.4.4 Experiment 17: Improving VOC performance using salient foreground detection

This experiment studies how the automatic foreground detection using saliency information and segmentation affects the performance of supervised VOC. In this experiment, the Caltech-101 [30] and Randomised Caltech-101 [52] image sets were used. The experiment was divided into three test cases: i) Only the ground truth foregrounds of the images are used; ii) Automatically detected salient segments are used; iii) Full images are used. The motivation in the experiment was to experimentally study the feasibility to use saliency guided segmentation to detect foreground from the images. According to Fig. 3.7 there is room for improvement (gap between r-Caltech (Full) and r-Caltech (Fg)). In this experiment, the image classes were learned and the images were classified using the BoF approach with the Hessian-Affine feature detector and SIFT descriptor in supervised manner. For generating the codebook, a 200 × 1 SOM was used. The codebook feature histograms were normalised using the L2-norm. Finally, the given images were classified using the 1-NN classification rule using 30 training images and 20 test images from each class. Supervised learning was used to simplify the experiment set-up. The results are shown in Fig. 5.9.

Fig. 5.9 shows that the classification performance can be improved by choosing local features only from the salient segments of the images if the background does not contain relevant information about the object in the foreground as in the case of Randomised Caltech-101. However, the performance is inferior to the performance using local features only from the ground truth foregrounds. Randomised Caltech-101 image set is quite challenging especially for saliency detectors because the backgrounds can also contain salient objects. With Caltech-101, the salient segment detector (RCC [10]) does not improve classification performance significantly. In the Caltech-101 dataset, backgrounds contain important information about the object as was found in the earlier experiment in Section 3.5.1 and in Fig. 3.7. In a more realistic case, salient segment detection could improve classification performance because it is very likely that the backgrounds have more variability than in the case of original Caltech-101.

5.4.5 Experiment 18: Improving UVOC performance using salient foreground detection

The UVOC experiment was carried out using the Caltech-256 [39] image set using the same test experiment as Tuytelaars et al. [92] used in their survey of UVOC methods. Images were selected from the same 20 categories that were used in [92]. The experiment is a comparison between three different categorisation methods: SOM [53], Neural Gas [62], and k-Means using the full images and only the salient segments detected using RCC by Cheng el al. [10]. The size of the SOM, number of nodes in Neural Gas and number of clusters in k-Means was set to the number of ground truth categories. Then the images were categorised using the clustering method with 20 clusters and the performance was measured by computing conditional entropy by Tuytelaars et al. [92] as in Eq. (2.6) and using the performance measure by Sivic et al. [83] defined in Eq. (2.2). Lower entropy means higher performance. Results of the experiment comparing performance of the UVOC using full images and only the salient segments are shown in Fig. 5.10.



(a)

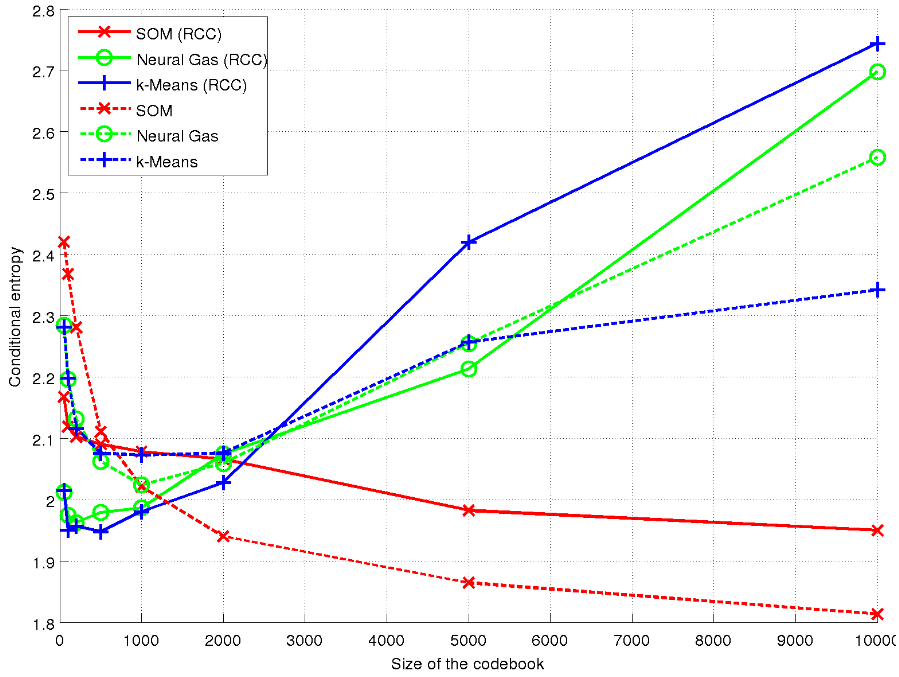
(b)

Figure 5.9: Classification performance using the salient segment detector (RCC) [10]: (a) Randomised Caltech-101; (b) Caltech-101. Performance using only local features from the foreground (red cross), from the salient segment (green circle), from whole image (blue plus).

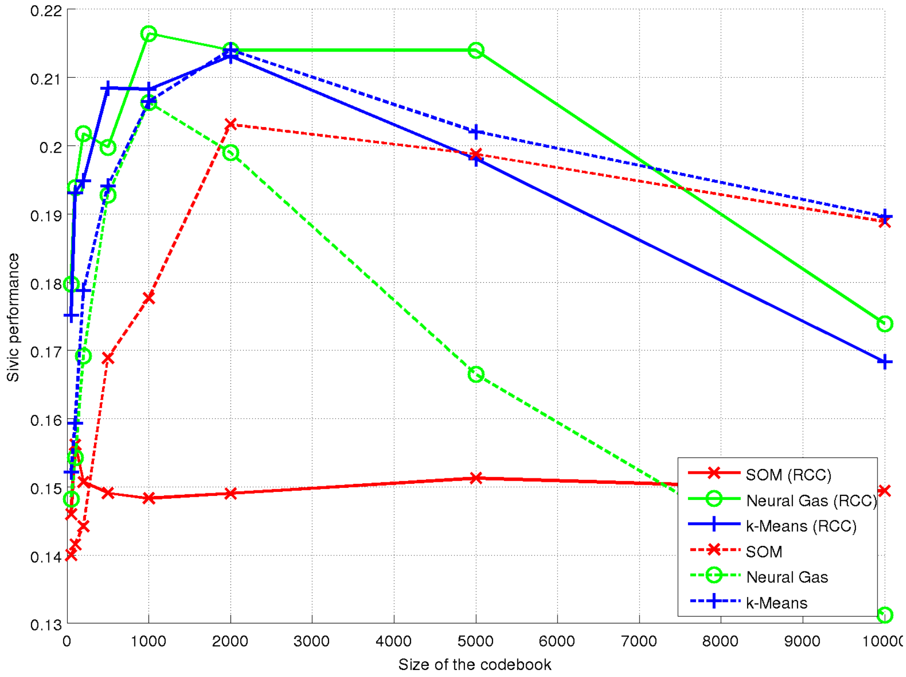
Results are slightly mixed, which is caused by the fact that the number of images in different categories varies considerably and the Sivic performance measure takes on average categorisation accuracy over the categories, whereas the conditional entropy is computed for the whole data set without balancing the categories. In Fig. 5.10, we can see that the categorisation performance was improved with RCC foreground detection when the size of the codebook is small (less than 500 words for SOM and less than 1000 words for k-Means and Neural Gas). When the size of the codebook was increased, the RCC foreground detection did not improve the categorisation performance. However, with k-Means and Neural Gas categorisation, the best performances were achieved with a small codebook and RCC foreground detection whereas with SOM categorisation and overall, the best performance was achieved with a larger codebook and without RCC foreground detection.

5.5 Summary

In this chapter, different local feature detectors were first compared in saliency prediction task. The idea behind the experiment was to study if the local feature detectors can capture local features from the salient objects. In the experiment in Sec. 5.4.1, the Hessian-Affine local feature detector performed the best. In the following experiment in Sec. 5.4.2, the saliency detector based on Hessian-Affine local features was compared with the current state-of-the-art saliency detectors. The current state-of-the-art detector by Judd et al. [46] performed the best, the saliency detector based on Hessian-Affine was the second best performing only slightly worse than the detector by Judd et al. This promising result lead to the next question; Is it possible to learn a model of salient local feature? To answer this question, the following approach was used to learn a model: compute the mean of ground truth saliency value of each detected local feature region and use it with the local feature descriptor as training data. Then a few methods were used to learn the model: codebook, neural network and nearest neighbour method, but all of the methods failed. The reason for the failures could be the high variation in saliency values for similar local features (which is shown in Fig. 5.8). Finally, in the last experiments, saliency information was used to improve the VOC performance. Saliency information was used to help the segmentation method to segment the foreground from the images. These foregrounds were used to filter out all the local features detected from the background. The conclusion from the experiment was that the foreground detection can improve the categorisation performance if the backgrounds of the images do not contain valuable information about the object category. Thus, foreground detection improved results with the Randomised Caltech-101 dataset, but with the original Caltech-101 dataset, the effect on the performance was insignificant. In the unsupervised VOC using the foreground segmentation based on saliency information, the performance was improved especially with small codebooks, e.g., for SOM the performance was improved when the size of the codebook was less than 500 and for Neural Gas and k-Means, the performance was improved with smaller than 1000 codes codebooks. However, with larger codebooks the improvement was small or even negative. It is also necessary to note that in Caltech-256, the background can provide important information about the foreground.



(a)



(b)

Figure 5.10: Categorisation performance of the UVOC test using Caltech-256 comparing BoF approaches with full images (dashed lines) and the only salient segments using RCC [10] (solid lines) were measured using: (a) Conditional entropy [92]; (b) Sivic performance [83].

Chapter VI

Discussion and future work

The main research question in the thesis was that is it possible to learn to detect visual object categories in the same manner as people using unsupervised learning. In this thesis, the approach to learning the object categories was to use the popular Bag-of-Features (BoF) approach and self-organisation. According to the results achieved in the work, it is possible to learn object categories, but the categorisation accuracy, or performance, is significantly lower than in supervised Visual Object Categorisation (VOC). However, unsupervised learning was chosen because supervised learning needs known data, i.e. images with ground truth labels, which can be laborious to obtain especially when the number of categories is increased to thousands of categories. Moreover, it is computationally and memory-wise demanding to train a classifier for thousands of categories [17]. Unsupervised learning does not need known data for training classifiers or class models, and thus, it can be used to categorise objects from thousands of categories. It can also be used when images do not have clear categories. For example, in the case of holiday images, it can be difficult to describe the content of the image with words.

The second research question concerned how spatial information can be used in Unsupervised Visual Object Categorisation (UVOC) using the BoF approach to improve the categorisation performance. In this thesis, the spatial information was used to verify the local feature matches. Then, a cumulative sum of distances of matching local feature descriptors was used to define how well two images match each other. In the supervised VOC, this information was used directly in the 1-NN classifier, which was shown to improve the classification performance significantly. However, in UVOC pairwise image distances could not be used directly. Thus, an alternative approach was used. At first, for each given image, a ranked list of similar images was generated based on pairwise image distances. Then, this information was transformed into similarity information which was given to the Normalised Cuts algorithm [82] to form the final clusters. In this way, the performance of the UVOC was also improved significantly compared with the standard BoF approach.

The third research question was whether is it possible to take advantage of saliency information in the VOC process. In the last experiment of the thesis, the RCC background/foreground segmentation method [10] was used to find the most salient segment from the input images. Only the local features that were extracted from the salient segment were used. The results were compared with the results using all the local features and local features from the ground truth foreground. The result was evident: salient segment prediction improves the classification performance, but it is still inferior to the performance using the ground truth segmentations. However, we must notice that the Randomised Caltech-101 image set is quite difficult since some of the backgrounds also contain salient areas, and thus, it is difficult to detect the foreground. Nevertheless, the foreground prediction based on saliency improved the results. However, with the original Caltech-101 dataset salient segment detection did not improve the classification accuracy significantly. The reason behind this is that the backgrounds contain important information about the object in the foreground.

The saliency detector based on the Hessian-Laplace detected regions performed better than the standard baseline Itti&Koch detector [42] and even the detector by Cheng et al. [10] that has been claimed to be a state-of-the-art detector. However, the Hessian-Affine based saliency detector did not perform as well as the state-of-the-art detector by Judd et al. [46], but it is not as complicated and does not need any training as the detector by Judd et al. needs. In addition, the Hessian-Affine detector does not use central biased saliency as Judd et al. does. Without central biased maps, the Judd et al. detector's performance was almost equal to that of the central biased maps, which is close to the performance of Hessian-Affine based detector.

6.1 Future work

In this thesis, UVOC using the BoF approach [14] was studied. The standard BoF approach was improved by using spatial matching to verify local feature matches. However, this study has also revealed new issues related to UVOC.

6.1.1 Spatial information in unsupervised visual object categorisation

In this work, the spatial information was used for pairwise local feature matching between the images. The result of the spatial local feature matching was a cumulative sum of distances of the best matching local feature descriptors. The result of the spatial matching phase is pairwise distances between the images. In the supervised VOC with a k-NN classifier, with this information is straightforward, but in the unsupervised VOC it is not clear how the pairwise distance information should be used. In this work, pairwise distances were converted into similarity information and Normalised Cuts [82] was used to form the final clusters, but one could also use, e.g., PageRank [75] or any other graph algorithm to find the clusters.

6.1.2 Combining saliency detection with spatial information

In the saliency detection experiment, the performance of VOC was improved by detecting salient regions from the images and then using only local features that were detected from the salient regions. One could combine this method with the spatial matching by first detecting the salient regions from both images and then filtering out all the local features that are extracted from the non-salient regions. This would decrease the computation time and perhaps improve the categorisation performance.

6.1.3 Model selection problem

In the unsupervised learning, one of the difficult problems is that of selecting the number of categories correctly. In the UVOC development using a standard benchmark the number of categories can be fixed using the ground truth information, but in real life this is not possible. Thus, this problem needs attention.

Chapter VII

Conclusion

The number of digital images is huge and rapidly increasing both in the Internet and in the personally owned devices. The enormous number of images makes a manual image search for a particular type of image laborious and slow. Thus, there are many image sharing services and image managing applications that provide an image search. However, most of the image searches contain a major problem: the images must be described manually beforehand. Therefore, the main research question was whether it is possible to learn visual object categories in an unsupervised manner? Thus, this thesis studied an approach which tries to automatically find groups of images containing an object from the same category; a process which is called unsupervised visual object categorisation.

In this work, a Bag-of-Features based framework was studied for the problem of unsupervised visual object categorisation because the Bag-of-Features approach has performed well in supervised visual object categorisation and Bag-of-Features can be scaled up to thousands of categories. However, the performance was much lower than for the supervised case, but the introduced unsupervised visual object categorisation method can provide an "automatic organisation of images" which is visually agreeable.

The performance of the basic unsupervised visual object categorisation using the Bag-of-Features approach suffers from false local feature matches in the feature generation step, and thus, codebook histograms can be confused between the images of different categories. This problem leads to the second research question which was whether spatial information can be used in unsupervised visual object categorisation using the Bag-of-Features approach to improve the categorisation performance? The problem of false matching local features with the codebook can be narrowed down by using spatial information on the local features. In the spatial matching, also the spatial configuration of matching local features is verified. The spatial matching improved categorisation accuracy significantly, but it also increased computation dramatically. However, by choosing candidate images wisely using the Bag-of-Features method, the computational need can be kept reasonable.

The third research question was that how the saliency information can be used to improve unsupervised visual object categorisation performance? In this thesis, the saliency information was used to detect the salient region from the images and then to use only the local features that were extracted from the salient region. In the experiments, it was shown that salient region detection can significantly improve categorisation performance if the backgrounds do not contain important information about the foreground.

In the future, the model selection problem should be solved in order for unsupervised visual object categorisation methods to be made completely unsupervised. Nowadays, most of the unsupervised visual object categorisation methods (including the proposed method) need to be given the number of categories. This is not a severe problem if one is using a public benchmark dataset with known data. However, in the real life, the model selection problem can be very severe.

One can also try to improve the performance of the proposed unsupervised visual object categorisation method by combining foreground segmentation using visual saliency information and spatial local feature verification. The foreground segmentation filters out local features detected from the background, which decreases computation, and could also improve the categorisation performance with spatial matching.

Bibliography

[1] Alhoniemi, E., Himberg, J., Parhankangas, J., and Vesanto, J. SOM Toolbox. http://www.cis.hut.fi/somtoolbox/, 2000.

[2] Bar-Hillel, A., and Weinshall, D. Efficient learning of relational object class models. *International Journal of Computer Vision 77* (2008), 175–198.

[3] Bart, E., Porteous, I., Perona, P., and Welling, M. Unsupervised learning of visual taxonomies. In *Proc. of Computer Vision and Pattern Recognition* (2008).

[4] Bay, H., Tuytelaars, T., and Gool, L. Surf: Speeded up robust features. In *Proc. of European Conference on Computer Vision* (2006), pp. 404–417.

[5] Biederman, I. Recognition-by-components: A theory of human image understanding. *Psychological Review 94(2)* (1987), 115–147.

[6] Blei, D. M., Ng, A. Y., and Jordan, M. I. Latent Dirichlet Allocation. *Journal of Machine Learning Research 3*, 4-5 (2003), 993–1022.

[7] Borg, I., and Groenen, P. *Modern multidimensional scaling*, 2 ed. New York: Springer, 2005.

[8] Canny, J. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence 8* (1986), 679–698.

[9] Chang, K.-y., Liu, T.-l., and Lai, S.-h. From Co-saliency to Co-segmentation: An Efficient and Fully Unsupervised Energy Minimization Model. In *Proc. of Computer Vision and Pattern Recognition* (2011).

[10] Cheng, M., Zhang, G., Mitra, N., Huang, X., and Hu, S. Global Contrast based Salient Region Detection. In *Proc. of Computer Vision and Pattern Recognition* (2011), pp. 409–416.

[11] Chum, O., and Matas, J. Unsupervised discovery of co-occurrence in sparse high dimensional data. In *Proc. of Computer Vision and Pattern Recognition* (2010), pp. 3416–3423.

[12] Chum, O., Mikulik, A., Perdoch, M., and Matas, J. Total recall ii: Query expansion revisited. In *Proc. of Computer Vision and Pattern Recognition* (2011), pp. 889 –896.

[13] Chum, O., Philbin, J., Sivic, J., Isard, M., and Zisserman, A. Total recall: Automatic query expansion with a generative feature model for object retrieval. In *Proc. of International Conference on Computer Vision* (2007), pp. 1 –8.

[14] Csurka, G., Dance, C., Willamowski, J., Fan, L., and Bray, C. Visual categorization with bags of keypoints. In *ECCV Workshop on Statistical Learning in Computer Vision* (2004).

[15] Dalal, N., and Triggs, B. Histograms of Oriented Gradients for Human Detection. In *Proc. of Computer Vision and Pattern Recognition* (San Diego, United States, 2005), C. Schmid, S. Soatto, and C. Tomasi, Eds., vol. 1, IEEE Computer Society, pp. 886–893.

[16] Datta, R., Joshi, D., Li, J., and Wang, J. Z. Image retrieval: Ideas, influences, and trends of the new age. *ACM Computing Surveys 40* (2008), 5:1–5:60.

[17] Deng, J., Berg, A., Li, K., and Fei-Fei, L. What does classifying more than 10,000 image categories tell us? In *Proc. of European Conference on Computer Vision* (2010), Springer, pp. 71–84.

[18] Donoser, M., Urschler, M., Hirzer, M., and Bischof, H. Saliency driven total variation segmentation. In *Proc. of International Conference on Computer Vision* (2009), Ieee, pp. 817–824.

[19] Duan, L., Wu, C., Miao, J., Qing, L., and Fu, Y. Visual Saliency Detection by Spatially Weighted Dissimilarity. In *Proc. of Computer Vision and Pattern Recognition* (2011), pp. 474–480.

[20] Duda, R. O., and Hart, P. E. Use of the Hough transformation to detect lines and curves in pictures. *Communications of the ACM 15* (1972), 11–15.

[21] Dueck, D., and Frey, B. Non-metric affinity propagation for unsupervised image categorization. In *Proc. of International Conference on Computer Vision* (2007), pp. 1 –8.

[22] Everingham, M., Gool, V., L., Williams, C., and Zisserman, A. Pascal visual object classes challenge results. http://pascallin.ecs.soton.ac.uk/challenges/VOC/voc2005/results.pdf.

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

[24] Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., and Zisserman, A. The PASCAL Visual Object Classes Challenge 2010 (VOC2010) Results. http://www.pascal-network.org/challenges/VOC/voc2010/workshop/index.html.

[25] Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., and Zisserman, A. The PASCAL Visual Object Classes Challenge 2011 (VOC2011) Results. http://www.pascal-network.org/challenges/VOC/voc2011/workshop/index.html.

[26] Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., and Zisserman, A. The PASCAL Visual Object Classes Challenge 2008 (VOC2008) Results. http://www.pascal-network.org/challenges/VOC/voc2008/workshop/index.html, 2008.

[27] Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., and Zisserman, A. The PASCAL Visual Object Classes Challenge 2009 (VOC2009) Results. http://www.pascal-network.org/challenges/VOC/voc2009/workshop/, 2009.

[28] Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., and Zisserman, A. The PASCAL Visual Object Classes (VOC) challenge. *International Journal of Computer Vision 88*, 2 (2010), 303–338.

[29] Everingham, M., Zisserman, A., Williams, C. K. I., and Van Gool, L. The PASCAL Visual Object Classes Challenge 2006 (VOC2006) Results.

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

[31] Fei-Fei, L., Fergus, R., and Perona, P. One-shot learning of object categories. *IEEE Transactions on Pattern Analysis and Machine Intelligence 28*, 4 (2006), 594.

[32] Felzenszwalb, P., McAllester, D., and Ramanan, D. A discriminatively trained, multiscale, deformable part model. In *Proc. of Computer Vision and Pattern Recognition* (2008).

[33] Felzenszwalb, P. F., and Huttenlocher, D. P. Efficient Graph-Based Image Segmentation. *International Journal of Computer Vision 59*, 2 (2004), 167–181.

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

[35] Frank, A., and Asuncion, A. UCI machine learning repository. http://archive.ics.uci.edu/ml, 2010.

[36] Fulkerson, B., Vedaldi, A., and S.Soatto. Localizing objects with smart dictionaries. In *Proc. of European Conference on Computer Vision* (2008).

[37] Goferman, S., Zelnik-Manor, L., and Tal, A. Context-aware saliency detection. *Proc. of Computer Vision and Pattern Recognition* (2010), 1–8.

[38] Grauman, K., and Darrell, T. Unsupervised learning of categories from sets of partially matching image features. *Proc. of Computer Vision and Pattern Recognition 1* (2006), 19–25.

[39] Griffin, G., Holub, A., and Perona, P. Caltech-256 object category dataset. Tech. Rep. 7694, California Institute of Technology, 2007.

[40] Harris, C., and Stephens, M. A combined corner and edge detector. In *Proceedings of the 4th Alvey Vision Conference* (1988), pp. 147–151.

[41] Hu, Y., Li, M., and Yu, N. Multiple-instance ranking: Learning to rank images for image retrieval. In *Proc. of Computer Vision and Pattern Recognition* (2008), pp. 1 –8.

[42] Itti, L., and Koch, C. Computational modelling of visual attention. *Nature Reviews Neuroscience 2*, 3 (2001), 194–203.

[43] Jain, A. K., Murty, M. N., and Flynn, P. J. Data clustering: a review. *ACM Computing Surveys 31* (1999), 264–323.

[44] Jollife, I. T. *Principal Component Analysis.* Springer Series in Statistics, 2002.

[45] Jones, K. S. A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation 28*, 1 (1972), 11–21.

[46] Judd, T., Ehinger, K., Durand, F., and Torralba, A. Learning to predict where humans look. In *Proc. of International Conference on Computer Vision* (2009).

[47] Jurie, F., and Triggs, B. Creating efficient codebooks for visual recognition. In *Proc. of International Conference on Computer Vision* (2005), pp. 604–610.

[48] Kim, G., Faloutsos, C., and Hebert, M. Unsupervised Modeling of Object Categories Using Link Analysis Techniques. In *Proc. of Computer Vision and Pattern Recognition* (2008).

[49] Kim, G., and Torralba, A. Unsupervised Detection of Regions of Interest using Iterative Link Analysis. In *Proc. of Neural Information Processing Systems* (2009).

[50] Kinnunen, T., Kamarainen, J.-K., Lensu, L., and Kälviäinen, H. Bag-of-features codebook generation by self-organization. In *International Workshop on Self-Organizing Maps (WSOM)* (2009).

[51] Kinnunen, T., Kamarainen, J.-K., Lensu, L., and Kälviäinen, H. Unsupervised visual object categorization via self-organization. In *Proc. of International Conference on Pattern Recognition* (2010).

[52] Kinnunen, T., Kamarainen, J.-K., Lensu, L., Lankinen, J., and Kälviäinen, H. Making visual object categorization more challenging: Randomized caltech 101 data set. In *Proc. of International Conference on Pattern Recognition* (2010).

[53] Kohonen, T. The self-organizing map. *Proceedings of the IEEE 78*, 9 (1990), 1464–1480.

[54] Kremerskothen, K. 6,000,000,000. http://blog.flickr.net/en/2011/08/04/6000000000/, 2011.

[55] Labusch, K., Barth, E., and Martinetz, T. Sparse coding neural gas: Learning of overcomplete data representations. *Neurocomputing 72*, 7-9 (2009), 1547 – 1555.

[56] Lankinen, J., and Kamarainen, J.-K. Local feature based unsupervised alignment of object class images. In *Proc. of British Machine Vision Conference* (2011).

[57] Lazebnik, S., Schmid, C., and Ponce, J. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Proc. of Computer Vision and Pattern Recognition* (2006), pp. 2169–2178.

[58] Leibe, B., Ettlin, A., and Schiele, B. Learning semantic object parts for object categorization. *Image and Vision Computing 26* (2008), 15–26.

[59] Lewis, D. Naive (bayes) at forty: The independence assumption in information retrieval. In *Machine Learning: ECML-98* (1998), C. Nédellec and E. Rouveirol, Eds., vol. 1398 of *Lecture Notes in Computer Science*, Springer Berlin / Heidelberg, pp. 4–15.

[60] Lowe, D. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision 20* (2004), 91–110.

[61] Marchesotti, L., Cifarelli, C., and Csurka, G. A framework for visual saliency detection with applications to image thumbnailing. In *Proc. of International Conference on Computer Vision* (2009), pp. 2232–2239.

[62] Martinetz, T., and Schulten, K. A "Neural-Gas" Network Learns Topologies. *Artificial Neural Networks I* (1991), 397–402.

[63] Matas, J., Chum, O., Urban, M., and Pajdla, T. Robust wide-baseline stereo from maximally stable extremal regions. In *Proc. of British Machine Vision Conference* (2002), pp. 384–393.

[64] Mikolajczyk, K., Leibe, B., and Schiele, B. Local features for object class recognition. In *Proc. of Computer Vision and Pattern Recognition* (2005).

[65] Mikolajczyk, K., and Schmid, C. Indexing based on scale invariant interest points. In *Proc. of International Conference on Computer Vision* (2001), pp. 525–531.

[66] Mikolajczyk, K., and Schmid, C. An affine invariant interest point detector. In *Proc. of European Conference on Computer Vision* (2002), pp. 128–142.

[67] Mikolajczyk, K., and Schmid, C. Scale & affine invariant interest point detectors. *International Journal of Computer Vision 60* (2004), 63–86.

[68] Mikolajczyk, K., and Schmid, C. A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence 27*, 10 (2005), 1615–1630.

[69] Mikolajczyk, K., Tuytelaars, T., Matas, J., Schmid, C., and Zisserman, A. Featurespace. http://www.featurespace.org/, referenced 2011.

[70] Mikolajczyk, K., Tuytelaars, T., Schmid, C., Zisserman, A., Matas, J., Schaffalitzky, F., Kadir, T., and Gool, L. V. A comparison of affine region detectors. *International Journal of Computer Vision 65*, 1/2 (2005), 43–72.

[71] Møller, M. F. A scaled conjugate gradient algorithm for fast supervised learning. *Neural Networks 6*, 4 (1993), 525–533.

[72] Murtagh, F. A survey of recent advances in hierarchical clustering algorithms. *The Computer Journal 26*, 4 (1983), 354–359.

[73] Nister, D., and Stewenius, H. Scalable recognition with a vocabulary tree. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on* (2006), vol. 2, pp. 2161 – 2168.

[74] Ojala, T., Pietikainen, M., and Harwood, D. Performance evaluation of texture measures with classification based on kullback discrimination of distributions. In *Proc. of International Conference on Pattern Recognition* (1994), vol. 1, pp. 582 –585 vol.1.

[75] Page, L., Brin, S., Motwani, R., and Winograd, T. The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab, November 1999. Previous number = SIDL-WP-1999-0120.

[76] Philbin, J., Chum, O., Isard, M., Sivic, J., and Zisserman, A. Object retrieval with large vocabularies and fast spatial matching. In *Proc. of Computer Vision and Pattern Recognition* (2007), pp. 1–8.

[77] Ponce, J., Berg, T., Everingham, M., Forsyth, D., Hebert, M., Lazebnik, S., Marszalek, M., Schmid, C., Russell, B., Torralba, A., Williams, C., Zhang, J., and Zisserman, A. Dataset issues in object recognition. In *Workshop on Category Level Object Recognition* (2006), pp. 29–48.

[78] Rother, C., Kolmogorov, V., and Blake, A. "grabcut": interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics 23* (2004), 309–314.

[79] Roweis, S., and Saul, L. Nonlinear Dimensionality Reduction by Locally Linear Embedding. *Science 290*, 5500 (2000), 2323–2326.

[80] Russell, B., Freeman, W., Efros, A., Sivic, J., and Zisserman, A. Using multiple segmentations to discover objects and their extent in image collections. In *Proc. of Computer Vision and Pattern Recognition* (2006), vol. 2, pp. 1605 – 1614.

[81] Russell, B., Torralba, A., Murphy, K., and Freeman, W. Labelme: A database and web-based tool for image annotation. *International Journal of Computer Vision 77*, 1-3 (2008), 157–173.

[82] Shi, J., and Malik, J. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence 22*, 8 (2000), 888 –905.

[83] Sivic, J., Russell, B. C., Zisserman, A., Freeman, W. T., and Efros, A. A. Unsupervised discovery of visual object class hierarchies. In *Proc. of Computer Vision and Pattern Recognition* (2008), pp. 1–8.

[84] Sivic, J., and Zisserman, A. Video google: a text retrieval approach to object matching in videos. In *Proc. of Computer Vision and Pattern Recognition* (2003), pp. 1470 –1477.

[85] Smeulders, A. W. M., Worring, M., Santini, S., Gupta, A., and Jain, R. Content-based image retrieval at the end of the early years. *IEEE Transactions on Pattern Analysis and Machine Intelligence 22* (2000), 1349–1380.

[86] Snavely, N., Seitz, S., and Szeliski, R. Photo tourism: exploring photo collections in 3D. In *ACM Transactions on Graphics* (2006), vol. 25, ACM, pp. 835–846.

[87] Song, Z., Chen, Q., Huang, Z., Hua, Y., and Yan, S. Contextualizing object detection and classification. In *CVPR* (2011), pp. 1585–1592.

[88] Su, H., Sun, M., Fei-Fei, L., and Savarese, S. Learning a dense multi-view representation for detection, viewpoint classification and synthesis of object categories. In *Proc. of International Conference on Computer Vision* (2009).

[89] Sugano, Y., Matsushita, Y., and Sato, Y. Calibration-free gaze sensing using saliency maps. In *Proc. of Computer Vision and Pattern Recognition* (2010), pp. 2667–2674.

[90] Tenenbaum, J., de Silva, V., and Langford, J. A Global Geometric Framework for Nonlinear Dimensionality Reduction. *Science 290*, 5500 (2000), 2319–2323.

[91] Tuytelaars, T. Dense interest points. In *Proc. of Computer Vision and Pattern Recognition* (2010).

[92] Tuytelaars, T., Lampert, C., Blaschko, M., and Buntine, W. Unsupervised object discovery: A comparison. *International Journal of Computer Vision 88*, 2 (2010).

[93] Tuytelaars, T., and Mikolajczyk, K. Local invariant feature detectors: a survey. *Foundations and Trends in Computer Graphics and Vision 3* (July 2008), 177–280.

[94] Valenti, R., Sebe, N., and Gevers, T. Image saliency by isocentric curvedness and color. In *Proc. of International Conference on Computer Vision* (2009), IEEE, pp. 2185–2192.

[95] van de Sande, K. E. A., Gevers, T., and Snoek, C. G. M. Evaluating color descriptors for object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence 32*, 9 (2010), 1582–1596.

[96] van Gemert, J., Geusebroek, J., Veenman, C., and Smeulders, A. Kernel codebooks for scene categorization. In *Proc. of European Conference on Computer Vision* (2008), pp. 696–709.

[97] Viola, P., and Jones, M. Robust real time object detection. *International Journal of Computer Vision* (2001).

[98] Wandell, B. *Foundations of Vision.* Sinauer Associates, Inc., Sunderland, Massachusetts, USA, 1995.

[99] Wang, G., and Forsyth, D. Joint learning of visual attributes, object classes and visual saliency. In *Proc. of International Conference on Computer Vision* (2009), pp. 537–544.

[100] Wang, M., Konrad, J., Ishwar, P., and Jing, K. Image Saliency: From Intrinsic to Extrinsic Context. In *Proc. of Computer Vision and Pattern Recognition* (2011), vol. 1, pp. 417–424.

[101] Wang, W., Wang, Y., Huang, Q., and Gao, W. Measuring visual saliency by site entropy rate. In *Proc. of Computer Vision and Pattern Recognition* (2010), no. 2, IEEE, pp. 2368–2375.

[102] Yang, J., Li, Y., Tian, Y., Duan, L., and Gao, W. Group-sensitive multiple kernel learning for object categorization. In *Proc. of International Conference on Computer Vision* (2009), pp. 436–443.

[103] Zhang, J., Marszalek, M., Lazebnik, S., and Schmid, C. Local features and kernels for classification of texture and object categories: A comprehensive study. *International Journal of Computer Vision 73*, 2 (2007).

[104] Zhao, W. LIP-VIREO local interest point extraction toolkit, referenced 2010. `http://vireo.cs.cityu.edu.hk`.

ACTA UNIVERSITATIS LAPPEENRANTAENSIS

415. ZAKHVALINSKII, VASILII. Magnetic and transport properties of $LaMnO_{3+\delta}$, $La_{1-x}Ca_xMnO_3$, $La_{1-x}Ca_xMn_{1-y}Fe_yO_3$ and $La_{1-x}Sr_xMn_{1-y}Fe_yO_{3.}$ 2010. Diss.

416. HATAKKA, HENRY. Effect of hydrodynamics on modelling, monitoring and control of crystallization. 2010. Diss.

417. SAMPO, JOUNI. On convergence of transforms based on parabolic scaling. 2010. Diss.

418. TURKU. IRINA. Adsorptive removal of harmful organic compounds from aqueous solutions. 2010. Diss.

419. TOURUNEN, ANTTI. A study of combustion phenomena in circulating fluidized beds by developing and applying experimental and modeling methods for laboratory-scale reactors. 2010. Diss.

420. CHIPOFYA, VICTOR. Training system for conceptual design and evaluation for wastewater treatment. 2010. Diss.

421. KORTELAINEN, SAMULI. Analysis of the sources of sustained competitive advantage: System dynamic approach. 2011. Diss.

422. KALJUNEN, LEENA. Johtamisopit kuntaorganisaatiossa – diskursiivinen tutkimus sosiaali- ja terveystoimesta 1980-luvulta 2000-luvulle. 2011. Diss.

423. PEKKARINEN, SATU. Innovations of ageing and societal transition. Dynamics of change of the socio-technical regime of ageing. 2011. Diss.

424. JUNTTILA, VIRPI. Automated, adapted methods for forest inventory. 2011. Diss.

425. VIRTA, MAARIT. Knowledge sharing between generations in an organization – Retention of the old or building the new 2011. Diss.

426. KUITTINEN, HANNA. Analysis on firm innovation boundaries. 2011. Diss.

427. AHONEN, TERO. Monitoring of centrifugal pump operation by a frequency converter. 2011. Diss.

428. MARKELOV, DENIS. Dynamical and structural properties of dendrimer macromolecules. 2011. Diss.

429. HÄMÄLÄINEN, SANNA. The effect of institutional settings on accounting conservatism – empirical evidence from the Nordic countries and the transitional economies of Europe. 2011. Diss.

430. ALAOUTINEN, SATU. Enabling constructive alignment in programming instruction. 2011. Diss.

431. ÅMAN, RAFAEL. Methods and models for accelerating dynamic simulation of fluid power circuits. 2011. Diss.

432. IMMONEN, MIKA. Public-private partnerships: managing organizational change for acquiring value creative capabilities. 2011. Diss.

433. EDELMANN, JAN. Experiences in using a structured method in finding and defining new innovations: the strategic options approach. 2011. Diss.

434. KAH, PAUL. Usability of laser - arc hybrid welding processes in industrial applications. 2011. Diss.

435. OLANDER, HEIDI. Formal and informal mechanisms for knowledge protection and sharing. 2011. Diss.

436. MINAV, TATIANA. Electric drive based control and electric energy regeneration in a hydraulic system. 2011. Diss.

437. REPO, EVELIINA. EDTA- and DTPA-functionalized silica gel and chitosan adsorbents for the removal of heavy metals from aqueous solutions. 2011. Diss.

438. PODMETINA, DARIA. Innovation and internationalization in Russian companies: challenges and opportunities of open innovation and cooperation. 2011. Diss.

439. SAVITSKAYA, IRINA. Environmental influences on the adoption of open innovation: analysis of structural, institutional and cultural impacts. 2011. Diss.

440. BALANDIN, SERGEY, KOUCHERYAVY, YEVGENI, JÄPPINEN, PEKKA, eds. Selected Papers from FRUCT 8 .2011.

441. LAHTI, MATTI. Atomic level phenomena on transition metal surfaces. 2011. Diss.

442. PAKARINEN, JOUNI. Recovery and refining of manganese as by-product from hydrometallurgical processes. 2011. Diss.

443. KASURINEN, JUSSI. Software test process development. 2011. Diss.

444. PEKKANEN, PETRA. Delay reduction in courts of justice – possibilities and challenges of process improvement in professional public organizations. 2011. Diss.

445. VANHALA, MIKA. Impersonal trust within the organization: what, how, and why? 2011. Diss.

446. HYNYNEN, KATJA. Broadband excitation in the system identification of active magnetic bearing rotor systems. 2011. Diss.

447. SOLONEN, ANTTI. Bayesian methods for estimation, optimization and experimental design. 2011. Diss.

448. JABLONSKA, MATYLDA. From fluid dynamics to human psychology. What drives financial markets towards extreme events. 2011. Diss.

449. MYÖHÄNEN, KARI. Modelling of combustion and sorbent reactions in three-dimensional flow environment of a circulating fluidized bed furnace. 2011. Diss.

450. LAATIKAINEN, MARKKU. Modeling of electrolyte sorption – from phase equilibria to dynamic separation systems. 2011. Diss.

451. MIELONEN, JUHA. Making Sense of Shared Leadership. A case study of leadership processes and practices without formal leadership structure in the team context. 2011. Diss.

452. PHAM, ANH TUAN. Sewage sludge electro-dewatering. 2011. Diss.

453. HENNALA, LEA. Kuulla vai kuunnella – käyttäjää osallistavan palveluinnovoinnin lähestymistavan haasteet julkisella sektorilla. 2011. Diss.

454. HEINIMÖ, JUSSI. Developing markets of energy biomass – local and global perspectives. 2011. Diss.

455. HUJALA, MAIJA. Structural dynamics in global pulp and paper industry. 2011. Diss.

456. KARVONEN, MATTI. Convergence in industry evolution. 2011. Diss.