



Universidad  
de Alcalá

*DPTO. DE TEORÍA DE LA SEÑAL Y  
COMUNICACIONES  
Campus Universitario  
Crta. Madrid-Barcelona, Km. 33.600  
28805 Alcalá de Henares  
Tlf: +34 91 885 66 90  
Fax: +34 91 885 66 99  
dpto.teoriadelasenal@uah.es*

D. SATURNINO MALDONADO BASCÓN, Catedrático de Escuela Universitaria del Área de Conocimiento de Teoría de la Señal y Comunicaciones de la Universidad de Alcalá

CERTIFICA:

Que la tesis “**Visual Vocabularies for Category-Level Object Recognition**”, presentada por D. Roberto Javier López Sastre, realizada en el Departamento de Teoría de la Señal y Comunicaciones bajo mi dirección, reúne méritos suficientes para optar al grado de Doctor, por lo que puede procederse a su depósito y lectura.

Alcalá de Henares, 8 de abril de 2010.

Fdo: Dr. D. Saturnino Maldonado Bascón





Universidad  
de Alcalá

*DPTO. DE TEORÍA DE LA SEÑAL Y  
COMUNICACIONES  
Campus Universitario  
Crta. Madrid-Barcelona, Km. 33.600  
28805 Alcalá de Henares  
Tlf: +34 91 885 66 90  
Fax: +34 91 885 66 99  
dpto.teoriadelasenal@uah.es*

D. Roberto Javier López Sastre ha realizado en el Departamento de Teoría de la Señal y Comunicaciones, bajo la dirección del Doctor D. Saturnino Maldonado Bascón, la tesis doctoral titulada “**Visual Vocabularies for Category-Level Object Recognition**”, cumpliéndose todos los requisitos para la tramitación que conduce a su posterior lectura.

Alcalá de Henares, 8 de abril de 2010.

EL DIRECTOR DEL DEPARTAMENTO

Fdo: Dr. D. Manuel Rosa Zurera.



# UNIVERSIDAD DE ALCALÁ

ESCUELA POLITÉCNICA SUPERIOR

DEPARTAMENTO DE TEORÍA DE LA SEÑAL Y  
COMUNICACIONES



## “Visual Vocabularies for Category-Level Object Recognition”

Ph.D. Thesis

Autor

Roberto Javier López Sastre

Director

Saturnino Maldonado Bascón

2010



*A Vanessa y a mi abuelo Francisco.*



# Agradecimientos

Han sido muchas las personas que me han acompañado y comprendido durante estos años de duro trabajo. A cada una de ellas le corresponde un trocito de esta tesis.

Este trabajo no hubiera sido posible sin el apoyo de mi director Satur. Quiero agradecerle que me diera la oportunidad de realizar esta tesis, que me dejara libertad a la hora de elegir el tema de investigación, que tenga siempre la puerta de su despacho abierta y su sincera amistad. Gracias también a todos los compañeros, becarios y estudiantes del grupo de investigación GRAM, por las revisiones de artículos, por las fiestas en el fondo de pasillo y por haber hecho esta aventura mas divertida.

My deepest gratitude travels now to Leuven, where there is another research group I feel I belong to: VISICS. First and foremost, I would like to thank Prof. Tinne Tuytelaars for giving me the opportunity to work during 6 months in such an exciting place. She has been next to me from the beginning, being a constant source of motivation. I am really thankful for her guidance, invaluable feedbacks, helpful criticism and availability (sometimes late nights) before paper deadlines. I am also grateful to my colleagues at the VISICS group for making my stay in Leuven so funny and enriching! Particular thanks go to David and Xinhai, the badminton team, for being great friends.

Tengo la suerte de trabajar en un departamento serio y activo. Quiero expresar mi mas sincero agradecimiento a todos mis compañeros y especialmente a Manuel Rosa y Francisco López por su confianza y por ofrecerme una oportunidad en la universidad.

También estoy tremendamente agradecido a Rafael Sendra por las reuniones que mantuvimos en *ciencias* y de las que siempre me marchaba con alguna nueva idea y un sabio consejo en la mochila.

Finalmente los mas importantes: mi familia. Ellos son los que han estado a mi lado todo este tiempo, apoyándome de forma desinteresada.

Gracias a los Virumbrales-Arilla por su cariño y por hacer que cada día que pasa me sienta mas unido a ellos.

Una mención especial va sin duda para mi tía Inma, excelente revisora de Inglés y mejor madrina.

Se que mi abuela está orgullosa por todo esto, pero soy yo el que debe estar orgulloso de tener una abuela como ella. Muchos besos.

A mis padres y hermana les debo todo lo que soy, y si he conseguido esto es gracias a ellos. La paciencia y dulzura de mi madre, el tesón y la autenticidad de mi padre, y la entrega y arte de mi hermana se encierran entre las tapas de este libro. Os quiero.

Y finalmente, Vanessa. Para ella es esta tesis, como no podía ser de otra manera. Siempre ahí, a mi lado, queriendo esperarme y esperándome mientras me quiere, pintando de color los malos ratos y dando sentido al esfuerzo. Ahora, por fin, el tiempo vuelve a ser nuestro, de los dos.

# Abstract

This thesis focuses on the study of visual vocabularies for category-level object recognition. Specifically, we state novel approaches for building visual codebooks. Our aim is not just to obtain more discriminative and more compact visual codebooks, but to bridge the gap between visual features and semantic concepts. A novel approach for obtaining class representative visual words is presented. It is based on a maximisation procedure, *i.e.* the Cluster Precision Maximisation (CPM), of a novel cluster precision criterion, and on an adaptive threshold refinement scheme for agglomerative clustering algorithms based on correlation clustering techniques. The objective is to increase the vocabulary compactness while at the same time improve the recognition rate and further increase the representativeness of the visual words.

Moreover, we describe a novel clustering aggregation based approach for building efficient and semantic visual vocabularies. It consist of a novel framework for incorporating neighboring appearances of local descriptors into the vocabulary construction, and a rigorous approach for adding meaningful spatial coherency among the local features into the visual codebooks.

We also propose an efficient high-dimensional data clustering algorithm, *i.e.* the Fast Reciprocal Nearest Neighbours (Fast-RNN). Our approach, which is a speeded up version of the standard RNN algorithm, is based on the projection search paradigm.

Finally, we release a new database of images called Image Collection of Annotated Real-world Objects (ICARO), which is especially designed for evaluating category-level object recognition systems. An exhaustive comparison of ICARO with other well-known datasets used within the same context is carried out. We also propose a benchmark for both object classification and detection.

**Index terms:** database, benchmark, object recognition, object detection, category-level, visual vocabulary, correlation clustering, cluster precision, Reciprocal Nearest Neighbours, bag of words, clustering aggregation.



# Resumen

Esta tesis se centra en el estudio de vocabularios visuales para el reconocimiento de categorías de objetos en imágenes. El objetivo que perseguimos no es solo que éstos vocabularios sean más compactos y discriminativos, sino que también permitan caracterizar la información semántica presente en las imágenes. Así, la tesis comienza describiendo una nueva propuesta que garantiza la obtención de vocabularios en los que las palabras visuales son representativas para cada una de las clases. La metodología diseñada se basa en la maximización de un nuevo criterio para medir la precisión de los *clusters*. Además, la tesis describe un algoritmo, basado en las técnicas conocidas como *correlation clustering*, que consigue reducir el tamaño del vocabulario, a la vez que lo hace más discriminativo.

La tesis también aborda la utilización de algoritmos de *clustering aggregation* para de nuevo conseguir vocabularios visuales que sean semánticos y que mejoren la eficiencia de los sistemas de categorización de objetos. La nueva propuesta incorpora en el proceso de construcción del vocabulario tanto información local como de apariencia de los descriptores que han sido extraídos de las imágenes de entrenamiento.

El problema de la cuantificación eficiente de vectores en espacios de altas dimensiones, para por ejemplo la obtención de palabras visuales, es otra de las líneas de trabajo de esta tesis. Se presenta una versión acelerada del algoritmo de clustering aglomerativo conocido como clustering de vecinos recíprocos más cercanos (RNN). El algoritmo propuesto utiliza el paradigma de la búsqueda por proyección para acelerar la construcción de las cadenas de vecinos más cercanos que se utilizan de forma intensiva en el algoritmo RNN.

Finalmente, destacar que la tesis también se enfrenta al problema del diseño y construcción de una base de datos de imágenes para la evaluación y comparación de algoritmos de reconocimiento y detección de categorías de objetos. La nueva base de datos se denomina *Image Collection of Annotated Real-world Objects* (ICARO).

**Palabras clave:** base de datos, reconocimiento de clases de objetos, detección de objetos, vocabulario visual, precisión de clusters, vecinos cercanos recíprocos, bolsa de palabras, clustering, palabras visuales.

# Glossary

<b>BoF</b>	Bag-of-Features
<b>BoW</b>	Bag-of-Words
<b>CP</b>	Cluster Precision
<b>CPM</b>	Cluster Precision Maximisation
<b>ERCF</b>	Extremely Randomized Clustering Forests
<b>GMM</b>	Gaussian Mixture Model
<b>HIK</b>	Histogram Intersection Kernel
<b>HOG</b>	Histogram of Oriented Gradients
<b>HSL</b>	Hue Saturation Lightness
<b>HSV</b>	Hue Saturation Value
<b>ICARO</b>	Image Collection of Annotated Real-world Objects
<b>ISM</b>	Implicit Shape Model
<b>JPEG</b>	Joint Photographic Experts Group
<b>LDA</b>	Latent Dirichlet Allocation
<b>MI</b>	Mutual Information
<b>MS</b>	Main Set
<b>NN</b>	Nearest Neighbour
<b>PASCAL VOC</b>	PASCAL Visual Object Classes

<b>PLSA</b>	Probabilistic Latent Semantic Analysis
<b>PSM</b>	Partial Surface Model
<b>RBF</b>	Radial Basis Function
<b>RGB</b>	Red Green Blue
<b>RNN</b>	Reciprocal Nearest Neighbours
<b>SCA</b>	Sorted Coordinate Array
<b>SIFT</b>	Scale-Invariant Feature Transform
<b>SS</b>	Secondary Set
<b>SURF</b>	Speeded Up Robust Features
<b>SVM</b>	Support Vector Machine
<b>UPGMA</b>	Unweighted Pair Group Method with Arithmetic mean
<b>XML</b>	Extensible Markup Language
$\chi^2$ - <b>Kernel</b>	Extended Gaussian Kernel with $\chi^2$ distance

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Recognising Object Categories . . . . .	1
1.1.1	The Challenges . . . . .	3
1.2	Motivation . . . . .	6
1.3	Outline of the thesis . . . . .	7
<b>2</b>	<b>State of the art</b>	<b>9</b>
2.1	Category-level Object Recognition . . . . .	9
2.1.1	Single Object Classes . . . . .	10
2.1.1.1	Bag-of-Parts Models . . . . .	10
2.1.1.2	Part-based models . . . . .	12
2.1.1.3	Discriminative approaches . . . . .	13
2.1.1.4	3D Object Categorisation . . . . .	14
2.1.2	Multiple Object Classes . . . . .	15
2.2	Learning Semantic Visual Words . . . . .	16
2.2.1	Supervised approaches . . . . .	16
2.2.2	Unsupervised approaches . . . . .	17
<b>3</b>	<b>Datasets</b>	<b>19</b>
3.1	Introduction . . . . .	20
3.2	Other datasets . . . . .	21
3.2.1	Large Scale Datasets . . . . .	22
3.2.2	Small Scale Datasets . . . . .	22
3.2.3	Summary of the Databases . . . . .	23
3.3	ICARO . . . . .	23
3.3.1	ICARO in Numbers . . . . .	23
3.4	Comparative Analysis of ICARO . . . . .	30
3.4.1	Objects Distribution and Scale . . . . .	30
3.4.2	Intra-class Variability . . . . .	33
3.5	Results . . . . .	35
3.5.1	Image Classification . . . . .	35

3.5.1.1	Benchmark . . . . .	35
3.5.1.2	Results . . . . .	39
3.5.2	Object Detection . . . . .	44
3.5.2.1	Benchmark . . . . .	44
3.5.2.2	Results . . . . .	44
3.6	Conclusion . . . . .	45
<b>4</b>	<b>Class Representative Visual Words</b>	<b>49</b>
4.1	Introduction . . . . .	50
4.2	Obtaining Class Representative Visual Words . . . . .	51
4.2.1	Agglomerative Clustering Algorithm . . . . .	52
4.2.1.1	RNN Clustering Algorithm: an overview . . . . .	53
4.2.2	The Cluster Precision Maximisation . . . . .	53
4.3	Correlation Clustering Based Refinement Step . . . . .	58
4.3.1	Correlation Clustering . . . . .	60
4.3.2	Refining Visual Words . . . . .	61
4.3.2.1	Adaptive Threshold Merging Algorithm . . . . .	62
4.4	Experiments . . . . .	63
4.4.1	Experimental Setup . . . . .	64
4.4.2	CPM performance with RNN clustering . . . . .	66
4.4.3	Increasing the recognition rate with the correlation clustering based refinement . . . . .	67
4.4.4	A comparison with $K$ -means codebooks . . . . .	71
4.4.4.1	CPM performance . . . . .	71
4.4.4.2	Classification performance . . . . .	71
4.4.4.3	Computational Complexity . . . . .	73
4.5	Conclusions and future work . . . . .	75
<b>5</b>	<b>Fast Reciprocal Nearest Neighbours Clustering</b>	<b>77</b>
5.1	Introduction . . . . .	77
5.2	Reciprocal Nearest Neighbours Clustering . . . . .	79
5.3	Fast-RNN . . . . .	82
5.3.1	Building NN chains via <i>slicing</i> . . . . .	84
5.3.2	Determining $\epsilon$ . . . . .	86
5.3.3	Data Structure . . . . .	89
5.3.4	Discussion . . . . .	92
5.4	Experimental Evaluation . . . . .	93
5.4.1	Experiments with local descriptors . . . . .	94
5.4.2	Normally Distributed Random Datasets . . . . .	96
5.4.3	Determining the best $\epsilon$ . . . . .	96
5.5	Conclusion . . . . .	98

---

<b>6</b>	<b>Aggregating Visual Words</b>	<b>99</b>
6.1	Introduction . . . . .	99
6.2	Clustering Aggregation . . . . .	102
6.3	Visual Words Aggregation . . . . .	105
6.3.1	Simple Aggregation . . . . .	106
6.3.2	Incorporating Semantic Information . . . . .	106
6.3.2.1	Via Regular Grids . . . . .	107
6.3.2.2	Via over-segmentation . . . . .	107
6.4	Results . . . . .	109
6.4.1	Experimental Setup . . . . .	109
6.4.2	Evaluation criteria . . . . .	110
6.4.3	Codebooks Performance in Image Classification . . . . .	111
6.4.3.1	PASCAL VOC Challenge 2007 dataset . . . . .	111
6.4.3.2	PASCAL VOC Challenge 2009 dataset . . . . .	114
6.5	Conclusion . . . . .	116
<b>7</b>	<b>Conclusions</b>	<b>123</b>
7.1	Summary and main contributions . . . . .	123
7.2	Future directions . . . . .	125
<b>A</b>	<b>Evolution of cluster precision</b>	<b>127</b>



# List of Figures

1.1	Recognising objects in a real-world scene. . . . .	2
1.2	Object categorisation example. . . . .	2
1.3	Intra-class variability example. . . . .	4
1.4	Inter-class variability problem. . . . .	4
1.5	Changes in the illumination conditions. . . . .	5
1.6	Scale invariance, occlusions and changes in viewpoint. . . . .	5
1.7	Submitted papers by primary subject area to the CVPR 2009. . . . .	6
2.1	BoW system overview. . . . .	11
2.2	The parts and structure model. . . . .	12
3.1	ICARO classes organised in a taxonomy of 7 main branches. . . . .	25
3.2	Sample images of the MS in ICARO. . . . .	27
3.3	Summary of the ICARO dataset content. . . . .	31
3.4	Examples of distributions of object locations in different datasets. . . . .	32
3.5	Variation of the bounding boxes sizes as a function of the number of objects in the database. . . . .	33
3.6	Comparison of 7 datasets used for object detection and recognition. . . . .	34
3.7	Comparison of the entropy of averages for three different objects categories. . . . .	36
3.8	Sample average images for different object categories. . . . .	37
3.9	Examples of images for each class in ICARO. . . . .	38
3.10	Performance of the different methods as a function of $N_{\text{train}}$ . The algorithm definition corresponds to $K$ - $L$ -Kernel. . . . .	41
3.11	ICARO classification results (I). . . . .	42
3.12	Ranked images for classes apple, bike, camera, car, foot, glasses, mug and traffic light. . . . .	43
3.13	ICARO detection results. . . . .	46
4.1	Example of class representative visual words. . . . .	52

4.2	Cluster precision problem. . . . .	56
4.3	Cluster precision calculation. . . . .	58
4.4	$P(t)$ evolution in a CPM execution. . . . .	59
4.5	Example of correlation clustering. . . . .	61
4.6	Correlation clustering based refinement. . . . .	62
4.7	Images examples from the Caltech-101 subset. . . . .	65
4.8	CPM performance. . . . .	68
4.9	Correlation Clustering results on the Caltech-101 subset. . . .	70
4.10	Comparison of the CP obtained with RNN and $K$ -means codebooks. . . . .	72
4.11	Average classification rate for different clustering algorithms ( $K$ -means, RNN, refined RNN), different vocabulary sizes and different kernels. . . . .	73
5.1	NN chain example. . . . .	82
5.2	Slicing a 2-dimensional space. . . . .	83
5.3	Incorrect NN search result. . . . .	86
5.4	Double slicing. . . . .	87
5.5	Normal probability plot. . . . .	88
5.6	$\epsilon$ vs. Probability of success. . . . .	90
5.7	Data structures of Fast-RNN. . . . .	91
5.8	Speedup for SIFT and PCA-SIFT descriptors. . . . .	95
5.9	Speedup for SURF descriptors. . . . .	97
5.10	Epsilon vs. Speedup for PCA-SIFT, SIFT and SURF descriptors. . . . .	98
6.1	Example of clustering aggregation. . . . .	103
6.2	Clustering aggregation toy example. . . . .	104
6.3	Flowchart of our approach for constructing a visual vocabulary using clustering aggregation. . . . .	106
6.4	Using regular grids to add spatial information to the vocabulary construction process. . . . .	108
6.5	Image over-segmented. . . . .	108
6.6	Using over-segmentation for adding spatial/semantic information to the vocabulary construction process. . . . .	109
6.7	Evaluation of codebooks on image categorisation over the PASCAL VOC 2007 Challenge. . . . .	112
6.8	Ranked images for the classes aeroplane, bicycle, boat, cat, horse and person. . . . .	114
6.9	PASCAL VOC Challenge 2009: Median Average Precision. . . .	116
6.10	PASCAL VOC Challenge 2009: average precision per class. . . .	117

6.11 PASCAL VOC Challenge 2009 vs. PASCAL VOC Challenge 2008. . . . .	118
6.12 Average Precision vs. Object Class Area (I). . . . .	119
6.13 Average Precision vs. Object Class Area (II). . . . .	120



# List of Tables

3.1	Comparison of the main datasets of visual objects. . . . .	24
3.2	ICARO 2009 in a nutshell. . . . .	26
3.3	MS statistics of ICARO. . . . .	28
3.4	SS statistics of ICARO. . . . .	28
3.5	Statistics of the annotated viewpoints in the SS of ICARO. . .	29
3.6	Detection results of the detector in the PASCAL VOC Challenge 2006-2007-2008 and ICARO datasets. . . . .	45
3.7	Detection results of the detector trained on the ICARO database. . . . .	45
4.1	Outline of the number of training and test images and features obtained from the database for the experiments performed in this work. . . . .	64
4.2	Confusion matrix before the adaptive clustering refinement process (a) and after (b). . . . .	69
5.1	Datasets for the experiments. . . . .	94
5.2	Fast-RNN vs. RNN with the 100K-PCA-SIFT and 100K-SIFT sets. . . . .	94
5.3	Fast-RNN vs. RNN with the different SURF datasets. . . . .	96
5.4	Fast-RNN vs. RNN with the normally distributed datasets. . . . .	97
6.1	Experiments descriptions for simple clustering aggregation. . .	112
6.2	<i>MAP</i> obtained for the experiments CA- <i>i</i> . . . . .	113
6.3	Clustering combinations for the experiments SCA- <i>i</i> . . . . .	113
6.4	<i>MAP</i> obtained for for the experiments SCA- <i>i</i> . . . . .	113
6.5	PASCAL VOC 2009 Results . . . . .	115
6.6	PASCAL VOC 2008 Results. . . . .	116



# Chapter 1

## Introduction

*As you set out for Ithaca  
hope the voyage is a long one,  
full of adventure, full of discovery.*

Constantine P. Cavafy, *Ithaca*.

---

### 1.1 Recognising Object Categories

We humans look at a picture and are able not just to see a pattern of colour and texture, but to comprehend it. Whatever the image depicts, we have the ability to interpret it. Furthermore, we do this with an astonishing ease.

Figure 1.1 shows a picture of a square in downtown Madrid. We immediately recognise hundreds of different *object categories*, such as cars, buses, people or buildings. But we can continue categorising more and more objects: windows, wheels, shoes, heads, etc. How many categories are there?

Biederman (1987) states that humans easily distinguish around 30.000 categories. However, we can categorise much more subtle distinctions within object categories. For instance, we can recognise our individual sister or car. Moreover, we are able to estimate the locations and poses of all the elements in the scene. In Figure 1.1 we can state whether the building is behind the car.

Vision is complicated indeed, and one of the formidable problems is the category-level object recognition, which is the objective of this thesis. Given an image, our aim is to predict the presence/absence of at least one object of an individual class. That is, to answer the question: does the image contain any instances of a particular object class? The output of an object cate-



Figure 1.1: Recognising objects in a real-world scene. This figure is best viewed in colour.



Figure 1.2: Object categorisation example.

gorisation system might be a real-valued confidence of the object's presence. Note that the classification problem is different from the detection problem, whose objective is to determine where the objects are within the images. Figure 1.2 shows an example of the output of an object categorisation system trained for the category bicycle.

Among the problems that an artificial categorisation system has to face, there are several evident ones: *representation*, *classification* and *learning*.

**Representation** How are we going to represent the images? Furthermore, how are the visual classes going to be represented? What local features are more suitable to address the categorisation problem?

**Classification** First, a category model has to be learnt from the features extracted from the images. Then, the classification problem can be formulated as follows: given a number of learnt classes, a new image has to be

processed and a decision has to be drawn, whether a known category appears in the data or not. What pattern recognition approach do we use? Does a probabilistic approach suit the category-level object recognition problem best? Do we implement a discriminative or a generative approach? Can kernel methods, such as Support Vector Machine (SVM), deal with the visual categorisation problem?

**Learning** A category-level object recognition system has to learn from a number of training images. Which learning paradigm do we choose – supervised, semi-supervised or unsupervised? Moreover, incremental learning techniques are very interesting if we want to be able to learn when a certain new object class is presented to the system.

In this dissertation we focus on the representation problem. Specifically, we focus on those systems known as Bag-of-Features (BoF), in which the images are represented as a set of local features extracted from them. These feature vectors, *e.g.* Scale-Invariant Feature Transform (SIFT) (Lowe 1999), are used to learn a discriminative approach from training images. Most of these BoF systems require a vector quantisation step to cluster the local descriptors in the feature (high-dimensional) space.  $K$ -means and hierarchical clustering algorithms are widely used. These approaches are often termed as Bag-of-Words (BoW) because the cluster centres obtained by such a vector quantisation step are known as visual words (Sivic and Zisserman 2003). Furthermore, the local descriptors quantisation is known as the visual vocabulary construction stage. The ultimate goal of this thesis is to work on this visual vocabulary construction process. Our aim is to obtain more compact and more discriminative visual vocabularies which also capture semantic information.

### 1.1.1 The Challenges

The category-level object recognition problem is not solved yet. It is considered as one of the most challenging and ambitious problems on computer vision. So far we have shown, with the square picture of Figure 1.1, what we humans do when we look at a picture: we recognise hundreds of categories. However, what appears to us as natural becomes tremendously difficult for a computer vision system.

**Intra-class variability** First, we need to handle the intra-class variability challenge. That is, we need systems which are able to deal with the large



Figure 1.3: Intra-class variability example.



(a) touring bike



(b) mountain bike

Figure 1.4: Inter-class variability problem.

degree of visual variability an individual category may have. For instance, Figure 1.3 shows some examples of images where the telephone cabin class appears. It is straightforward to perceive that these three telephone cabins are very different in terms of visual appearance, but all must be classified within the same class.

**Inter-class variability** Another impressive problem is undoubtedly the inter-class variability. See Figure 1.4. We do not want our systems confuse between the mountain bike class and the touring bike class, *i.e.* classes that can be easily confused because they have a very similar shape and appearance.

**Illumination** The illumination has to be taken into account too, *i.e.* we have to be able to recognise object classes under different illuminations (*e.g.*



Figure 1.5: Changes of appearance due to changes in the illumination conditions.

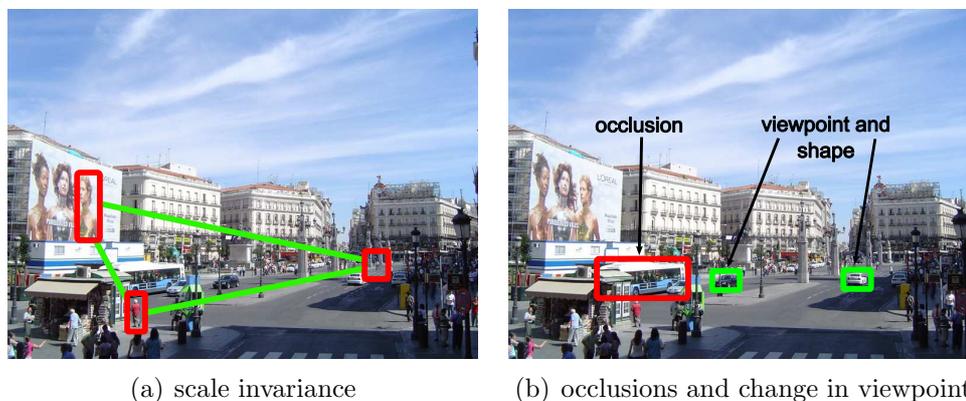


Figure 1.6: Scale invariance, occlusions and changes in viewpoint and shape. This figure is best viewed in colour.

Figure 1.5). It is difficult for a computer vision system to deal with these variations in appearance due to a change in the illumination conditions.

**Scale invariance, occlusions and changes in viewpoint** The same object category seen under different scales or from different viewpoints has to be categorised within the same class. If we look again at the picture of the square in downtown Madrid, we are able to, for example: recognise humans at very different scales within the picture (see Figure 1.6(a)); identify the bus although it is partially occluded and recognise the cars from different viewpoints (*e.g.* Figure 1.6(b)).

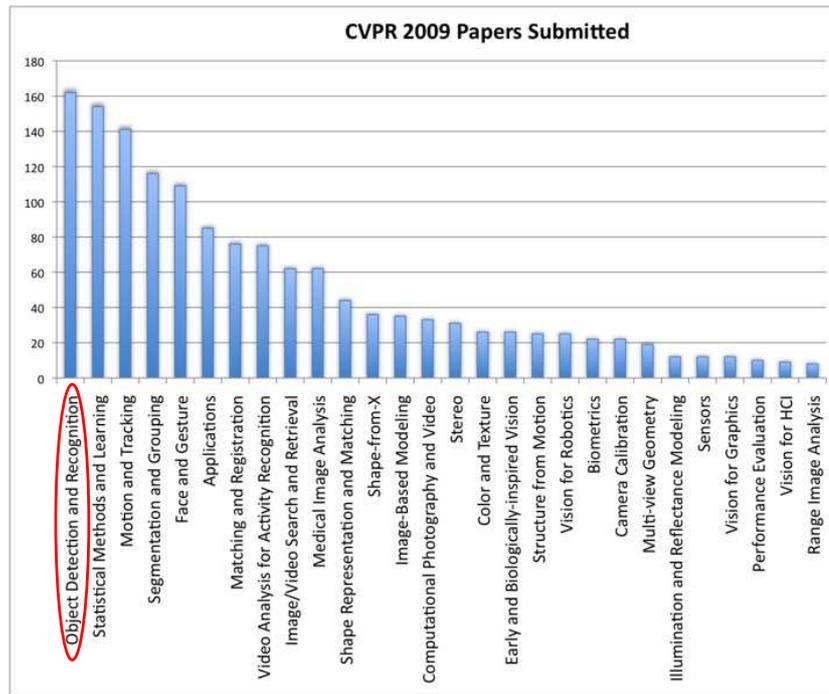


Figure 1.7: Submitted papers by primary subject area to the CVPR 2009. Plot reproduced from (CVPR 2009).

## 1.2 Motivation

Object recognition and detection is a very active topic of research. For instance, in the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR) 2009, this subject area received the highest number of submitted papers (see Figure 1.7). This indicates that we clearly deal with an unsolved problem.

However, there has been tremendous progress in object class recognition during the last 10 years. The category-level object recognition technology has reached a point at which impressive applications are becoming possible, and the future is bright.

For example, there is a large need for effective and efficient tools for multimedia information retrieval. In order to face the limitations of traditional information systems, there is a very active area of research devoted to content-based indexing via automatic object recognition techniques. Some examples of applications are: database annotation, video annotation or image retrieval.

Moreover, Internet image search engines are becoming *true* image search engines, *i.e.* the image engines can take queries as images, so we can use pictures to search the web (*e.g.* Like.com (Like.com 2005), Kooaba (Bay and Quack 2006), Google goggles (Google 2010)).

But the applications of object categorisation go far beyond that. Driver assistance, traffic signs recognition, interactive games, autonomous robots and video surveillance are just some examples.

## 1.3 Outline of the thesis

This dissertation is organised as follows:

- In **Chapter 2** we start reviewing existing work in the field of category-level object recognition. Then we focus on those works devoted to learning discriminative visual vocabularies (Section 2.2). Special attention is given to those approaches that try to group semantically meaningful object parts, hence narrowing the semantic gap.
- Freely available databases of object categories have played a key role in the tremendous progress made in object classification and detection. However, 3D categorisation systems have traditionally had specific datasets isolated from the other category-level oriented ones. In **Chapter 3** we introduce the new dataset Image Collection of Annotated Real-world Objects (ICARO) which combines sets of images that enable the 3D categorisation systems to learn a representation of the 3D geometry of the object classes, with a set of real-world challenging images with: high quality annotation and significant variability in terms of object size, pose, illumination, position, orientation and occlusion. We present a comparative analysis of ICARO with the existing datasets used within the context of object categorisation and detection. A benchmark for both object classification and detection (in 2D) is stated as well.
- **Chapter 4** discusses the proposed approaches for building class representative visual words. We first focus on an approach, the Cluster Precision Maximisation (CPM), for obtaining class representative visual words by maximising a new cluster precision criterion. Next, a novel adaptive threshold refinement scheme is proposed. Using correlation clustering techniques, we introduce an algorithm for increasing vocabulary compactness while at the same time improving the recognition rate and further increasing the representativeness of the visual

words for category-level object recognition. To enhance the readability of this chapter some mathematical analyses are in Appendix A.

- **Chapter 5** deals with the problem of efficient hierarchical clustering algorithms in high-dimensional spaces. We present a speeded up version of the Reciprocal Nearest Neighbours (RNN) algorithm based on the projection search paradigm: the Fast-RNN.
- In **Chapter 6** we propose a novel approach for building efficient visual codebooks using clustering aggregation techniques. A rigorous approach for adding meaningful spatial coherency among the local features into the codebooks is described.
- Finally, **Chapter 7** presents the conclusions of this dissertation, states the main contributions and discusses future work.

# Chapter 2

## State of the art

*If I have seen further it is only by standing on the shoulders of giants.*

Sir Isaac Newton.

---

In this chapter we start reviewing the most recent and significant works in the literature on category-level object recognition. Next, we focus on different methods for building discriminative visual vocabularies within the same context. Special attention is given to methods bridging the gap between visual features and semantic concepts. For a more detailed overview of the literature that is specific to *correlation clustering* and *clustering aggregation*, we refer to Sections 4.3.1 and 6.2 respectively.

### 2.1 Category-level Object Recognition

The research goals of category-level object recognition are to detect objects in images and to categorise them, *i.e.* to determine the generic classes they belong to (*e.g.* car, chair, person, dog). This is clearly in contrast to the recognition of specific, individual objects.

Recent works have established several categorisation methods that are based on local salient structures in the images. Some of them use just a Bag-of-Features (BoF) model, while others include a certain amount of geometric modeling of 2D and/or 3D spatial relations between parts, or constellations of parts. For instance, one can structure the literature as follows:

- appearance-based.

- keypoint-based.
- contour/shape-based.
- graph-based.
- 3D reconstruction-based.

However, along with the explosion of image and video data on the Internet, there have been appearing in the literature models for large-scale object recognition. The question that immediately rises is whether traditional methods, *e.g.* BoF, can deal with such amount of data.

With this in mind, we have decided to structure this section in two big blocks: single object class recognition, and multiple object class recognition. For a more detailed overview of the object categorisation literature we refer to the works (Pinz 2006, Fei-Fei et al. 2009).

## 2.1.1 Single Object Classes

### 2.1.1.1 Bag-of-Parts Models

Bag-of-parts approaches model the object class as a collection of object parts without any geometric relations between them. The idea is motivated by the success of the Bag-of-Words (BoW) technique in text categorisation and text information retrieval, where documents are represented as a vector of word counts. Normally, the object parts are vector-quantised local features, in which case the method is called a BoF or BoW method.

Vector-quantised local features have been referred to as *textons* (Leung and Malik 2001), *object parts* (Fergus et al. 2003), *visual words* (Sivic and Zisserman 2003) and *codebooks* (Leibe and Schiele 2003). A BoW representation is built as a histogram of visual word occurrences (Sivic and Zisserman 2003, Csurka et al. 2004). This image representation has been shown to characterise the images and objects within it in a robust yet descriptive manner, in spite of the fact that it ignores the spatial configuration between visual words.

A simple BoW approach consists of 3 main parts:

- local feature extraction (and selection).
- vocabulary construction (using vector quantisation algorithms).
- discriminative classifier to determine whether an image contains objects of a determined class.

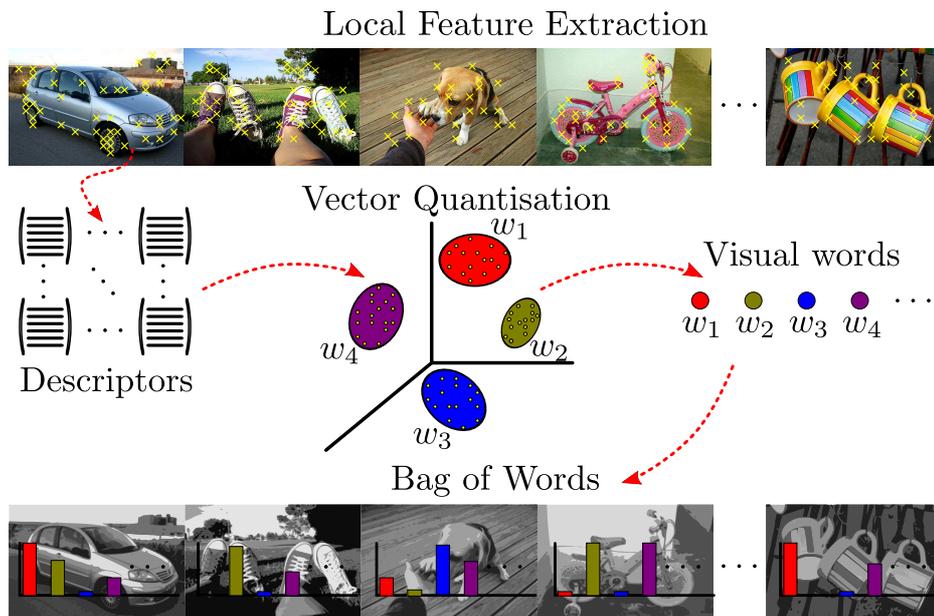


Figure 2.1: BoW system overview.

An overview of this type of approach is shown in Figure 2.1. These BoW systems have shown impressive results lately (van de Sande et al. 2008, Zhang et al. 2007, Tuytelaars et al. 2009). Variations on the BoW scheme won the recent PASCAL Visual Object Classes (PASCAL VOC) Challenge on object classification (Everingham et al. 2007, 2008).

Grauman and Darrell (2005) describe an efficient approach to compute a matching score between two sets of features using the pyramid match kernel. Instead of first quantising the local features into visual words, the authors use this kernel in a SVM. The basic idea of the method is to map sets of features to multi-resolution histograms, and then compare the histogram with a weighted intersection measure. The objective is to approximate the similarity of the best partial matching between the feature sets.

The approach proposed by Lazebnik et al. (2006) extends the work of Grauman and Darrell (2005). The novel technique works partitioning the image into increasingly fine sub-regions and computing histograms of local features found inside each sub-region. The authors show that the approach significantly improves the performance on challenging scene categorisation tasks. This image representation technique has shown an excellent performance in several recent works, *e.g.* (Chum and Zisserman 2007, van Gemert et al. 2008, Tahir et al. 2009, van de Sande et al. 2010). Furthermore, in this dissertation we follow the image representation approach of (Lazebnik et al.

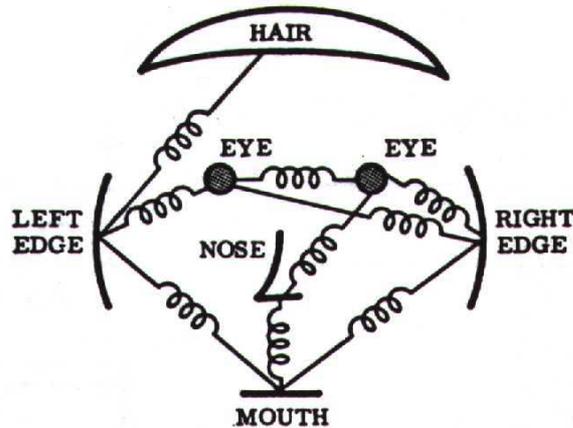


Figure 2.2: The parts structure model of Fischler and Elschlager (1973).

2006) for evaluating the proposed (semantic) visual codebooks.

There are other works that add spatial information to the BoF model. Sudderth et al. (2005) build a hierarchical probabilistic model based on a set of parts which describe the expected appearance and position of low-level features. Each object class has its own distribution over these parts. Savarese and Criminisi (2006) incorporate appearance and shape information jointly by simply concatenating histograms of visual words with histograms of *correlatons*, *i.e.* vector-quantised correlograms. Niebles and Fei-Fei (2007) present a novel hierarchical model that can be characterised as a constellation of BoF and that is able to combine both spatial and spatial-temporal features. The model combines the strong shape representation of the constellation model (*i.e.* (Fergus et al. 2003)) with the large number of features that a BoW model utilises.

### 2.1.1.2 Part-based models

Observing the merits of pure BoF approaches, it seems obvious to try to extend the idea for building models that capture the spatial relationship between parts. Fischler and Elschlager (1973) were the first to propose what is often termed the *parts and structure model*. In contrast to the orderless BoF models, this model consist of a series of parts arranged in some geometric configuration, *i.e.* the structure. The parts are flexibly related to each other. Such a model can be deformed to a certain degree (one can image the different parts being connected by springs, see Figure 2.2).

So, the idea behind this part-based model is to construct a model with individual parts linked by an spatial model. To fit the model means to

minimise a cost function which comprises the local fit for each part plus a global deformation term. For a more complete review of recent work in this direction we refer to (Felzenszwalb and Huttenlocher 2005).

The *constellation* model (Weber et al. 2000a, Fergus et al. 2003) is another geometrically rigid part-based model where the object is modelled as a set of parts and the geometry of the parts is modeled by using a generative model. The main limitations of the constellation approach lie in the prohibitive complexity of learning models with many parts, and in the amount of required supervision.

Another interesting work within the same category is the Implicit Shape Model (ISM) described in (Leibe et al. 2004, 2008b). While previously discussed approaches model the shape explicitly in terms of constellations, Leibe et al. propose a codebook of local appearance and an implicit shape model. The ISM maps the location of the visual words relative to the object centre by a probabilistic voting scheme.

### 2.1.1.3 Discriminative approaches

Within this section we find the classifier-based methods. The object class recognition (and detection) is formulated as a classification problem. Basically, the image is divided into a set of overlapping subwindows and the classifier is evaluated in each of them to determine whether the particular object class appears. In (Dalal and Triggs 2005), grids of Histogram of Oriented Gradients (HOG) are used in conjunction with linear SVMs for human detection. Chum and Zisserman (2007) propose a combination of a dense histogram of edge orientations and a sparse histogram of visual words. The image representation technique of (Lazebnik et al. 2006) is used for building the histograms.

Felzenszwalb et al. (2008) propose the use of HOG descriptor combined with a linear SVM detector, but they enrich the model proposed by (Dalal and Triggs 2005) using a star-structured part-based model defined by a *root* filter plus a collection of part filters and associated deformation models. To allow the examination of every location in the image Felzenszwalb et al. (2008) use distance transforms (Felzenszwalb and Huttenlocher 2004b).

Other interesting work is (Lampert et al. 2009), where a branch and bound scheme that allows efficient maximisation of a large class of quality functions over all possible subimages is proposed. They introduce an efficient subwindow search technique to efficiently predict the best location of an object in an image. The gain in speed and robustness allows the use of better local classifiers.

#### 2.1.1.4 3D Object Categorisation

By modeling objects and their relations in 3D, it is possible to provide robustness to changes in pose and viewpoint. That is why the true 3D object categorisation problem has been recently investigated. The main challenges these approaches have to confront are:

- to model the shape variability.
- to model the appearance variability.
- to link appearance across different views.

In the literature we can find approaches dealing with the problem of single 3D object recognition, *e.g.* (Ullman 1998, Murase and Nayar 1993, Lowe 2001, Rothganger et al. 2006, Ferrari et al. 2006, Kushal and Ponce 2006).

Within the domain of category-level object recognition, Weber et al. (2000b) present an approach for learning models of human heads for the purpose of detection from different viewing angles. Thomas et al. (2006) incorporate shape and appearance information into a 3D object model. First, they build an ISM (Leibe et al. 2004) for each viewpoint. Then, the method described in Ferrari et al. (2006) is used for matching the images of the same object instance across different viewpoints. Region tracks are constructed to transfer the ISM votes from one view to its neighboring viewpoints. In contrast, Kushal et al. (2007) employ a single appearance model for object parts across different viewpoints. The object classes are represented by assembling Partial Surface Models (PSMs). These PSMs are formed of dense, locally rigid assemblies of image features. Pairs of PSMs which regularly occur near each other at consistent relative positions are linked. These local connections are then used to build a probabilistic graphical model for the geometry and appearance of the PSMs making up an object.

Yan et al. (2007) propose a method that establishes spatial connections between views by mapping them directly to the surface of a 3D model, instead of using a mechanism for relating multiple 2D training views. Hoiem et al. (2007) extend the work of Winn and Shotton (2006). They propose to use a coarse 3D model of the object class to roughly correspond physical parts across instances at different viewpoints and to include a description of the colour of the object. Liebelt et al. (2008) render a synthetic model from different viewpoints and extracts a set of poses and class discriminative features. Local features are matched to the synthetically trained ones during detection. In (Xiao et al. 2008), the structural information is represented in

their true 3D locations. The locations and outlines of objects instances, as well as the camera parameters are automatically determined by reconstructing 3D visual word exemplar models.

In the works of (Savarese and Fei-Fei 2007, 2008), the authors propose a model for 3D object categorisation and localisation. An object category is represented as a collection of view-invariant regions linked by transformations that capture the relative change of pose among parts. The model has the ability to generate unseen views, but achieves limited accuracy in classification due to the lack of an explicit background model. In (Sun et al. 2009, Su et al. 2009) probabilistic models for representing multi-view object categories are described. Objects are represented as a coherent ensemble of parts that are consistent under 3D viewpoint transformations. Each part is a collection of salient image features. A generative framework is used for learning a model that captures the relative position of parts within each of the discretised viewpoints. Contrary to most of the existing mixture of viewpoints models, this model establishes explicit correspondences of parts across different viewpoints of the object class.

### 2.1.2 Multiple Object Classes

Within this section we find those models that focus on scaling rather than understanding the images.

First, some vision techniques for large-scale recognition have been developed. For example, efficient matching methods have appeared. Grauman and Darrell (2005) present a new fast kernel function which maps unordered feature sets to multi-resolution histograms and computes a weighted histogram intersection. This pyramid match computation is linear in the number of features, and it implicitly finds correspondences based on the finest resolution histogram cell where a matched pair first appears. Extensions to this pyramid match kernel have been applied to scene recognition (Lazebnik et al. 2006), shape representation (Bosch et al. 2007) and action recognition (Lv and Nevatia 2007).

Other works learn how to compare images, *i.e.* how to exploit similarity constraints to build more useful distance functions, *e.g.* (Varma and Ray 2007, Jegou et al. 2007).

Moreover, some efforts to compact descriptors have been made. Torralba et al. (2008b) transform the Gist descriptor (Oliva and Torralba 2001) to a compact binary code, with a few hundred bits per image. They directly learn a mapping from the image to the binary code.

Within this large scale scenario, we find all those works recognising multiple object categories (Torralba et al. 2004, Opelt et al. 2006, Shotton et al.

2006, Fergus et al. 2009). Biederman (1987) estimates that we humans are able to discriminate 30.000 categories. If we do not care about efficiency, it is possible to use a set of independent binary classifiers (one per category), *e.g.* (Schneiderman and Kanade 2000). Others have started to investigate whether it is possible to share features from one object category to another. Torralba et al. (2004), with a multi-task learning procedure (based on boosted decision stumps), reduce the computational and sample complexity, by finding common features that can be shared across the classes. Furthermore, Fergus et al. (2009) propose a semi-supervised learning approach to be used against gigantic image collections. Semi-supervised learning is a principled framework for combining clean and noisy labels. However, it scales polynomially with the number of images. To obtain highly efficient approximations that are linear in the number of images, they use the convergence of the eigenvectors of the normalised graph Laplacian to eigenfunctions of weighted Laplace-Beltrani operators.

## 2.2 Learning Semantic Visual Words

Several attempts have been made to bring the gap between visual features and semantic concepts. It is possible to categorise these attempts into two major classes: the supervised and unsupervised approaches.

### 2.2.1 Supervised approaches

Local patch annotation is used by Vogel and Schiele (2007) to build a semantic vocabulary by manually associating the local patches to some concepts (sky, water, grass, trunks, foliage, field, rocks, flowers and sand). So, in a first stage, the local image regions are classified into semantic concept classes. In a second stage, this region-wise information of the concept classifiers is combined to a global image representation: the concept occurrence vector, *i.e.* a normalised histogram of the concepts occurrences in an image.

Other supervised approaches use image annotation to guide the semantic visual vocabulary construction (Moosmann et al. 2006, Winn et al. 2005, Yang et al. 2008). Specifically, Moosmann et al. (2006) utilise Extremely Randomized Clustering Forests (ERCF) to organise the vocabulary. They provide a rapid and highly discriminative approach for quantising large numbers of high-dimensional image descriptors into many label classes. The classification trees are obtained first, and the authors assign a visual word label to each leaf. Although the method can be used with unlabelled data, it benefits significantly from labels when they are available. To obtain more compact

vocabularies, Winn et al. (2005) propose a statistical algorithm for learning a compact and yet discriminative appearance-based object class models. An optimally compact visual dictionary is learnt by pair-wise merging of visual words from a initially large dictionary (quantised by the  $K$ -means algorithm). Gaussian Mixture Models (GMMs) are used to describe the final visual words.

Several other methods use Mutual Information (MI) between the features and class labels to create the semantic vocabulary from an initial and relatively larger vocabulary (Fulkerson et al. 2008, Lazebnik and Raginsky 2007).

Yang et al. (2008) propose a novel framework for object category recognition that unifies visual codebook construction with classifier training. Each image feature is encoded by a sequence of *visual bits* optimised for each category. The optimal *visual bits* and their associated weights are identified following an iterative algorithm.

Perronnin et al. (2006) define a universal vocabulary, which describes the visual content of all the considered classes, and class vocabularies, which are obtained through the adaptation of the universal vocabulary using class-specific data. The images are characterised using a set of category-specific histograms, where each histogram describes whether the content can best be modeled by the universal vocabulary or by its corresponding category vocabulary. Another interesting work is (Perronnin and Dance 2007), where the use of Fisher Kernels (Jaakkola and Haussler 1999) is proposed, as their gradient representation has much higher dimensionality than a histogram representation, resulting in very compact vocabularies yet highly informative representations.

### 2.2.2 Unsupervised approaches

Some unsupervised approaches have been inspired by the success of the textual topic models in text categorisation, *e.g.* Probabilistic Latent Semantic Analysis (PLSA) (Bosch et al. 2006, Quelhas et al. 2005, Sivic et al. 2005) and Latent Dirichlet Allocation (LDA) (Fei-Fei and Perona 2005). In all these works an image is represented as the mixture distribution of hidden topics that can essentially be a semantic visual vocabulary.

Others have tried to add more local geometric information to their codebook generation algorithms. Lazebnik et al. (2004) construct a codebook with groups of nearby regions whose appearance and spatial configuration occur repeatedly in the training set. Leibe et al. (2008a) present how to learn semantic object parts for object categorisation. They use what they call co-location and co-activation to learn a visual vocabulary that generalises beyond the appearance of single objects, and often obtains semantic object

parts.

There are several works based on frequent itemset mining (Gilbert et al. 2009, Quack et al. 2007, Sivic and Zisserman 2004, Yuan et al. 2007). Typically, finding representative visual words boils down to finding frequent co-occurring groups of descriptors in a transaction database obtained from the training images (Sivic and Zisserman 2004). Quack et al. (2007) apply *Association rule* data mining to object recognition by mining spatially grouped SIFT descriptors. Yuan et al. (2007) generate a higher-level lexicon, *i.e.* visual phrase lexicon, where a visual phrase is a meaningful spatially co-occurrent pattern of visual words. This higher-level lexicon is much less ambiguous than the lower-level one.

Another interesting work uses diffusion maps to learn a semantic visual vocabulary from abundant quantised midlevel features using *K*-means (Liu et al. 2009).

# Chapter 3

## Datasets

*...but Parmenides, I think the most likely view is, that these ideas exist in nature as patterns, and the other things resemble them and are imitations of them; their participation in ideas is assimilation to them, that and nothing else.*

Plato, *Parmenides* (132d).

---

In this thesis, we will evaluate our visual vocabularies and categorisation algorithms on different datasets recently used in the literature. This chapter starts introducing the variety of datasets used within the context of category-level object recognition. Moreover, we present a new database of images called Image Collection of Annotated Real-world Objects (ICARO). ICARO is an image collection of more than 3900 images, including ground-truth labels for category-level object recognition and detection. ICARO provides especially designed sets of images that enable the systems to learn suitable representations of the 3D geometry of the object classes that can be exploited for recognition. This chapter provides a detailed quantification of ICARO contents in their current state, as well as a comparison with the existing datasets used within the context of object detection and image categorisation. We have benchmarked ICARO using two of the state-of-the-art approaches for both classification and detection of object classes. Results confirm that ICARO can be considered to be a challenging dataset. ICARO has been made publicly available for scientific research purposes:

<http://agamenon.tsc.uah.es/Personales/rlopez/data/icaro/>

### 3.1 Introduction

Take a picture of a crowded street. When you look at it, you will discover thousands of objects within the scene. Furthermore, you are able not only to categorise them, but to estimate their locations and poses: you understand the scene. It seems that humans are able to recognise about 30.000 object categories (Biederman 1987). However, what comes so naturally to us is tremendously difficult for a computer vision system. Changes in lighting, viewpoint, pose, as well as intra-class differences, lead to enormous appearance variation, making the problem extremely challenging.

Several recent works have achieved very impressive results in detection and recognition of a few object classes (Dalal and Triggs 2005, Felzenszwalb et al. 2008, Lazebnik et al. 2006, van de Sande et al. 2008, Zhang et al. 2007), as well as in scene understanding (Hoiem et al. 2006, Thomas et al. 2007).

Publicly available datasets have played a key role in the success of recent category-level recognition systems. The Internet and the digital era offer an immeasurable amount of images and video. There are more than 3 billion photos and videos on Flickr and Youtube respectively, and an even larger number of images in Google Image Search. This means that there is currently a huge amount of information that may be used for learning visual class models and for testing the performance of detection and classification algorithms.

Image databases specially designed for training category-level object recognition systems are essential for scaling the existing methods to thousands of object categories. Two extremely large datasets have recently appeared (Torralba et al. 2008a, Deng et al. 2009). Both are large-scale ontologies of images built upon the backbone of the WordNet (Fellbaum 1998) structure. The aim is to explore how the data itself can help to solve the problem of category-level object recognition.

Moreover, while great progress has been achieved in 2D-based recognition approaches, very little work has been done to address the considerable problem of true 3D object categorisation (Savarese and Fei-Fei 2007, Sun et al. 2009, Thomas et al. 2006, Xiao et al. 2008). The objective is not only to classify the object, but also to infer its pose, view and scale. Image databases that enable to learn the 3D geometry of object classes for use in recognition are therefore also needed.

In recent years, the vast majority of efforts have sought to develop better models and parametric representations for recognition, with less attention paid to data. However, some directives have been expressed by Ponce et al. (2006) and by the PASCAL VOC Challenge standard (Everingham et al. 2009a) for designing datasets that indeed allow the progress in algorithms

capability to be assessed. This is achieved by annotating more realistic and less restrictive image conditions: multiple object class instances within a single image, with partial occlusions and truncation, with viewpoint and scale variations, with annotated views (rear,front,left,...). This control over the annotation stage is extremely costly, and does not scale very well to large datasets.

ICARO is not just another dataset of images, although that would be useful too. It is a database of object categories that closely follows the guidelines mentioned in (Ponce et al. 2006), and it is structured so that it allows the *true* 3D object categorisation systems to work with it. These systems have traditionally had specific databases, consisting of several objects recorded from different and (more) controlled viewing angles. These image collections are tailored to solving a specific problem, and are isolated from the other category-level oriented datasets. ICARO brings together the two types of datasets providing annotations for both training and test sets, and makes them publicly available.

In this chapter we first review the existing state-of-the-art datasets for category-level object recognition (Section 3.2). Section 3.3 includes a detailed quantification of the current state of ICARO contents. Section 3.4 includes a comparative analysis of ICARO and some other datasets described in Section 3.2. A benchmark for both object categorisation and detection is established in Section 3.5. Finally, the chapter concludes in Section 3.6.

## 3.2 Other datasets

In this section we perform a brief comparison of the existing state-of-the-art image datasets, in order to establish a benchmark for further comparison with ICARO.

There are many publicly available databases, and they can not all be reviewed in depth here. However, a comparison of most of them is possible. Very different criteria can be used in this analysis. Datasets may be designed for specific object instance detection, or for category-level object recognition. They may differ in the number of classes they contain, or in whether the objects are embedded in a real scene or isolated with little or no clutter. Another key aspect is the annotation, which may be or may not be provided. The type of annotation is also a database characteristic, and there are several options: image labels, bounding boxes, complex polygons and segmentation masks. The datasets can be divided into two main groups by their scale.

### 3.2.1 Large Scale Datasets

First in this group is the Caltech family of datasets, Caltech-101 (Fei-Fei et al. 2004) and Caltech-256 (Griffin et al. 2007). They have become a *de facto* standard for evaluating algorithms for multi-class category-level recognition. Moreover, they are two of the most diverse datasets in terms of inter-class variability. The principal aim of this family of datasets is to investigate multi-category object recognition with a limited number of training images.

The MIT-CSAIL database (Torralba et al. 2004) also belongs to this group. It contains more than 160.000 images, but the number of labelled objects and object categories increases over time thanks to the publicly available web-based annotation tool LabelMe (Russell et al. 2008). The dataset is mostly incompletely labelled, which means that it is impossible to estimate precision and recall accurately as in the PASCAL VOC Challenge. However, it has an incalculable value as a source of training images.

Another large scale database is the TinyImage (Torralba et al. 2008a): a dataset of 80 million of  $32 \times 32$  low resolution images, collected by sending all words in WordNet (Fellbaum 1998) as queries to image search engines. Each synset contains 1000 images, of which 10 – 25% are possibly clean images.

More recently, ImageNet (Deng et al. 2009) has appeared. This is another large scale hierarchical image database, again organised according to the WordNet hierarchy. In ImageNet the aim is to provide an average of 1000 clean and full resolution images to illustrate each synset. The images of each concept are quality-controlled and human-annotated using the Amazon Mechanical Turk service, an on-line platform where tasks can be uploaded for users to complete and to get paid.

### 3.2.2 Small Scale Datasets

Most of the available datasets belong to the this group: the UIUC image database for Car detection (Agarwal et al. 2004), TU Darmstadt (Leibe et al. 2004), TU Graz02 (Opelt et al. 2006), and the MSRC (Shotton et al. 2006) among others.

The PASCAL VOC Challenge datasets (Everingham et al. 2009a) may also be in this group. The last challenge dataset, *i.e.* 2009 (Everingham et al. 2009b), provides a total of 14.743 images of 20 classes. The PASCAL VOC Challenge has been groundbreaking in the provision of high quality annotation of challenging images. Furthermore, the challenge and its associated dataset have become accepted as the benchmark for object detection.

The small scale datasets also include those especially built for modeling both objects and classes in 3D. Illustrative examples are: COIL-100 (Nene

et al. 1996), SOIL-74 (Burianek et al. 2000), ALOI (Geusebroek et al. 2005), the Kushal & Ponce 3D Object Recognition Dataset (Kushal and Ponce 2006) and the ETHZ Toys dataset (Ferrari et al. 2006). There are also the datasets for 3D generic object categorisation collected by (Savarese and Fei-Fei 2007) and (Thomas et al. 2006).

### 3.2.3 Summary of the Databases

Table 3.1 provides a comparison of all the databases mentioned above. Several aspects are analysed, including the number of images and classes; whether the database is designed to allow the recognition of object classes or particular object instances; whether the objects are in real-world scenes; whether the objects appear with occlusions, in different positions, on different scales and from different viewpoints; whether the images contain multiple objects; the type of Annotation (if any) –*e.g.* bounding box, polygon, segmentation mask or class label–; and whether the viewpoints (front, rear, right, left, ...) are annotated. ICARO has also been included in Table 3.1 for comparison purposes, though the database is reviewed in depth in the next section.

## 3.3 ICARO

ICARO is a dataset of images especially designed for category-level object recognition and detection. The goal of ICARO is to establish a benchmark for investigating the performance of classification and detection methods over a wide spectrum of real-world images. Furthermore, ICARO has been designed to establish a framework for learning visual models of 3D object categories. This section offers a detailed quantification of ICARO contents as of November 20, 2009.

### 3.3.1 ICARO in Numbers

The dataset consists of 3.993 images of 26 classes, with a total of 10.872 annotated objects. The classes we have chosen are: apple, banana, bike, bottle, camera, car, chair, dustbin, foot, fork, glasses, hand, hat, knife, laptop, mobile phone, monitor/TV, motorbike, mug, phone, pistol, potted plant, shoe, sofa, telephone cabin and traffic light. This selection of object classes can be organised in a taxonomy with 7 main branches (Figure 3.1): household items, person, fruits, outdoor, vehicles, clothes and weapons.

Database	IR/CR	# Classes	# Images	RS	O	DP	DS	DV	MO	A	AV
Caltech-101	CR	102	9146	✓	-	-	+	-	-	BB	□
Caltech-256	CR	257	30607	✓	+	+	+	+	+	IL	□
MIT-CSAIL	CR	183	>160000	✓	++	++	++	++	++	P	□
TinyImages	CR	>50000	>80000000	✓	+	+	+	+	+	P	□
ImageNet	CR	>14000	>9000000	✓	++	++	++	++	++	IL	□
UIUC	CR	2	1328	✓	-	-	-	-	+	BB	□
TU Darmstadt	CR	3	327	✓	-	+	+	-	+	BB, SM	□
TU Graz02	CR	4	1476	✓	++	++	++	++	++	SM	□
MSRC	CR	21	591	✓	++	++	++	++	++	SM	□
PASCAL-2005	CR	4	2655	✓	++	++	++	++	++	BB, SM	✓
PASCAL-2006	CR	10	5304	✓	++	++	++	++	++	BB	✓
PASCAL-2007	CR	20	9963	✓	++	++	++	++	++	BB, SM	✓
PASCAL-2008	CR	20	10057	✓	++	++	++	++	++	BB, SM	✓
PASCAL-2009	CR	20	14743	✓	++	++	++	++	++	BB, SM	✓
<b>ICARO</b>	CR	26	3993	✓	++	++	++	++	++	BB	✓
COIL-100	IR		7200	□	-	-	-	++	-		✓
SOIL-47	IR		1996	□	-	-	-	++	+		✓
ALOI	IR		110250	□	-	-	-	++	-		✓
Kushal & Ponce	IR		246	□	-	-	++	+	-		✓
ETHZ Toys	IR		63	□	-	-	-	+	+		✓
Savarese	CR	8	~7000	✓	-	-	++	++	-	SM	✓
Thomas	CR	2	1468	✓	-	-	-	++	++	SM	✓

Table 3.1: Comparison of the main datasets of visual objects. IR: Instance Recognition. CR: object Class Recognition. RS: objects in Real-world Scenes. O: Occlusions. DP: Different Positions. DS: Different Scales. DV: Different Viewpoints. MO: Multiple Objects. A: type of Annotation (BB: Bounding Box, SM: Segmentation Mask, IL: Image Label, P: Polygon). AV: Annotated Viewpoint.

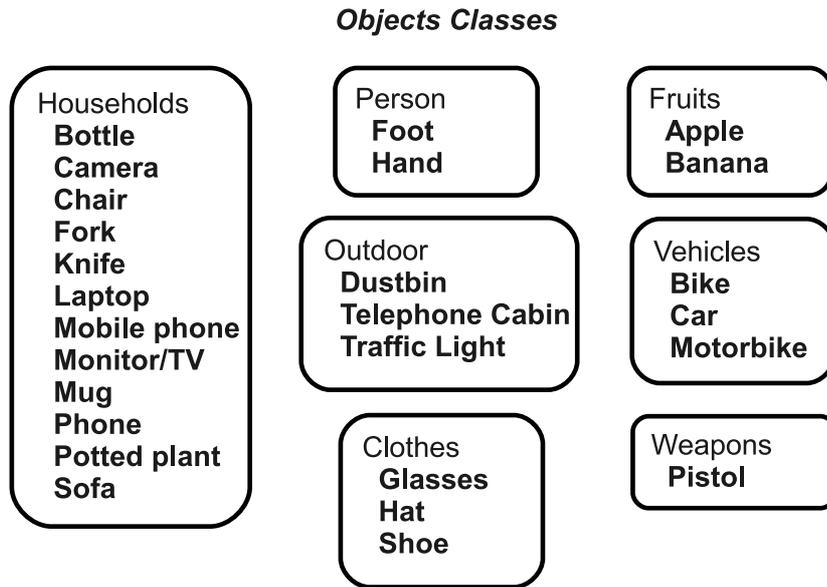


Figure 3.1: ICARO classes organised in a taxonomy of 7 main branches: household items, person, fruits, outdoor, vehicles, clothes and weapons.

The images were mainly downloaded from Flickr<sup>1</sup>, although some images were taken by us (24, 3%). The images must contain the object class of interest, but in a real-world scene. In other words, if the class *hand* is going to be annotated, we not only search for images with the tags ‘hand’, ‘hands’, ‘gloves’, etc., but also for tags like ‘volleyball’. The image collection procedure was performed strictly following the recommendations provided in (Ponce et al. 2006). This is essential to ensure variability in terms of object pose, size, illumination and truncation. Table 3.2 summarises the dataset statistics.

The annotation of each object was performed by a bounding box and it was stored in an Extensible Markup Language (XML) format compatible with both LabelMe (Russell et al. 2008) and PASCAL VOC Challenge (Everingham et al. 2009a) datasets annotations. Based on the annotation guidelines provided by the PASCAL VOC Challenge, every annotated object in ICARO also has the following attributes: class, bounding box, viewpoint, truncation, occlusion and difficulty. The viewpoint tag is one of the following: frontal, frontal-left, left, rear-left, rear, rear-right, right, frontal-right, top, bottom and unspecified. While PASCAL VOC Challenge datasets pro-

<sup>1</sup>Use of these images must respect the Flickr terms of use: <http://www.flickr.com/terms.gne?legacy=1>.

Class	# Images	# Objects
Apple	304	409
Banana	18	30
Bike	111	137
Bottle	163	218
Camera	403	474
Car	351	639
Chair	137	192
Dustbin	305	596
Foot	282	535
Fork	265	369
Glasses	523	633
Hand	1224	2043
Hat	345	426
Knife	338	506
Laptop	21	23
Mobile phone	39	42
Monitor/TV	81	82
Motorbike	68	83
Mug	304	532
Phone	185	213
Pistol	256	437
Potted plant	75	89
Shoe	469	960
Sofa	62	63
Telephone cabin	243	324
Traffic light	333	769

Table 3.2: ICARO 2009 in a nutshell.

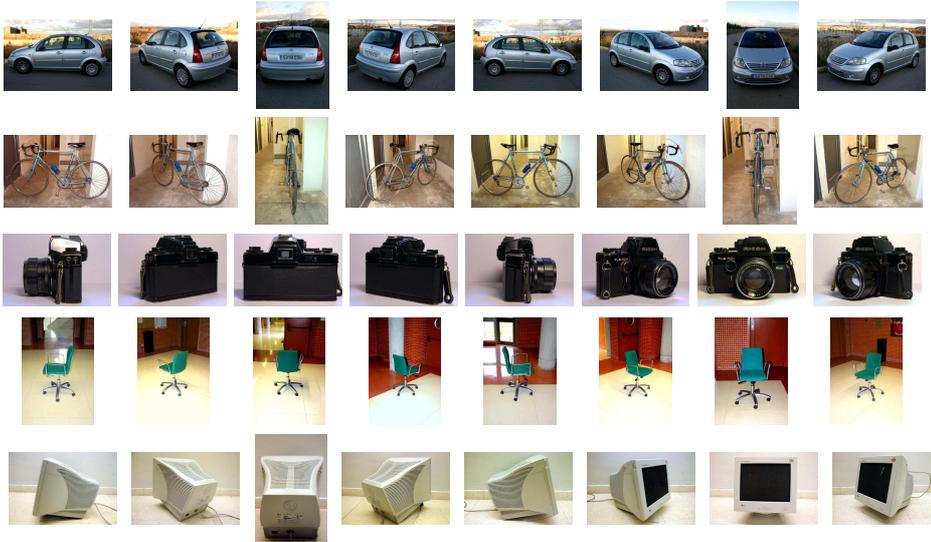


Figure 3.2: Sample images of the MS in ICARO. Only five object classes are represented in this figure: Car, Bike, Camera, Chair and Monitor/TV.

vide 5 possible viewpoints, ICARO increases the number to 11, in order to provide a more precise dataset for methods which use different viewpoints. Truncation occurs when the objects extend beyond the images, while occlusions are tagged when the objects are occluded within the image, *e.g.* a bicycle behind a car. Finally, the difficulty label is assigned to objects that are difficult to detect due to various factors: illumination, size, etc.

ICARO consists of two image sets: Main Set (MS) and Secondary Set (SS). The MS includes images from various object categories that have been captured under 8 controlled viewing angles, as Figure 3.2 shows. This set has been especially designed for training systems which incorporate the shape and appearance information into a 3D object-class model (*e.g.* (Thomas et al. 2006, Savarese and Fei-Fei 2007, Xiao et al. 2008)). It consists of 13 object classes (bike, bottle, camera, car, chair, glasses, hat, mobile phone, monitor/TV, motorbike, phone, potted plant and sofa) and contains 584 images. Each class set is listed in Table 3.3.

The SS includes images from various real-world scenes containing the categories of interest. The set currently comprises 26 classes, with a total of 3.409 images, and 10.288 annotated objects. Statistics for the number of images and annotated poses are shown in Tables 3.4 and 3.5 respectively.

Figure 3.3(a) shows a histogram of the number of annotated images as a function of the percentage of labelled pixels per image. The graph shows

Object Class	# Images	# models
<b>Bike</b>	40	5
<b>Bottle</b>	40	5
<b>Camera</b>	48	6
<b>Car</b>	64	8
<b>Chair</b>	40	5
<b>Glasses</b>	40	5
<b>Hat</b>	40	5
<b>Mobile phone</b>	40	5
<b>Monitor/TV</b>	56	7
<b>Motorbike</b>	40	5
<b>Phone</b>	48	6
<b>Potted plant</b>	40	5
<b>Sofa</b>	40	5

Table 3.3: MS statistics of ICARO.

Class	# Images	# Objects	# Occluded	# Truncated	# Difficult
<b>Apple</b>	304	409	154	65	41,1%
<b>Banana</b>	18	30	16	13	60%
<b>Bike</b>	71	97	38	19	51,8%
<b>Bottle</b>	123	178	74	53	73,8%
<b>Camera</b>	355	426	232	72	43%
<b>Car</b>	289	575	291	225	85,6%
<b>Chair</b>	97	152	112	97	78,1%
<b>Dustbin</b>	305	596	138	81	46,5%
<b>Foot</b>	282	535	242	98	26,4%
<b>Fork</b>	265	369	127	114	38,2%
<b>Glasses</b>	483	593	82	58	44,1%
<b>Hand</b>	1224	2043	777	333	80%
<b>Hat</b>	305	386	130	71	16,7%
<b>Knife</b>	338	506	236	90	46,6%
<b>Laptop</b>	21	23	15	10	69,6%
<b>Mobile phone</b>	39	42	9	3	61,9%
<b>Monitor/TV</b>	25	26	6	17	23,2%
<b>Motorbike</b>	28	43	27	13	39,8%
<b>Mug</b>	304	532	176	84	39,5%
<b>Phone</b>	137	165	15	10	55,7%
<b>Pistol</b>	256	437	53	27	5,5%
<b>Potted plant</b>	35	49	22	15	54%
<b>Shoe</b>	469	960	222	102	54,9%
<b>Sofa</b>	22	23	22	15	34,9%
<b>Telephone cabin</b>	243	324	57	45	29,3%
<b>Traffic light</b>	333	769	68	39	34,2%

Table 3.4: SS statistics of ICARO.

Class	Frontal	Rear	Left	Right	Top	Bottom	Other
Apple	157	-	19	30	45	15	143
Banana	4	9	3	6	-	-	8
Bike	10	10	29	17	-	-	31
Bottle	94	-	4	1	2	-	77
Camera	153	27	18	22	46	-	160
Car	60	92	107	65	2	-	249
Chair	27	34	19	25	4	-	43
Dustbin	207	30	50	52	13	-	244
Foot	43	3	27	39	188	107	128
Fork	113	70	-	-	-	-	186
Glasses	260	6	42	28	28	-	229
Hand	271	573	-	-	-	-	1199
Hat	159	33	58	56	30	8	42
Knife	251	-	-	-	-	-	255
Laptop	14	2	3	1	1	-	2
Mobile phone	20	2	2	1	-	-	17
Monitor/TV	18	2	-	-	-	-	6
Motorbike	3	8	10	11	-	-	11
Mug	39	33	124	92	35	-	209
Phone	106	2	8	8	-	-	41
Pistol	18	2	232	147	13	13	12
Potted plant	31	-	10	3	2	-	3
Shoe	167	31	156	139	179	30	258
Sofa	12	-	3	-	1	-	7
Telephone cabin	130	8	37	31	-	-	118
Traffic light	211	58	120	147	-	-	233

Table 3.5: Statistics of the annotated viewpoints in the SS of ICARO. Note that the 11 viewpoints have been collapsed into 7 in this table (in Other we bring together rear-right, rear-left, frontal-right, frontal-left and unspecified)

that 958 images have less than 10% of their pixels labelled and around 19 images have more than 90%. There are 865 images with at least 50% of their area annotated. Figure 3.3(b) shows a histogram of the number of images as a function of the number of objects in the image. On average, there are 3 annotated objects per image in the secondary set. Moreover, there are 573 images with at least 5 annotated objects.

## 3.4 Comparative Analysis of ICARO

This section gives a comparative analysis between ICARO <sup>2</sup> and 3 of the most widely used datasets for object detection and classification: LabelMe (Russell et al. 2008), Caltech-101 (Fei-Fei et al. 2004) and PASCAL VOC Challenge datasets ((Everingham et al. 2006, 2007, 2008, 2009b)).

### 3.4.1 Objects Distribution and Scale

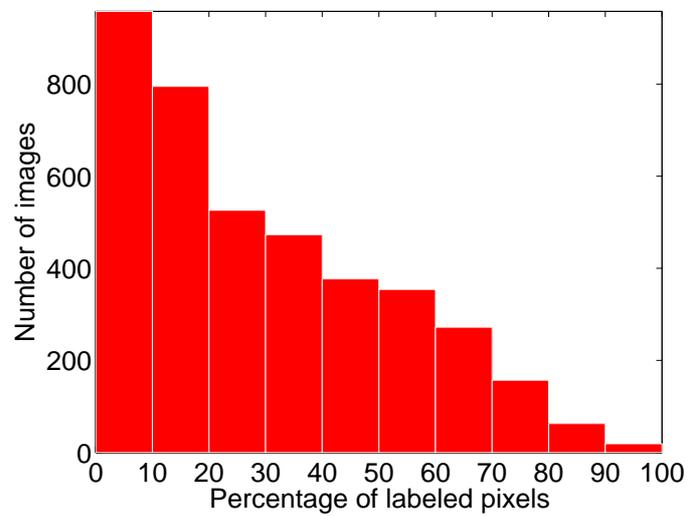
Caltech-101 (Fei-Fei et al. 2004) is one of the most diverse dataset in terms of inter-class variability, but there is a common criticism about the intra-class variability it encompasses. The images have little or no clutter, the variation in pose is limited (the objects tend to appear centred and in a stereotypical pose), and some of them have even been manually aligned to reduce the appearance variability. Caltech-256 (Griffin et al. 2007) solves most of these shortcomings, and it has become a *de facto* standard for multi-class recognition algorithms, but is not recommended for object localisation tests, *i.e.* it still suffers from a limited variation in pose.

Like PASCAL VOC datasets, ICARO also addresses this problem, and presents a highly variable distribution of object location within the images. Figure 3.4 depicts a density plot showing where in the image each object instance occurs, for some object categories of ICARO 3.4(a), the PASCAL VOC Challenge 2007 3.4(d), 2008 3.4(c) and 2009 3.4(b) datasets. It is apparent that the ICARO and PASCAL VOC Challenge datasets present a high level of variability in terms of object location for almost all classes. However, some classes within the PASCAL VOC datasets appear to be concentrated near the centre of the image (*e.g.* the sofa class in PASCAL VOC Challenge 2009). The greater this variability in the object's position, the more suitable for evaluating the database's performance in object detection.

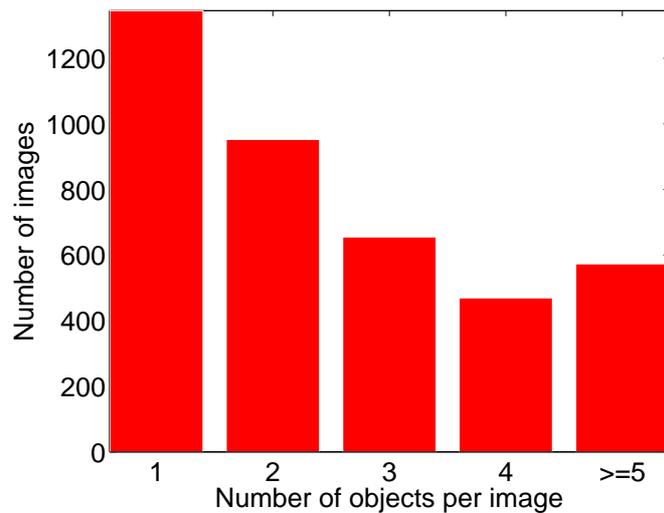
It is also possible to analyse the object sizes these datasets contain. Figure 3.5 shows the bounding box areas as a function of the number of objects in

---

<sup>2</sup>For this comparative analysis we only include images in the secondary set of ICARO, *i.e.* 3409 images with 10288 annotated objects, distributed in 26 object categories.

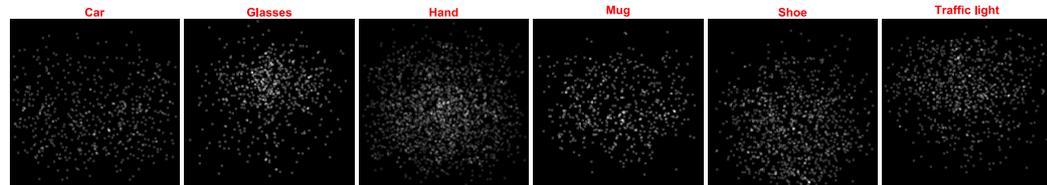


(a)

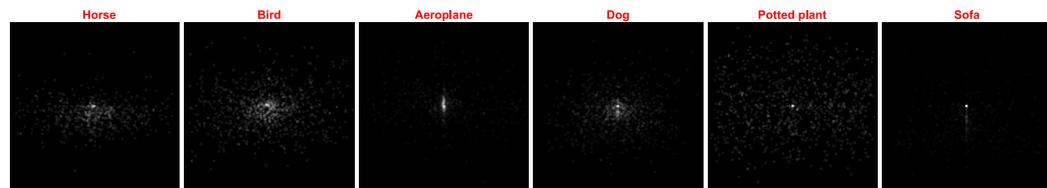


(b)

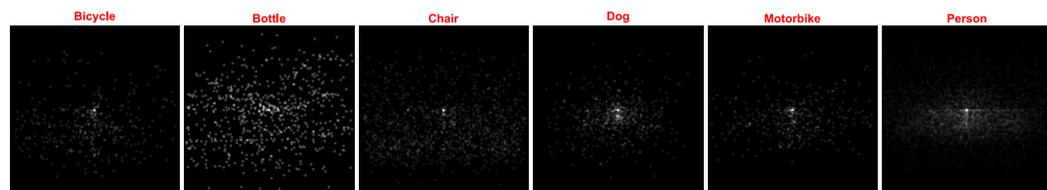
Figure 3.3: Summary of the ICARO dataset content. (a) Histogram of the number of annotated images as a function of the labelled area. The first bin shows that 958 images have less than 10% of labelled area and the last bin shows that there are 19 images with more than 90% of labelled pixels. (b) Histogram of the number of annotated objects per image. The last bin shows that there are 573 images with more than 5 annotated objects.



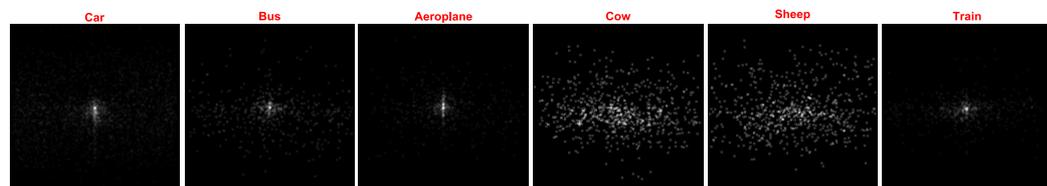
(a) ICARO



(b) PASCAL VOC Challenge 2009



(c) PASCAL VOC Challenge 2008



(d) PASCAL VOC Challenge 2007

Figure 3.4: Examples of distributions of object locations in different databases. These distributions are shown as a black image of  $500 \times 500$  pixels where the object centroids are represented. Note that only the distribution graph of PASCAL VOC Challenge 2007 dataset has been generated with the *trainval* plus the *test* set, for the PASCAL VOC Challenge 2008 and 2009 datasets the *test* set has not been made publicly available.

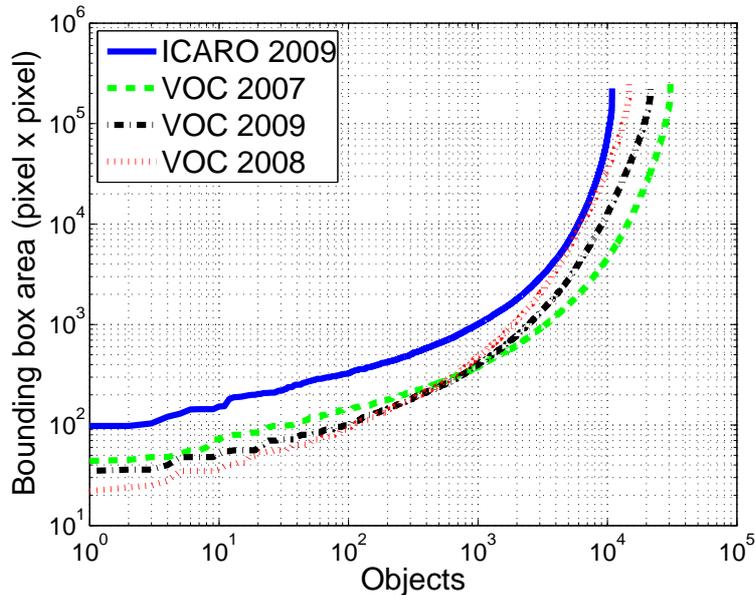


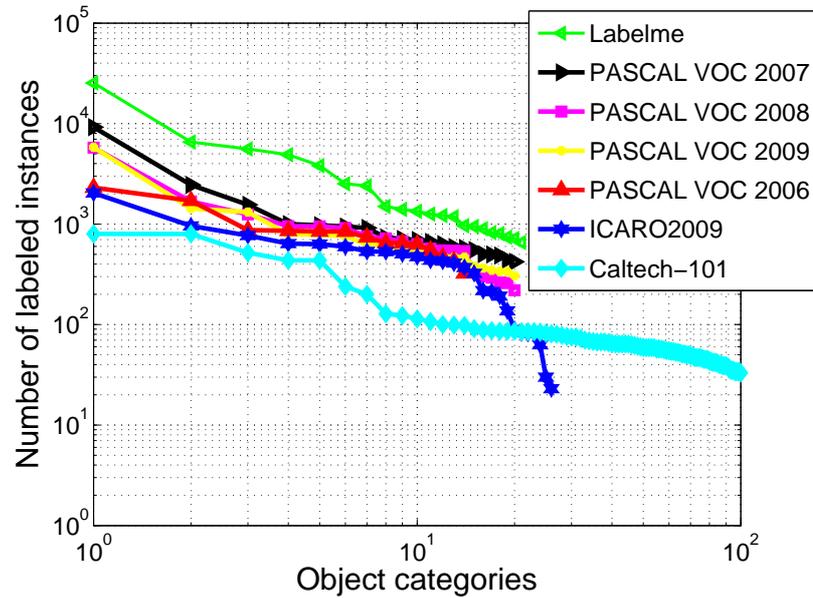
Figure 3.5: Variation of the bounding boxes sizes as a function of the number of objects in the database. The bounding boxes sizes are sorted in ascending order.

the dataset. These areas are sorted in ascending order. All datasets present a continuous variation in object sizes, *i.e.* the curves in Figure 3.5 do not stabilise. Moreover, PASCAL VOC Challenge 2008 dataset presents the biggest difference between the minimum and maximum area that has been annotated.

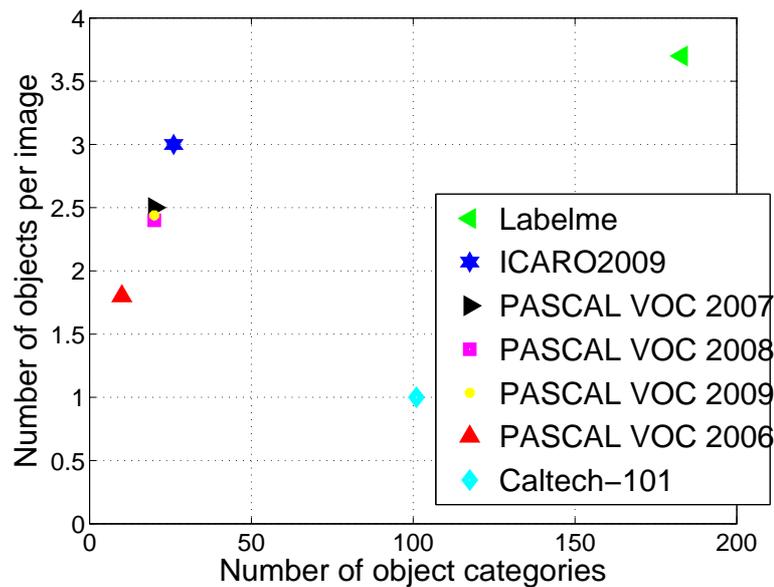
Figure 3.6(a) shows the number of labelled instances per object class for the 7 datasets. Comparison of this aspect shows the superiority of LabelMe, which confirms the success of such a web-based annotation interface. ICARO has to increase the number of labelled instances for some classes, especially those in which the main set images dominate. Figure 3.6(b) depicts the average annotated objects per image for each dataset. LabelMe wins again, while ICARO beats the other datasets.

### 3.4.2 Intra-class Variability

ICARO was built with the aim of objects in images having not only variable sizes and locations, but also variable appearances as well as background clutter and occlusions. Ponce et al. (Ponce et al. 2006) proposed tackling the problem of quantifying intra-class diversity by averaging the RGB values of



(a)



(b)

Figure 3.6: Comparison of 7 datasets used for object detection and recognition: Caltech-101, PASCAL VOC 2006-2007-2008-2009, LabelMe and ICARO. (a) Number of labelled instances per object category, sorted in descending order based on the number of labelled objects. (b) Annotated objects per image, on average, over the entire datasets. For the analysis with LabelMe we only include 30.369 images with 111.490 labelled polygons, resulting in a total of 183 object categories (Russell et al. 2008).

all images for each object class. The averaged images of each class are calculated by first resizing all the images to  $500 \times 500$ . Images with a high level of intra-class diversity will result in a roughly homogeneous field. In (Deng et al. 2009) authors measure the lossless JPEG file size of these average images, as measured by the amount of information therein, whereas we propose calculating the entropy of these averages instead. Figure 3.7 compares image diversity in 3 classes that the Caltech-256, PASCAL VOC Challenge 2007 and ICARO datasets have in common: car, bike and bottle. The ICARO averages have entropies that are always lower than those of Caltech-256. The cars average in the PASCAL VOC 2007 dataset presents the highest entropy level, although the entropies obtain the lowest values for the bike and bottle classes.

Figure 3.8 shows the average images for different object classes in the Caltech-101, Caltech-256, PASCAL VOC Challenge 2006-2007-2008-2009 and ICARO datasets. Over the years, the PASCAL VOC Challenge datasets have evolved into blurrier averages, in which is difficult to ascertain which classes they represent. However, the Caltech-256 inherits almost the same deficiency in terms of intra-class variability as the Caltech-101.

## 3.5 Results

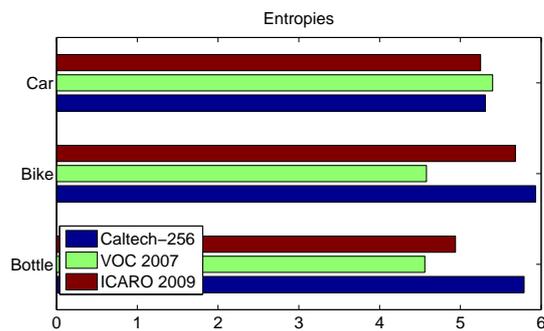
In this section, we report the ICARO dataset results for object categorisation and detection. Our aim is to benchmark the dataset within the image classification and object detection problems. We suggest two testing paradigms to allow further consistent comparisons of different algorithms.

### 3.5.1 Image Classification

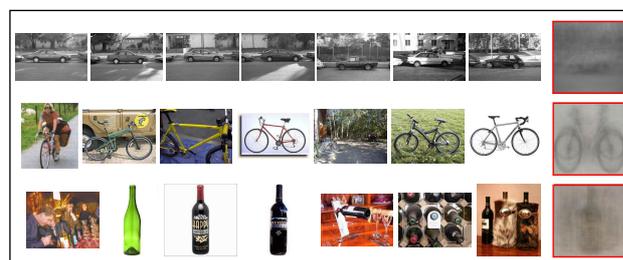
#### 3.5.1.1 Benchmark

A principal aim of the ICARO dataset is to explore how multi-category object recognition systems perform when the number of training images is controlled. First we selected the classes of ICARO that contained more than 100 images. For the ICARO release as of November 20, 2009, these were: apple, bike, bottle, camera, car, chair, foot, fork, dustbin, glasses, hand, hat, knife, mug, phone, pistol, shoe, telephone cabin and traffic light. Examples of images for each class in the dataset can be seen in Figure 3.9.

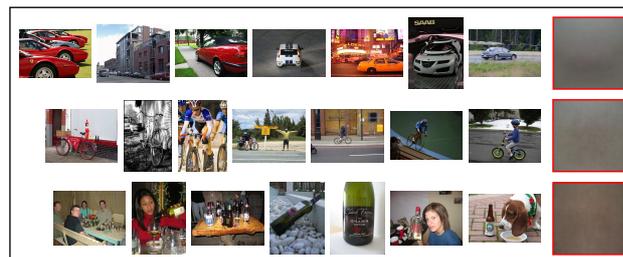
ICARO can be divided into 2 subsets of images: training and test. The training set is further divided into train and validation sets. In ICARO, the user can randomly generate the training and test subsets. Furthermore, the number of training images per class ( $N_{\text{train}}$ ) can be set by the user. For the



(a) Entropy of averages



(b) Caltech-256



(c) PASCAL VOC 2007



(d) ICARO

Figure 3.7: ICARO provides diversified images. Comparison of the entropy of averages for three different objects categories in Caltech-256, PASCAL VOC 2007 and ICARO. For each shown category, the average image is computed using all images within the respective dataset.

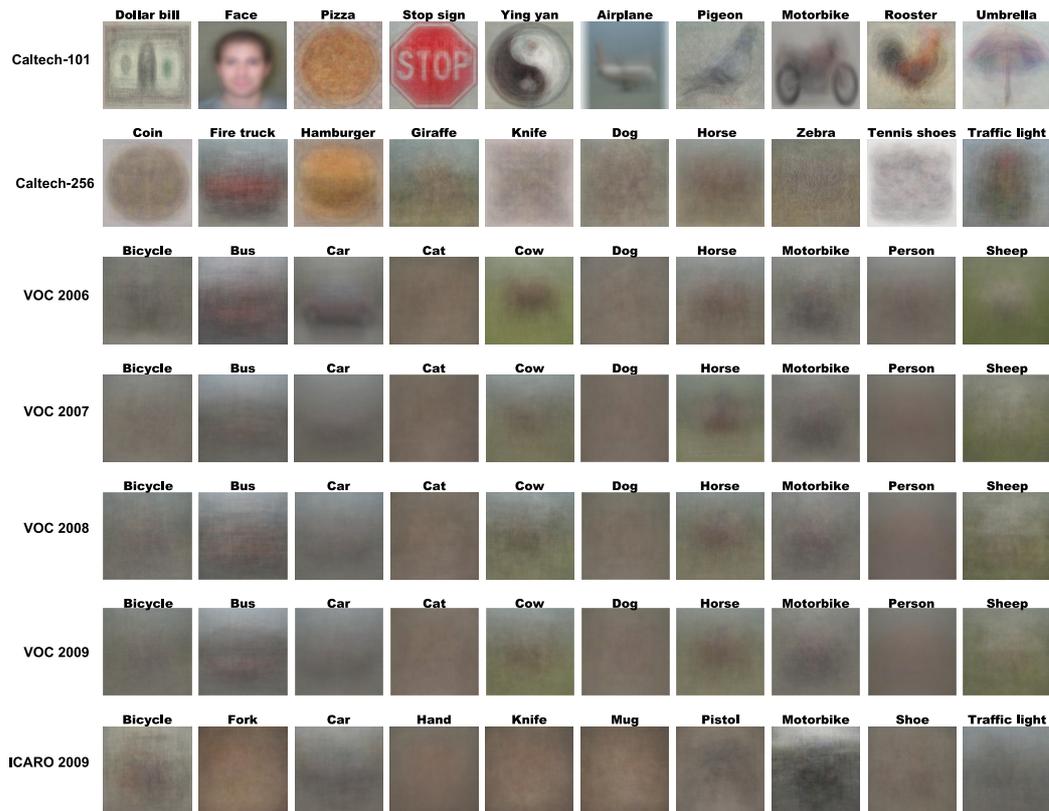


Figure 3.8: Sample average images for different object categories in Caltech-101, PASCAL VOC Challenge 2006-2007-2008-2009, and ICARO datasets.

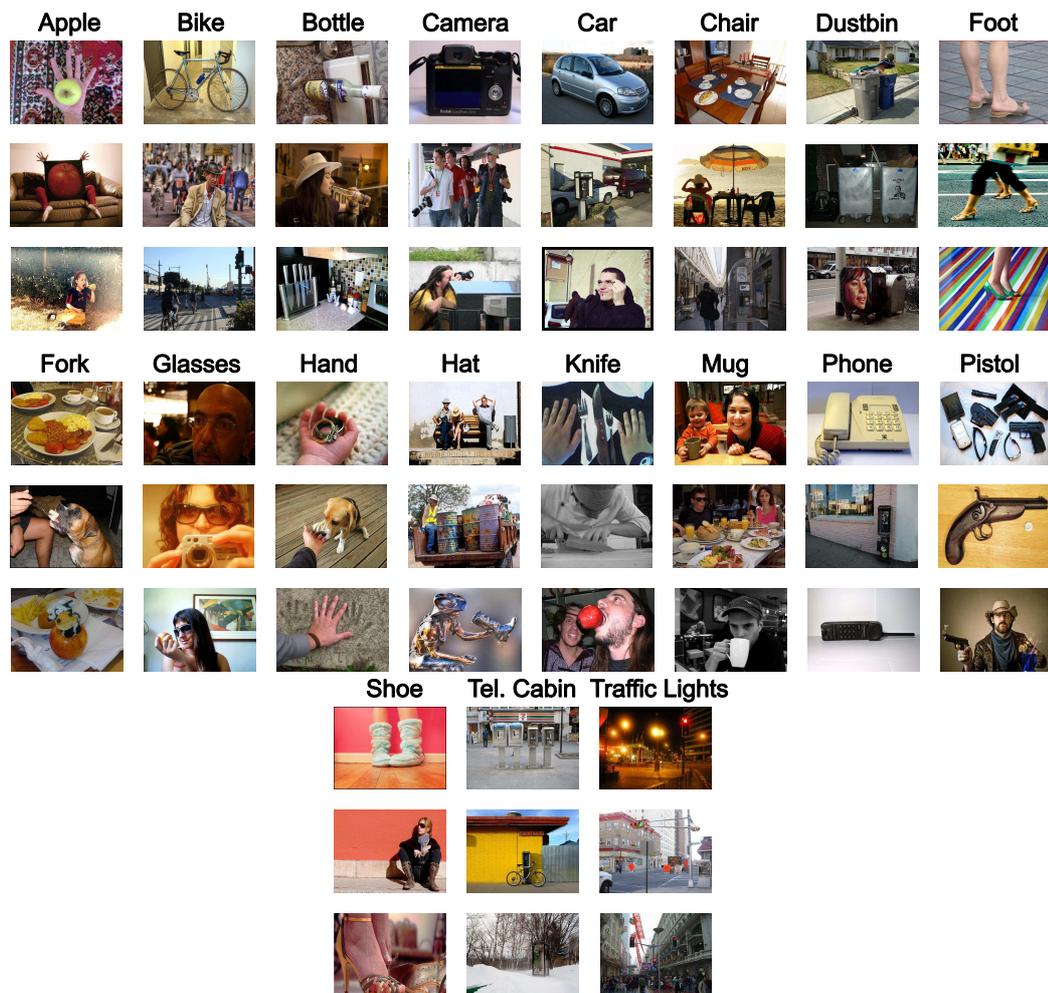


Figure 3.9: Examples of images for each class in ICARO.

image classification benchmark, we chose  $N_{\text{train}} = 10, 20, 30, 40, 50, 60$ . A training set size of  $N_{\text{train}} = 60$  leaves  $N_{\text{test}} \geq 51$  available for testing in all the categories previously selected.

For each of the 19 selected classes, the objective is to train a classifier which predicts the presence of at least one object from the corresponding class in a test image. This prediction must be a real-valued confidence of the object presence for each image. We closely followed the image classification evaluation procedure proposed by the PASCAL VOC Challenge (Everingham et al. 2009a), using the precision/recall curve for each class. Recall is defined as the proportion of all positive examples ranked above a given rank. Precision is the proportion of all examples above that rank which are from a positive class. The interpolated average precision  $AP$  is measured as the mean precision in a set of 11 equally spaced recall levels ( $[0, 0.1, \dots, 1]$ ):

$$AP = \frac{1}{11} \sum_{r \in \{0, 0.1, \dots, 1\}} p_{\text{interp}}(r). \quad (3.1)$$

Varying  $N_{\text{train}}$  the  $AP$  lets us define a benchmark for further comparisons, and it is possible to establish two types of rankings. First, it is possible to compare how a particular method performs for a particular class. The second type of ranking evaluates how a method performs for all classes. When performing experiments over multiple object classes, the average precisions of the individual classes can be aggregated. This aggregation is called mean average precision ( $MAP$ ), which is computed by taking the mean of the average precisions.

### 3.5.1.2 Results

For image representation, we followed the procedure used by Lazebnik et al. (2006) since it performed excellently on datasets similar to ICARO for the image classification problem. Images are represented using local features. To extract these, we experimented with a dense sampling of images patches using a regular grid. Specifically, we used SIFT (Lowe 1999) descriptors of  $16 \times 16$  pixel patches computed over a grid with spacing of 8 pixels. We then performed  $K$ -means clustering of a random subset of 500,000 patches from the training set to build a visual vocabulary. In order to obtain reliable results, we repeated the experimental process 10 times. We experimented with vocabularies of  $K = 200$  and  $K = 400$  words. Finally, each image is represented by a spatial pyramid. Typical pyramid levels ( $L$ ) values for our experiments are  $L = 1, 2, 3, 4$ .

We used SVMs for classification. The decision function of an SVM clas-

sifier for a test sample with feature vector  $\mathbf{x}$  has the form

$$g(\mathbf{x}) = \sum_i \alpha_i y_i k(\mathbf{x}_i, \mathbf{x}) - b, \quad (3.2)$$

where  $y_i$  is the class label of  $\mathbf{x}_i$  ( $-1$  or  $+1$ ),  $\alpha_i$  is the learnt weight of train sample  $\mathbf{x}_i$ ,  $b$  is a learnt threshold parameter and  $k(\mathbf{x}_i, \mathbf{x})$  is the value of a kernel function. We experimented with two kernel functions which have shown good results in object recognition: the Histogram Intersection Kernel (HIK) and the Extended Gaussian Kernel with  $\chi^2$  distance ( $\chi^2$ -Kernel) (Zhang et al. 2007).

The HIK applied to two feature vectors  $\mathbf{x}$  and  $\mathbf{x}'$  of dimension  $D$  is defined as

$$k(\mathbf{x}, \mathbf{x}') = \sum_{i=1}^D \min(\mathbf{x}(i), \mathbf{x}'(i)). \quad (3.3)$$

When the  $\chi^2$ -Kernel is used, we first compute the  $\chi^2$  distance between feature vectors  $\mathbf{x}$  and  $\mathbf{x}'$  as

$$d_{\chi^2}(\mathbf{x}, \mathbf{x}') = \frac{1}{2} \sum_{i=1}^D \frac{(\mathbf{x}(i) - \mathbf{x}'(i))^2}{\mathbf{x}(i) + \mathbf{x}'(i)}. \quad (3.4)$$

Note that we assumed that  $0/0$  is equal to  $0 \iff \mathbf{x}(i) = \mathbf{x}'(i) = 0$ .

The  $\chi^2$ -Kernel is then defined as

$$k(\mathbf{x}, \mathbf{x}') = e^{-\frac{1}{\sigma} d_{\chi^2}(\mathbf{x}, \mathbf{x}')}, \quad (3.5)$$

where  $\sigma$  is a scalar which normalises the distances. It possible to set this  $\sigma$  parameter as the average  $\chi^2$  distance between all elements of the train set, or to tune it by a cross-validation procedure. The latter has not shown the best results in our experiments, so to reduce the training time we opted for the former.

Specifically, we used libSVM (Chang and Lin 2001) and the built-in one-versus-one approach for multi-class classification. A 10-fold cross-validation on the train set to tune SVM parameters was conducted to train each classifier.

We present the results obtained for image classification below. Figure 3.10 shows the top 4 methods by average precision as a function of  $N_{\text{train}}$ . The best results are obtained by using a dictionary of 400 visual words ( $K = 400$ ), 3 pyramid levels ( $L = 3$ ) and the  $\chi^2$ -Kernel. The average precisions with a vocabulary size of 400 are slightly higher than for vocabularies of 200 visual

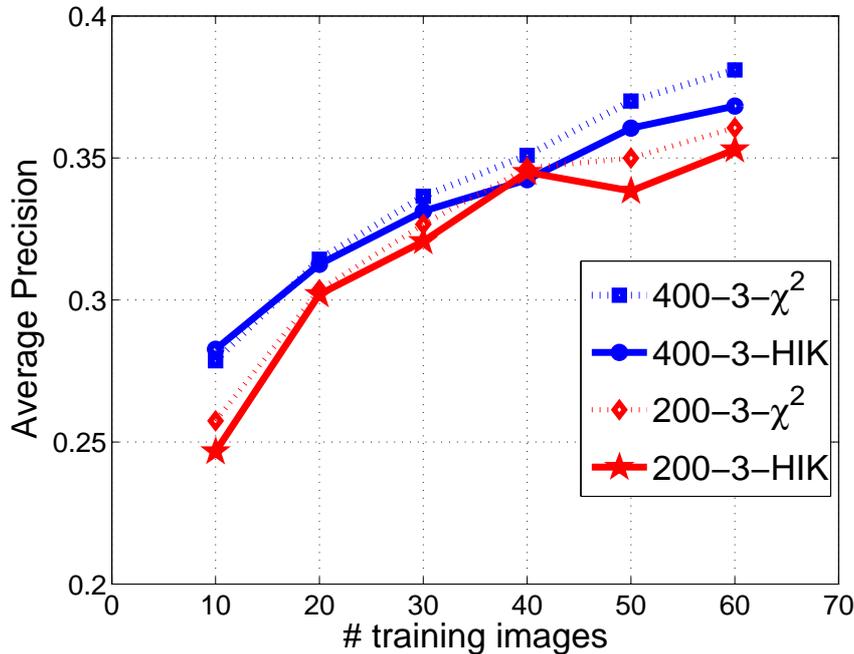


Figure 3.10: Performance of the different methods as a function of  $N_{\text{train}}$ . The algorithm definition corresponds to  $K$ - $L$ -Kernel.

words. The best results are obtained when 3 pyramid levels (i.e.  $L = 3$ ) are used. Moreover, increasing the pyramid levels to  $L = 4$  leads to a small decline in performance. It is also interesting to compare the performance of different kernels. It seems that the  $\chi^2$ -Kernel reports better results when the number of training images (i.e.  $N_{\text{train}}$ ) increases. The precision and recall curves for a representative sample of classes are shown in Figure 3.11.

Figure 3.12 shows ranked images for classes: apple, bike, camera, car, foot, glasses, mug and traffic light. First we show the 3 positive images assigned the highest rank, followed by the 3 positive images assigned the lowest rank, i.e. images that the methods cannot clearly recognise. Finally, we show the 3 negative images assigned the highest rank, i.e. images that confuse the methods.

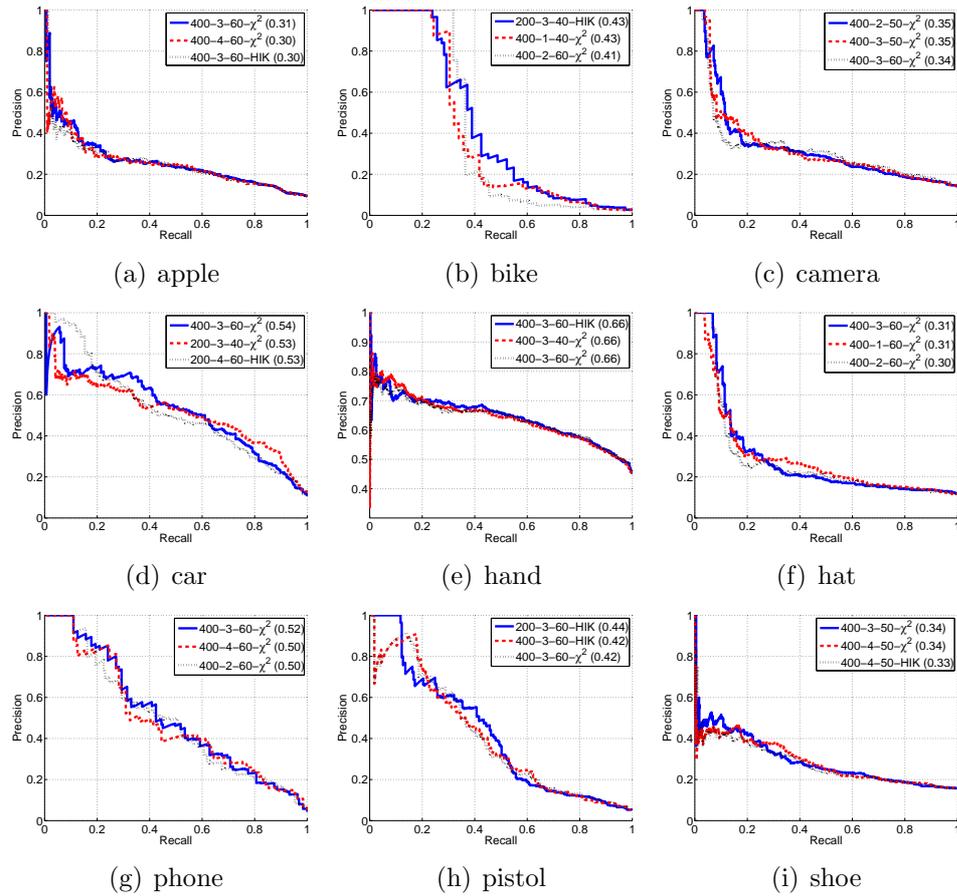


Figure 3.11: Classification results. Precision/recall curves are shown for the first 12 classes. The legend indicates the Average Precision obtained for the corresponding best 3 methods.

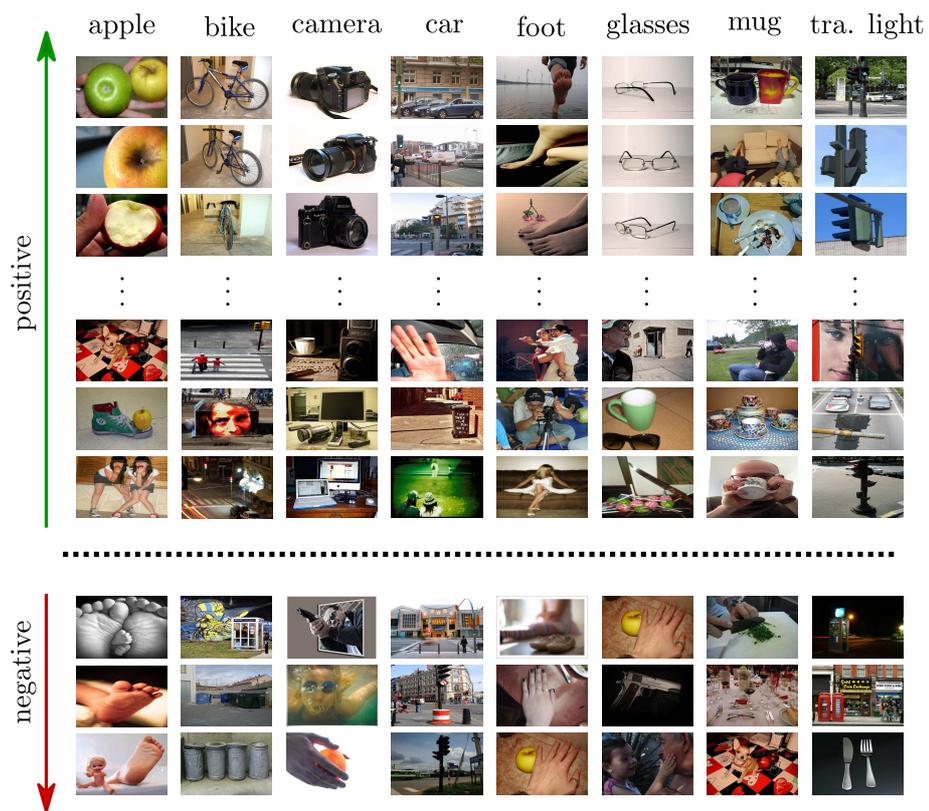


Figure 3.12: Ranked images for classes apple, bike, camera, car, foot, glasses, mug and traffic light.

## 3.5.2 Object Detection

### 3.5.2.1 Benchmark

Object detection involves predicting the bounding boxes where the objects are located in a test image, with an associated real-valued confidence. We propose two possible testing paradigms for the detection benchmark, depending on the training data used, as defined in the PASCAL VOC challenge for detection (Everingham et al. 2009a). First, it is possible to use training images from any source apart from the ICARO images. The aim is to determine the level of success that can be achieved for different algorithms when ICARO is used for testing. Second, there is the testing paradigm where the algorithms can only use the images provided in ICARO.

For detection evaluation, we again closely followed the guidelines of the PASCAL VOC Challenge of the interpolated average precision ( $AP$ ) being the principal quantitative measure used (Equation (3.1)). For the bounding box evaluation, each detection is assigned to ground truth objects and judged to be true/false positive by measuring the bounding box overlap. As defined in (Everingham et al. 2009a), in order to be considered a correct detection, the area of overlap  $a_o$  between the predicted bounding box  $B_p$  and ground truth bounding box  $B_{gt}$  must exceed 50% using the formula:

$$a_o = \frac{\text{area}(B_p \cap B_{gt})}{\text{area}(B_p \cup B_{gt})}. \quad (3.6)$$

### 3.5.2.2 Results

Object detection results were obtained with the object detector described in (Felzenszwalb et al. 2008). This detector has won the PASCAL VOC Challenge since 2007. We use the public implementation<sup>3</sup> of the detector based on mixtures of multiscale deformable part models.

**Training images from any source apart from the ICARO dataset**  
Felzenszwalb et al. (2008) provide models trained on the PASCAL VOC Challenge datasets 2006, 2007 and 2008. We use these models for each of the classes that the PASCAL VOC Challenge datasets and ICARO have in common: bike, bottle, car, chair and motorbike. The detector is run class-by-class for all the images ICARO contains, providing a bounding box plus an associated real-valued confidence of the detection (if any).

---

<sup>3</sup>Available for download: <http://people.cs.uchicago.edu/~pff/latent/>

Dataset	bike	bottle	car	chair	motorbike
<b>VOC 2006</b>	0.620	–	0.635	–	0.579
<b>VOC 2007</b>	0.551	0.265	0.502	0.165	0.383
<b>VOC 2008</b>	0.381	0.278	0.331	0.153	0.366
<b>ICARO</b>	0.440	0.336	0.329	0.146	0.11

Table 3.6: Detection results of the detector in the PASCAL VOC Challenge 2006-2007-2008 and ICARO datasets. For the ICARO dataset we have selected the results obtained with the detector trained on the PASCAL VOC Challenge 2007 dataset, because it presented the best average precision.

MAP	apple	bike	bottle	camera	car	chair	dustbin	foot	fork	glasses	hand	hat	knife	mug	phone	pistol	shoe	t.cabin	t. light
<b>.38</b>	.55	.70	.41	.19	0.77	.008	.48	.23	.10	.31	.009	.23	.15	.57	.25	.74	.02	.61	.85

Table 3.7: Detection results of the detector trained on the ICARO database.

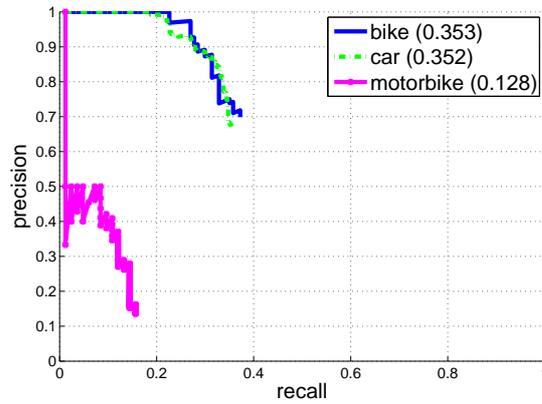
In Figure 3.13, we show the precision/recall curves considering all the annotated objects for the classes of interest. The detector trained on the PASCAL VOC 2007 obtains the best average precision.

Table 3.6 shows a comparison of the performance of the detector on the PASCAL VOC Challenge datasets and ICARO. For the ICARO dataset, we selected the results obtained with the detector trained with the PASCAL VOC 2007 database, because it is the one that presented the best results.

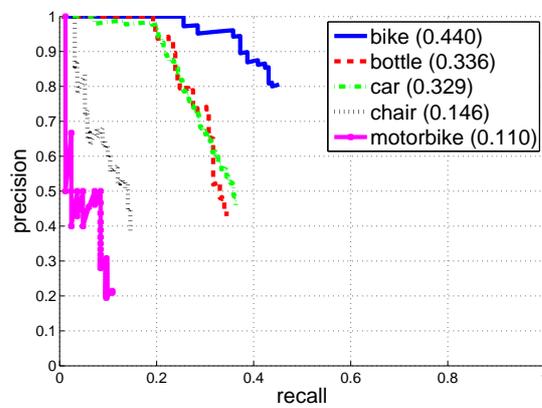
**Training images only from the ICARO database** As in the image classification benchmark, ICARO is divided into 2 subsets of images: training and test. For the experiments, the number of training images per class was set to 100, i.e. ( $N_{\text{train}} = 100$ ). We adapted the publicly available model learning code of Felzenszwalb et al. (2008) for working with ICARO, and trained the object detector for the classes in ICARO with more than 100 images. In specific terms, each model was trained with 2 components and 6 part filters. The results obtained are detailed in Table 3.7.

## 3.6 Conclusion

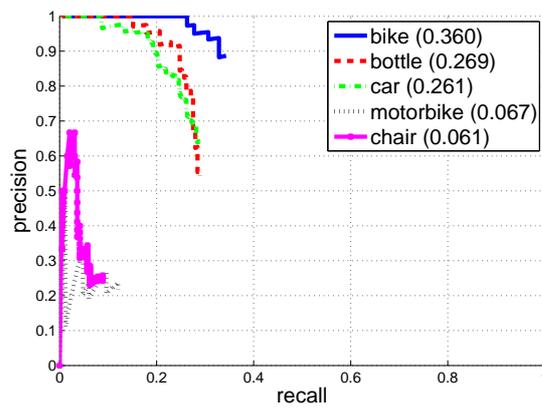
This chapter has presented the variety of datasets used within the context of category-level object recognition. Some of them are used in this thesis for evaluating the new visual vocabularies and categorisation algorithms that



(a) VOC 2006



(b) VOC 2007



(c) VOC 2008

Figure 3.13: Detection results. Precision/recall curves are shown for the classes that PASCAL VOC and ICARO datasets have in common. The legend indicates the Average Precision obtained for the corresponding class.

have been developed. Moreover, a comparative analysis has been carried out.

The main contribution of this chapter is the release of the new database ICARO. We provide a detailed description of ICARO, as well as a comparison with other well-known datasets. Our aim with ICARO is not only to provide a new benchmark for computer vision, but also to offer a challenging dataset that will grow under high quality annotation directives.

Freely available datasets of visual categories have played a key role in the tremendous progress made in both object classification and detection. While advances in machine learning and image feature representation have led to great progress in 2D object recognition/detection approaches, recent work suggests that major gains can be made by modeling objects and their relations in 3D. ICARO provides annotations of 11 viewpoints per bounding box and the especially designed MSs in order to enable the systems to learn suitable representations of the 3D geometry of the object classes that can be exploited for recognition.

Benchmarks for object detection and image categorisation have been established. In the future, we plan to describe a benchmark for recognising and detecting in 3D. Others datasets have been used to that end, but the objects of interest neither appear in different positions within the images, nor with other object classes in real world-scenes.

The ICARO database is publicly available for scientific research purposes. All the images and the annotations can be downloaded from

<http://agamenon.tsc.uah.es/Personales/rlopez/data/icaro/>



# Chapter 4

## Class Representative Visual Words

*“Then,” said he, “the ideas themselves, Socrates, are divisible into parts, and the objects which partake of them would partake of a part, and in each of them there would be not the whole, but only a part of each idea.”*

Plato, *Parmenides* (131c).

---

We present a novel method for constructing a visual vocabulary that takes into account the class labels of images, thus resulting in better recognition performance and more efficient learning. Our method consists of two stages: Cluster Precision Maximisation (CPM) and adaptive refinement. In the first stage, a Reciprocal Nearest Neighbours (RNN) clustering algorithm is guided towards class representative visual words by maximising a new cluster precision criterion. As we are able to avoid expensive cross-validation through optimisation of the vocabulary, the overall training time is significantly reduced without a negative impact on the results. Next, an adaptive threshold refinement scheme is proposed with the aim of increasing vocabulary compactness while at the same time improving the recognition rate and further increasing the representativeness of the visual words for category-level object recognition. This is a correlation clustering based approach, which works as a meta-clustering and optimises the cut-off threshold for each cluster separately. In the experiments we analyse the recognition rate of different vocabularies for a subset of the Caltech-101 dataset, showing how the CPM selects the codebooks, and how the clustering refinement step succeeds in further increasing the recognition rate.

## 4.1 Introduction

A popular strategy for representing images within the context of category-level object recognition is the BoW approach (Sivic and Zisserman 2003, Csurka et al. 2004). The key idea is to vector quantise the high dimensional space of local image descriptors such as SIFT (Lowe 1999), to obtain a codebook of so-called visual words, sometimes also referred to as a visual vocabulary.

In the literature (e.g. (Sivic et al. 2005)), visual words are sometimes justified on the basis of their ability to group semantically meaningful object parts such as wheels or eyes, hence narrowing the semantic gap. In practise though, this only holds for a limited number of visual words, and only if the dataset is sufficiently coherent (e.g. only images of one particular object class). When applied to a more diverse set of images, synonyms and polysemies are, unfortunately, the norm rather than the exception (Quelhas et al. 2007).

Using a fixed feature detector and descriptor scheme, different vocabularies can be obtained for the same data set, and their quality depends on several parameters: the number of visual words, the chosen distance function to measure the similarity between descriptors, and the clustering algorithm. There are various clusterers methods for creating visual codebooks.  $K$ -means or variants thereof, such as approximate  $K$ -means (Philbin et al. 2007) or vocabulary trees (Nister and Stewenius 2006), are currently the most common. Also random trees are used more and more often because their simplicity, speed and performance (Lepetit and Fua 2006).

All of these parameters are normally determined empirically, *i.e.* by applying a grid search over different parameter combinations, training the entire system, and selecting the optimal parameter setting on a validation set. Given that the normally used classifiers (*e.g.* SVM) also have their own set of parameters that require tuning, this results in a computationally heavy process, usually spread over multiple cores or days of processing. To find the codebook size is a complex procedure that should be as effective as possible.

Our aim in this chapter is to study how to adapt the vector quantisation process so as to yield class representative visual words, *i.e.* how to exploit the class labels information during the process of vocabulary construction. This allows us to construct an optimal visual vocabulary without the need for cross-validation in the outer system-loop.

The contribution this chapter makes is twofold. First we propose a novel cluster precision criterion. This evaluates the clusters representativeness. High representativeness is assigned to visual words that generalise well over the within-class variability, yet are discriminative with respect to the object

class (between-class variability). We determine the optimal clustering by maximising this measure in an agglomerative clustering approach. We refer to this procedure as Cluster Precision Maximisation (CPM).

Second, a new adaptive cluster threshold refinement scheme is proposed. It is based on a correlation clustering scheme (Bansal et al. 2004), and its objective is to increase the recognition rate and representativeness of the codebook while at the same time reducing its size. The correlation clustering of an  $n$  vertex weighted graph is the partition of vertices which minimises the sum of positive weights that are cut minus the negative weights that are uncut. The number of clusters is not fixed by the user, but determined as part of the clustering process. The key idea behind our refinement procedure is to exploit the object class label of each feature in a traditional correlation clustering algorithm, *i.e.* we consider two clusters to be similar if their features come from the same object classes. A meta-clustering step such as this might also be useful in other scenarios (as long as cluster similarity is well defined). To the best of our knowledge, this is the first work to describe a correlation clustering approach within this context.

Both contributions pursue the same objective: increase the representativeness of the visual words. The basic assumption behind the whole chapter is that the better this representativeness, the better the classification rate (and this will indeed be validated experimentally as well).

Figure 4.1 shows two examples of visual words. The first (upper row) represents a bad cluster with image patches that clearly come from different object classes and from single object instances, not generalising well within the class. The second example (lower row) on the other hand, shows a class representative visual word found almost exclusively on objects of a single class and including different instances of this class. Our goal is to have more clusters of the latter type than what one normally gets when using *e.g.* standard  $K$ -means clustering.

The rest of the chapter is organised as follows. Section 4.2 first formalises the cluster precision computation problem, then gives a complete description of the RNN clustering algorithm and finally describes the CPM approach. The adaptive threshold refinement using correlation clustering is explained in Section 4.3. Section 4.4 shows the results obtained and Section 4.5 concludes the chapter.

## 4.2 Obtaining Class Representative Visual Words

In this section we present a novel methodology for obtaining more discriminative visual vocabularies. It is called Cluster Precision Maximisation (CPM),



Figure 4.1: First row: image patches that have been clustered together. Clearly all come from different object categories and from single object instances. Second row: cluster with image patches of the same object category and from different instances of the class, *i.e.* it contains class representative visual words.

and results in class representative visual words using a RNN clustering algorithm.

### 4.2.1 Agglomerative Clustering Algorithm

$K$ -means clustering is the most common vector quantisation algorithm used in a BoW approach to category-level object recognition. It is important to point out its main limitations. It does not take into account class-labels. Its output depends on the initialisation as the procedure only undertakes the search for a local optimum and it requires the user to specify the number of clusters in advance. Moreover, there is no guarantee that the clusters obtained are visually compact and it is computationally expensive for big values of  $K$ . Its time complexity is  $O(NKdl)$ , where  $N$  is the number of vectors to quantise,  $K$  is the desired number of clusters,  $d$  is the dimensionality of data and  $l$  is the number of iterations until the process converges.

Other approaches use hierarchical clustering schemes that can be divisive or agglomerative. For the latter, each vector starts in its own cluster, and pairs of clusters are merged as they move up the hierarchy. It is possible to fix a cut-off threshold  $t$  on cluster compactness, which determines the number of clusters by successively merging clusters until the threshold is reached. Both the runtime and the memory requirements are often significantly higher for hierarchical methods. For example, it is known that the standard average-link algorithm requires a  $O(N^2)$  distance matrix to be stored, with  $N$  the number of vectors to be clustered. Fortunately, there are more efficient hierarchical algorithms, making their use in this context feasible. One of them is the RNN algorithm.

### 4.2.1.1 RNN Clustering Algorithm: an overview

The RNN clustering algorithm was first described by de Rham (1980). It is based on the construction of RNN pairs of centroids, *i.e.* pairs of vectors  $\mathbf{x}$  and  $\mathbf{y}$ , so that  $\mathbf{x}$  is the nearest neighbour to  $\mathbf{y}$ , and vice versa. As soon as an RNN pair is found it can be agglomerated. Benzécri (1982) developed an efficient implementation that ensures that RNN can be found with as little re-computation as possible. This is achieved by building a Nearest Neighbours (NNs) chain, which consists of an arbitrary vector, followed by its nearest neighbour, which is again followed by its nearest neighbour among the remaining points and so on. Each of these chains finishes with an RNN pair.

The algorithm starts with an arbitrary centroid. Then an NNs chain is built. When an RNN pair is found, *i.e.* no more centroids can be added to the current chain, the corresponding clusters are merged if their similarity is above a fixed cut-off threshold otherwise the algorithm discards the chain. When the current chain is empty, a new arbitrary point is selected, and a new chain is started.

Benzécri (1982) demonstrates that the run-time is bound by the time required to find nearest neighbours. The key point is how to recompute the similarity between a new cluster (after merging an RNN pair) and the others. Leibe et al. (2008b) show this can be done (efficiently) if the cluster similarity can be expressed in terms of centroids, which holds for a group average criterion based on correlation or Euclidean distances. For a more detailed description of the implementation described in Leibe et al. (2008b)<sup>1</sup> we refer to Section 5.2.

The implementation described in Leibe et al. (2008b) has  $O(N^2d)$  time and  $O(N)$  space complexity. Once a hierarchical agglomerative and efficient clustering algorithm has been identified, the next step is to develop the CPM approach. Algorithm (4.1) describes both the RNN clustering algorithm and its integration in a CPM approach.

## 4.2.2 The Cluster Precision Maximisation

Whether a particular clustering (or codebook) is better than the baseline or not can be evaluated within the context of the entire system, *i.e.* by evaluating its effect on the accuracy of the resulting classifier on a validation set. Let us suppose we are given a set of local features extracted from images in a database of object classes. We could fix the number of clusters we want

---

<sup>1</sup>Our own implementation of the RNN clustering algorithm can be downloaded from <http://agamenon.tsc.uah.es/Personales/rlopez/docs/rnn.tar.gz>

---

**Algorithm 4.1** RNN clustering algorithm integrated in a CPM approach
 

---

```

 $P_{max} = 0$ 
for  $thres = min$  to  $max$  do
   $C = \emptyset$ ;  $last \leftarrow 0$ ;  $lastsim[0] \leftarrow 0$ ; //C will contain a clusters list
   $L[last] \leftarrow v \in V$ ; //Start chain L with a random vector v
   $R \leftarrow V \setminus v$ ; //All remaining points are kept in R
  while  $R \neq \emptyset$  do
     $(s, sim) \leftarrow getNearestNeighbor(L[last], R)$ ;
    if  $sim > lastsim[last]$  then
       $last \leftarrow last + 1$ ;  $L[last] \leftarrow s$ ;  $R \leftarrow R \setminus \{s\}$ 
    else
      if  $lastsim[last] > thres$  then
         $s \leftarrow agglomerate(L[last], L[last - 1])$ ;  $R \leftarrow R \cup \{s\}$ ;  $last \leftarrow last - 2$ 
      else
         $C \leftarrow C \cup L$ ;  $last \leftarrow -1$ ;  $L = \emptyset$ ;
      end if
    end if
    if  $last < 0$  then
       $last \leftarrow last + 1$ ;  $L[last] \leftarrow v \in R$ ;  $R \leftarrow R \setminus s$ 
    end if
  end while
   $P \leftarrow getP(C)$ ; //Evaluate P
  if  $P > P_{max}$  then
     $P_{max} \leftarrow P$ ;  $C_{optimum} \leftarrow C$ ;
  end if
end for

```

---

for the codebook by using, for example,  $K$ -means, and apply the clustering algorithm of our choice to obtain a visual vocabulary for the object classes. One strategy to follow is to empirically find the optimal cluster parameters by testing whether this codebook reports an accurate categorisation on a validation set. If it does not, we run the algorithm again with different parameters, until we obtain a codebook casting the lowest empirical risk in categorisation.

The main problem with this approach is clear: to validate just the clustering it is necessary to go through the system's entire pipeline. Moreover, this validation usually in turn involves cross-validation for tuning the parameters of the classifier. The combinatorics involved results in an explosion of the number of experiments needed; and the result also becomes dependent on the classifier chosen.

Our aim is to directly build a codebook whose clusters have a high precision and representativeness. The objective is to obtain a codebook that allows the best recognition rates, but to train the classifier over and over again. The CPM method searches directly for class representative visual words, *i.e.* representative clusters. The key idea behind this approach is that, the more images with different object instances of a particular class contain a particular visual word, the more representative the word is for that class and the better the categorisation of a codebook that includes it.

Closer to our approach are the works of Mikolajczyk et al. (2005) and Stark and Schiele (2007). In (Mikolajczyk et al. 2005) the performances of local detectors and descriptors are compared within the context of the object class recognition problem, and a new evaluation criterion based on the clusters' precision is proposed. The problem is that following this approach, many clusters with features from only one object instance get high precision scores. Stark and Schiele (2007) decided to give higher scores to feature descriptors that generalise across multiple instances of an object class, and proposed a new cluster precision definition, but their approach obtains the best score when each cluster contains only one vector.

Let us assume a database  $\mathcal{DB}$  contains  $N$  images of  $M$  different object classes,  $\mathcal{DB} = \{\mathcal{I}_1, \mathcal{I}_2, \dots, \mathcal{I}_N\}$ . For each image in the database we first extract local features  $f$ , so an image  $\mathcal{I}_i$  can be represented with a set of features  $\mathcal{I}_i = \{f_{i_1}, f_{i_2}, \dots\}$ . Note that this will not be a BoW representation until the vector quantisation is done. After the clustering, a codebook  $\mathcal{W} = \{w_1, w_2, \dots, w_K\}$  with  $K$  words is obtained, and thus the images can be represented with it. See Figure 2.1 for a graphical overview of a traditional BoW approach.

Mikolajczyk et al. (2005) evaluate the codebook  $\mathcal{W}$  by computing the average cluster precision for all the object classes. Suppose there are  $K$

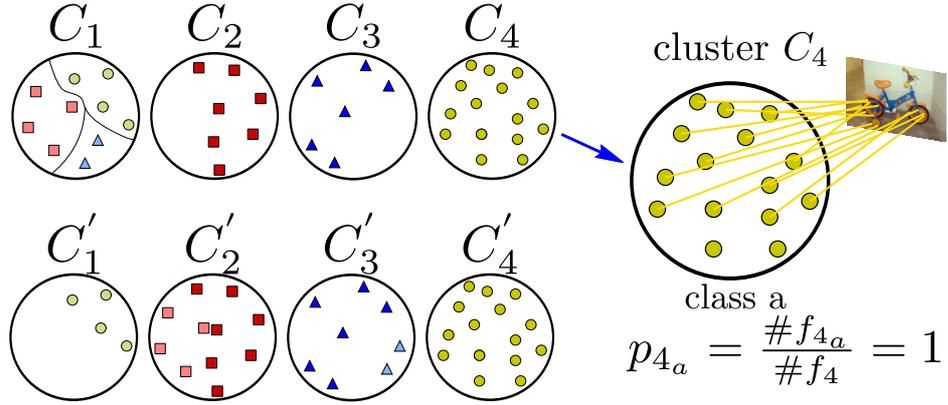


Figure 4.2: Left– Following equation (4.1) clustering  $\{C_1, C_2, C_3, C_4\}$  obtains cluster precisions of 1 for all classes, that is the same score as for a nearly perfect clustering as  $\{C'_1, C'_2, C'_3, C'_4\}$ . Right – Can the equation (4.1) determine whether or not cluster  $C_4$  is representative? As it is shown, cluster  $C_4$  would get a high score for  $P_m$ , but it is not a representative cluster at all. It brings together a bunch of features, from class  $a$ , but only one image is represented by it.

clusters in the vocabulary, but only  $K_m$  in which class  $m$  dominates, *i.e.* clusters where there are more vectors belonging to class  $m$  than any other. Mikolajczyk et al. (2005) define the average precision for a given object class  $m$  as

$$P_m = \frac{1}{K_m} \sum_{j=1}^{K_m} p_{jm} , \quad (4.1)$$

where  $p_{jm}$  is the number of features of class  $m$  in cluster  $j$ , *i.e.*  $\#f_{jm}$ , divided by the total number of features in cluster  $j$ , *i.e.*  $\#f_j$ .

Equation (4.1) was used in (Mikolajczyk et al. 2005) for comparing the performance of several local detectors and descriptors within the context of object recognition. The main problem with equation (4.1) is that clusters for which none of the classes dominate do not have any impact on the score at all. For instance, in Figure 4.2 (left), the clustering  $\{C_1, C_2, C_3, C_4\}$  obtains cluster precisions of 1 for all classes, that is the same score as for a nearly perfect clustering as  $\{C'_1, C'_2, C'_3, C'_4\}$ . Moreover, equation (4.1) can obtain high scores for those clusters that contain features from only a single object instance, as shown in Figure 4.2 (right). Stark and Schiele (2007) discount such clusters by summing over the fraction of *objects* of a class  $m$  in cluster  $j$  ( $p'_{jm}$ ) instead of individual features, and weight these fractions by cluster

sizes, obtaining a new expression,

$$P_m = \left( \sum_{j=1}^{K'_m} s_j \right)^{-1} \sum_{j=1}^{K'_m} s_j p'_{j_m} , \quad (4.2)$$

where  $j$  now ranges over all  $K'_m$  clusters in which objects of class  $m$  dominate and  $s_j$  is the total number of features in cluster  $j$ . This new cluster precision definition gives higher scores to clusters that generalise across multiple instances of an object class, but it casts the maximum score when each cluster contains only one vector again.

Because neither of the two cluster precision definitions seem to meet our goal of selecting class representative visual words without artefacts, we propose a new cluster precision, this time summing over the number of features of class  $m$  times the number of objects of class  $m$  in each cluster,

$$P_m = \frac{K}{M} \sum_{j=1}^K s_{j_m} n_{j_m} , \quad (4.3)$$

where  $s_{j_m}$  is the number of features found in images of object class  $m$  in cluster  $j$ ,  $n_{j_m}$  is the number of different objects (*i.e.* images) of class  $m$  represented in cluster  $j$ ,  $K$  is the number of clusters and  $M$  is the number of classes. This is illustrated in Figure 4.3. This new cluster precision definition varies from  $S \times S_m/M$  (*i.e.* there is a cluster per feature), to  $S_m \times N_m/M$  (*i.e.* one cluster brings together all the features),  $S_m$  being the number of features extracted from images of the object class  $m$ ,  $S$  the total number of extracted features and  $N_m$  the number of different images of class  $m$  in the database.

Equation (4.3) is the core of the CPM approach. With this new formulation, it is possible to evaluate the cluster's precision and representativeness after each iteration  $i$ . Let  $RNN(t_i)$  be the execution of the RNN clustering algorithm with threshold  $t_i$  in the iteration  $i$ , then

$$P^{(i)} = \sum_{m=1}^M P_m^{(i)} , \quad (4.4)$$

where superscript  $i$  indicates the iteration number. The aim of CPM is finding the value of the threshold  $t_i$  that maximises (4.4)

$$t_{opt} = \operatorname{argmax}_{t_i \in [t_{min}; t_{max}]} P^{(i)} . \quad (4.5)$$

These threshold limits  $t_{min}$  and  $t_{max}$  must be chosen depending on the dis-

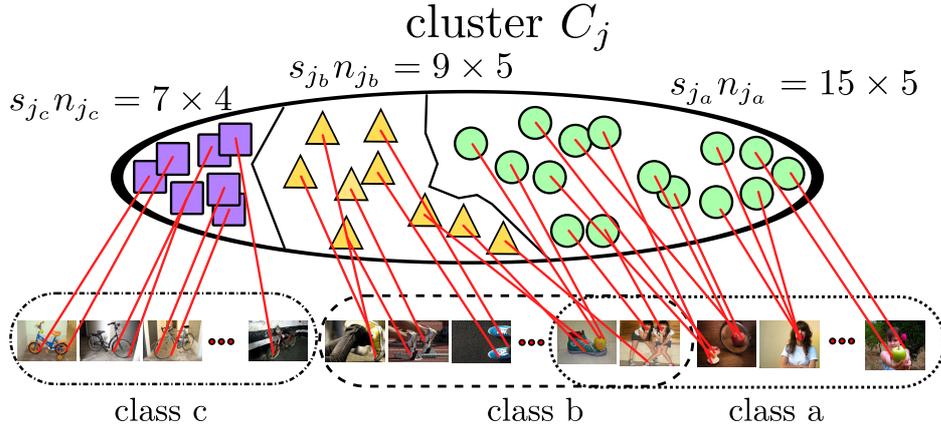


Figure 4.3: Cluster  $C_j$  contains features from three different classes  $\{a, b, c\}$ . Equation (4.3) takes into account the number of objects instances represented by the cluster  $C_j$ , i.e. the number of different images from which the local features are extracted.

tance function used and the normalisation carried out with the data. Figure 4.4 shows how  $P^{(i)}$  evolves while the threshold varies in each iteration. For the first iteration, e.g.  $t_0 = 0$ , RNN casts only singletons and  $P^{(0)} = S^2/M$ . As the threshold increases,  $P^{(i)}$  should also grow until the maximum is reached ( $t = t_{opt}$ ). Then it quickly drops until RNN casts only one cluster bringing together all vectors ( $t_{max}$ ) and

$$P^{(max)} = \sum_{m=1}^M \frac{S_m \times N_m}{M}. \quad (4.6)$$

Normally  $S_m \gg N_m$ , so  $P^{(0)} \gg P^{(max)}$ . The experiments must confirm what is shown in Figure 4.4, i.e. whether the CPM approach is able to find a maximum for (4.4). In Appendix A we theoretically analyse the evolution of  $P^{(i)}$  through iterations in the CPM approach.

### 4.3 Correlation Clustering Based Refinement Step

In this section, an adaptive threshold refinement step for compacting the codebook while further increasing the average recognition rate in categorisation is described.

Just like the  $K$ -means algorithm, the agglomerative clustering algorithms in general, and the RNN clustering algorithm in particular, have several

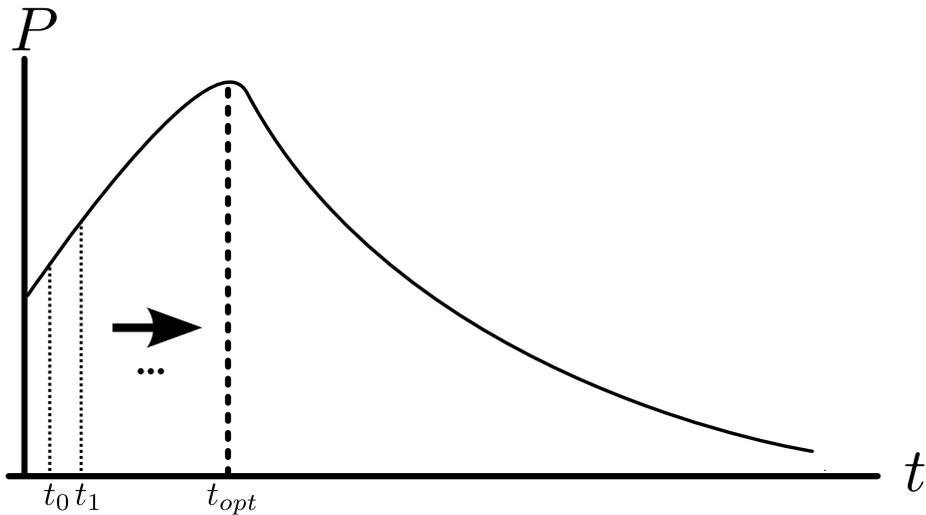


Figure 4.4:  $P(t)$  evolution in a CPM execution.

known deficiencies. The RNN algorithm overcomes those related to both the time and space complexity which are often significantly higher for agglomerative methods. It is clear that it is not necessary to fix the number of clusters beforehand, but a cut-off threshold must be provided instead, which divides the high dimensional vector space into clusters with a compactness that is always below the threshold. Because this threshold is the same for all clusters, it may occur that some real clusters are still split into several clusters.

Moreover, in our experiments, we found that after an RNN clustering algorithm execution, *e.g.* with the threshold obtained by the CPM approach, there is often lots of singletons, *i.e.* clusters with only one vector. This can be explained by the fact that during the process chains of NNs are sometimes discarded. This increases the size of the codebook and has a negative impact on the computation time during the on-line classification.

One possible solution is to consider these points as outliers, so they are not used to define any cluster centre. They can then simply be assigned to the nearest cluster centre after the clustering is finished. But following this approach we neither guarantee the resulting clusters are class representative, nor increase the discriminativity of the codebook.

In this dissertation, we study whether it is not possible to follow a different policy for decreasing the number of clusters while pursuing this threefold objective:

- I. to integrate the smaller clusters in bigger clusters to obtain more

representative visual words.

- II. to obtain compact codebooks.
- III. to increase the recognition rate of the overall system.

In this section we describe a new correlation clustering based refinement step so as to obtain an adaptive threshold scheme where the clusters are merged in accordance with the qualitative information we have. This approach is developed in Section 4.3.2, but first, Section 4.3.1 introduces the correlation clustering techniques.

### 4.3.1 Correlation Clustering

There is extensive literature on the problem of correlation clustering, first defined by Bansal et al. (2004). They consider the problem of having a complete graph of  $n$  vertices, where each edge  $(u, v)$  is labelled either  $+1$  or  $-1$  depending on whether  $u$  and  $v$  have been deemed to be similar or different. The objective is to partition the nodes in order to minimise the number of positive edges that are cut, and the number of negative edges that are uncut. Cutting an edge, means that the nodes are not merged. An example of an undirected graph with 5 vertexes is shown in Figure 4.5. In the example, the possible weights are  $+1$  or  $-1$ . The correlation clustering is depicted on the right weight matrix in Figure 4.5. The correlation clustering contains two clusters, one bringing together vertexes  $\{2, 3\}$  and the other  $\{1, 4, 5\}$ .

The best known approximation algorithm for this problem is by Charikar et al. (2003) who give an LP-based algorithm that achieves an approximation factor of 4. When the edge weights are arbitrary, the problem is equivalent to the multicut problem shown in Demaine et al. (2006), and there is a  $O(\log n)$ -approximation bound. The approach implemented in this chapter is based on the correlation clustering technique described by Gionis et al. (2005), where a graph with edge weights  $W_{uv}$  satisfying the triangle inequality must be given. These weights represent how dissimilar the extremes of each edge are.

Spectral clustering methods also build a graph, and then try to find the optimal cut (*e.g.* normalised cuts (Ng et al. 2001)). They rely on the eigen-decomposition of a modified similarity matrix to project data prior to clustering. Choosing a good similarity graph is not trivial, and spectral clustering techniques can be quite unstable under different choices of the parameters for the neighbourhood graphs. Correlation clustering techniques use a fully-connected graph. Moreover, the number of clusters is determined as a part of

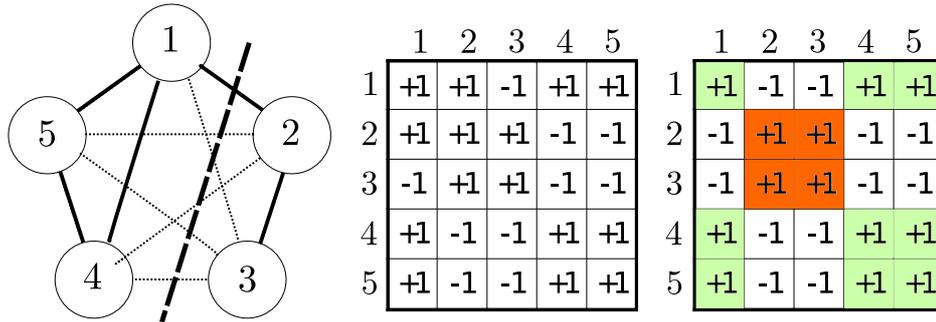


Figure 4.5: Example of correlation clustering. Solid edges indicate weight  $+1$ , and dashed edges indicate weight  $-1$ . Observe the algorithm cuts edge  $(1, 2)$  with weight  $+1$ , while it does not fail to cut edges with weight  $-1$ . The correlation clustering is depicted on the right weight matrix. The correlation clustering contains two clusters, one containing vertexes  $\{2, 3\}$  and the other  $\{1, 4, 5\}$

the optimisation process, *i.e.* one does not need to fix the number of clusters as a separate parameter.

### 4.3.2 Refining Visual Words

The idea for refining the visual words is simple. It is illustrated in Figure 4.6. Once the clustering has been done, *e.g.* following the CPM approach with an RNN clustering algorithm, it is possible to construct a complete graph with  $K$  vertices, one per cluster, singletons included, where each edge  $(C_u, C_v)$  is labelled either 0 or 1 depending on whether the clusters  $C_u$  and  $C_v$  have been deemed to be similar or different, respectively.

Once such a graph is built the goal is to find a partition of the vertices of the graph that minimises the sum of 0 weight edges that are cut minus the 1 weight edges that are uncut. This is achieved by the correlation clustering algorithm proposed by Bansal et al. (2004).

In our approach, how similar two clusters are depends on: the distance between the cluster centres, their sizes, and on the class labels of the features they bring together. One of the objectives is to reduce the number of small clusters, but not to merge clusters that have nothing in common. That is why the class labels are used. The label of each feature in a cluster is known, *i.e.* which class it belongs to. Only those clusters with classes in common may be merged by this refinement step.

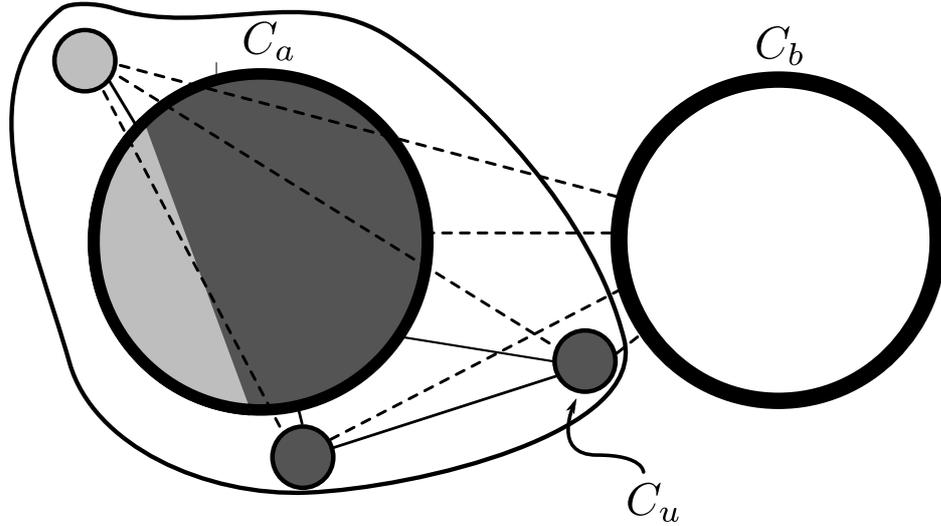


Figure 4.6: Correlation clustering based refinement. Solid edges indicate weight 0, and dashed edges indicate weight 1. Observe singleton  $C_u$ , which is clustered with the one of the same class  $C_a$ , even though it is nearer to  $C_b$  but they do not share features with the same class label.

#### 4.3.2.1 Adaptive Threshold Merging Algorithm

The adaptive threshold merging algorithm first builds a complete graph  $G_0$ . Let  $C = \{C_1, C_2, \dots, C_k\}$  be the complete set of clusters.  $S$  is the set of *small* clusters in  $C$ , and the rest of clusters belong to  $\bar{S}$ , so  $C = S \cup \bar{S}$ . We consider that a cluster is *small* if the number of vectors it contains is below a fixed threshold. All the edges in  $G_0$  joining two clusters in  $\bar{S}$  are labelled with 1, *i.e.* clusters that are not small will not be merged. Furthermore, all the edges linking clusters that have no features with the same class labels are labelled with 1 (recall Figure 4.6). The remaining edges of  $G_0$  are labelled with the distance between centroids. For the experiments we report results using the Euclidean distance between centroids.

Now the adaptive threshold merging of clusters begins. In each iteration the algorithm increases a threshold  $t_i$ . Let  $t_{min}$  and  $t_{max}$  be the minimum and maximum threshold respectively, and  $t_i$  the threshold to be used in the iteration  $i$ . At the beginning of each iteration  $G_i = G_0$ . Edges in  $G_i$  with distance over  $t_i$  are automatically labelled with 1. Suppose  $N$  is the number of nodes in  $G_i$  that might be merged, *i.e.* vertices whose common edges have a distance  $d \leq t_i$ . These edges are labelled with 0 in  $G_i$ , which becomes a binary graph. Then the correlation clustering approach is run over the graph, and a new codebook for each iteration is obtained. Until  $t_{max}$  is reached the

procedure is repeated. For every new codebook a classifier is trained and the average class recognition rate is measured. At the end of this refinement process the codebook allowing the best recognition results is selected.

As described earlier, in each iteration a correlation clustering algorithm is run. This correlation clustering works as a meta-clustering refinement merging clusters, reducing the codebook size and increasing the recognition rate. It is possible to apply any correlation clustering algorithm described in (Bansal et al. 2004). However, we propose a correlation clustering following the formulation detailed in (Gionis et al. 2007). Our algorithm takes a complete graph  $G_i$  as input. The weight of the edge linking clusters  $C_u$  and  $C_v$  is called  $W_{uv}$ . An edge with a weight of 0 or 1 indicates that both  $u$  and  $v$  should be merged or separated, respectively. The algorithm goes through the nodes in  $G_i$ , *i.e.* the centroids, and it considers merging them with a different centroid or allowing them to continue being alone. The node is merged with the cluster that yields the minimum cost. The process iterates until there is no move that can improve the cost or a maximum number of iterations is exceeded. The cost function used in this approach is similar to the one proposed in (Gionis et al. 2007) for correlation clustering. Given a complete graph  $G_i$ , each edge  $(C_u, C_v)$  has weight  $W_{uv} \in \{0, 1\}$ . The cost  $c(C_u, C'_i)$  of assigning node  $C_u$  to a new cluster  $C'_i$  is computed as follows

$$c(C_u, C'_i) = \sum_{C_v \in C'_i} W_{uv} + \sum_{C_v \in \overline{C'_i}} (1 - W_{uv}) . \quad (4.7)$$

The first term is the cost of merging  $C_u$  in  $C'_i$ , while the second is the cost of not merging node  $C_u$  with nodes not in  $C'_i$ . The experiments show that the algorithm is quite effective, improving the recognition rate from a clustering partition obtained with an RNN algorithm while the size of the codebook is dramatically reduced.

This refinement procedure can be run iteratively. For instance, we first run the refinement for reducing the singletons. With the resulting codebook, we run again the correlation clustering approach for reducing clusters up to size 2, and so on.

## 4.4 Experiments

In the experiments we show how the CPM succeeds in finding the threshold which obtains the maximum for the cluster precision. It is shown that the average classification rate drastically drops with codebooks obtained with thresholds over this bound; furthermore the best results in classification are

Dataset	#Training/Test Images	#Training/Test Features
Caltech-101 subset	446/444	55679/56590
<b>Total</b>	890	112269

Table 4.1: Outline of the number of training and test images and features obtained from the database for the experiments performed in this work.

obtained by codebooks with high values of Cluster Precision (CP). We also show how the correlation clustering refinement algorithm improves the average recognition rate while reducing the size of the codebooks.

#### 4.4.1 Experimental Setup

We use a subset of the Caltech-101 database (Fei-Fei et al. 2004). This dataset contains 101 object categories with 40 to 800 images per class. Most of the images in the database have little or no clutter. Furthermore, the objects tend to lie in the centre of the images and appear in similar poses. Some images have a partially black background due to artificial image rotations. For more details we refer to Chapter 3. For our experiments we have randomly chosen the following 10 categories: ewer, sunflower, kangaroo, starfish, trilobite, menorah, helicopter, butterfly, brain and grand piano. In the experiments we randomly select 50% of the images for training the system. Some of the images are shown in Figure 4.7. More details are summarised in Table 4.1.

We show results using a traditional BoW approach, which is based on regions of interest and not on dense sampling, although the latter has been shown to outperform interest point detector based methods in image classification (Perronnin et al. 2006). The objective of this chapter is to demonstrate that the CPM procedure is able to identify the thresholds for a RNN agglomerative clustering that cast the best results in recognition of object classes. But once the codebook is obtained, any BoW approach could be used.

There are many different techniques for detecting and describing local image regions (Tuytelaars and Mikolajczyk 2008, Mikolajczyk and Schmid 2005). Here we briefly describe the detectors and descriptors used in the experiments <sup>2</sup>. The region detector used is the Hessian-Laplace (Mikolajczyk and Schmid 2005) that responds to blob-like structures. It searches for local maxima of the Hessian determinant, and selects a characteristic scale via the Laplacian. We experiment with SIFT (Lowe 1999) as descriptor, which is

<sup>2</sup>The binaries have been taken from <http://www.robots.ox.ac.uk/~vgg/research/affine/>

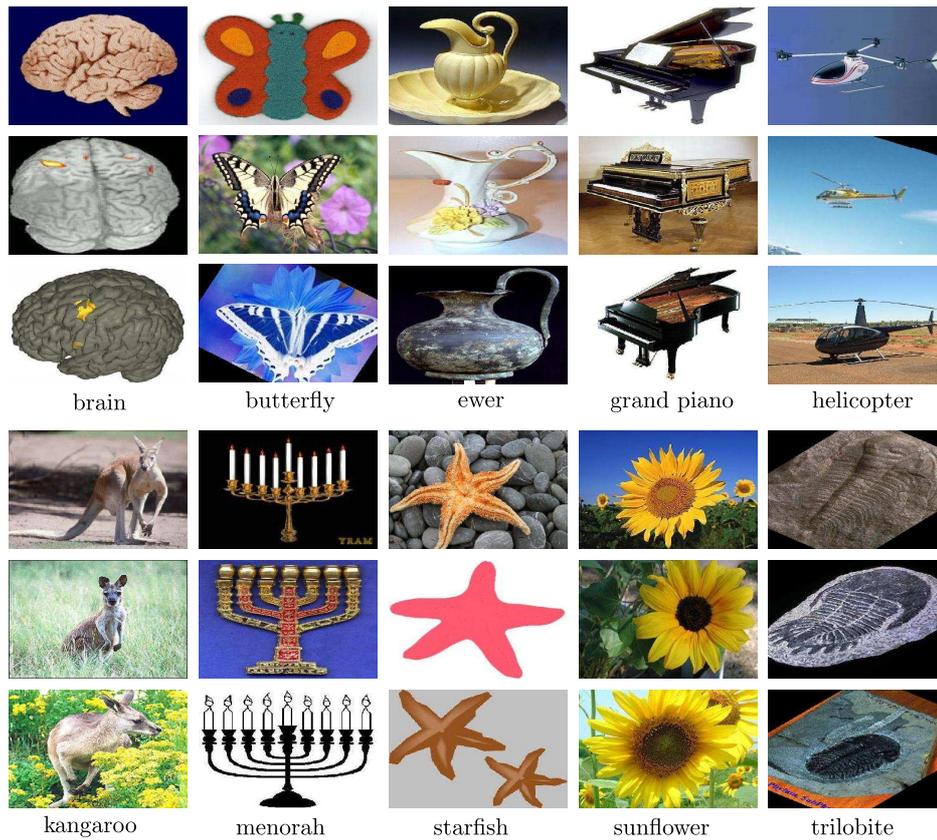


Figure 4.7: Images examples from the Caltech-101 subset used in the experiments. Note that some images have a partially black background due to artificial image rotations (*e.g.* corners of the third butterfly image).

a 3D histogram over local gradient locations and orientations, weighted by gradients magnitude.

For classification we use SVM with the kernels most widely used within the context of category-level object recognition in images: HIK (Maji et al. 2008) and the  $\chi^2$ -Kernel (Zhang et al. 2007). We also present, for comparison, results obtained with general Radial Basis Functions (RBFs) using Euclidean distances between the histograms.

Specifically, we use libSVM (Chang and Lin 2001) and the built in one-versus-one approach for multi-class classification. A 10-fold cross-validation on the train set to tune SVM parameters is conducted. When the RBF kernel is used two parameters need to be tuned:  $C$  and  $\sigma$ .

The HIK applied to two feature vectors (histograms of visual words)  $H_x$  and  $H_y$  of dimension  $D$  is defined as

$$k(H_x, H_y) = \sum_{i=1}^D \min(H_x(i), H_y(i)) . \quad (4.8)$$

For this particular case, only the  $C$  parameter needs to be tuned. When the  $\chi^2$ -Kernel is used, we first compute the  $\chi^2$  distance between feature vectors  $H_x$  and  $H_y$  as

$$d_{\chi^2}(H_x, H_y) = \frac{1}{2} \sum_{i=1}^D \frac{(H_x(i) - H_y(i))^2}{H_x(i) + H_y(i)} . \quad (4.9)$$

It is assumed that  $0/0$  is equal  $0 \iff H_x(i) = H_y(i) = 0$ . The kernel function based on the  $\chi^2$  distance is then defined as

$$k(H_x, H_y) = e^{-\frac{1}{\sigma} d_{\chi^2}(H_x, H_y)} , \quad (4.10)$$

where  $\sigma$  is a scalar which normalises the distances. It is possible to fix this  $\sigma$  parameter to the mean of all distances, or to tune it through the cross-validation approach. In our experiments the latter has not shown better results, so to reduce training time we opted for the former.

#### 4.4.2 CPM performance with RNN clustering

In these initial experiments the objective is to confirm that the CPM approach is able to find the threshold for an RNN clustering that will obtain a maximum for the CP. Figure 4.8(a) confirms what was proved in section 4.2.2: the CP obtains one maximum for a specific threshold ( $t_{opt} = 0.25$ ) for the subset of Caltech-101 database, and then it quickly drops to suboptimal values. Moreover, Figure 4.8(a) shows a comparison of our cluster precision

formulation in Equation (4.3) with the formulations in Equations (4.1) and (4.2), proposed by Mikolajczyk et al. (2005) and Stark and Schiele (2007) respectively. It is clear that while our cluster precision obtains a maximum for a determined threshold, the other formulations are monotonically decreasing functions.

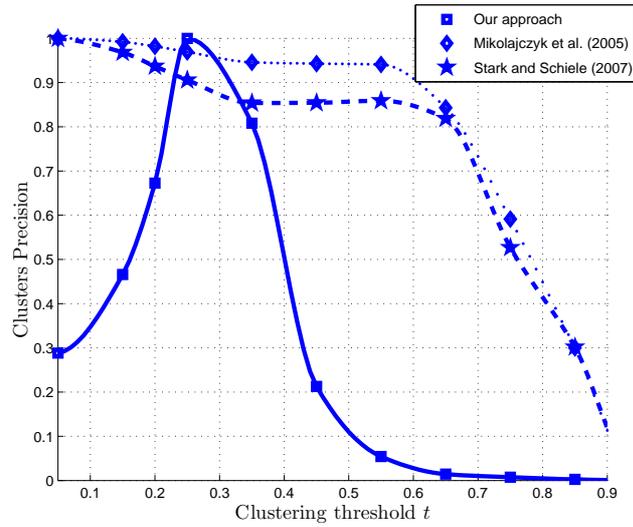
As expected, codebooks obtained with thresholds over the bound  $t_{opt}$  defined by the maximum of CP allow poor recognition rates for the three different types of kernels. We trained and tested the classifier, for each kernel type, with the different codebooks obtained for all the thresholds the CPM used in its iterations. For each codebook, the average recognition rate the classifier obtains on the test images is shown in Figure 4.8(b). For the RBF kernel, the highest CP coincides with the best classification rate. Both the HIK and the  $\chi^2$ -Kernel obtain the best results for thresholds near  $t_{opt}$ . It can also be seen that for thresholds over this bound the classification rate drastically drops. So the CPM is able to identify for a RNN the thresholds that are going to allow the best recognition rates. As we are able to avoid expensive cross-validation through the optimisation of vocabulary, the CPM method significantly reduces the overall training time without a negative impact on the results.

#### 4.4.3 Increasing the recognition rate with the correlation clustering based refinement

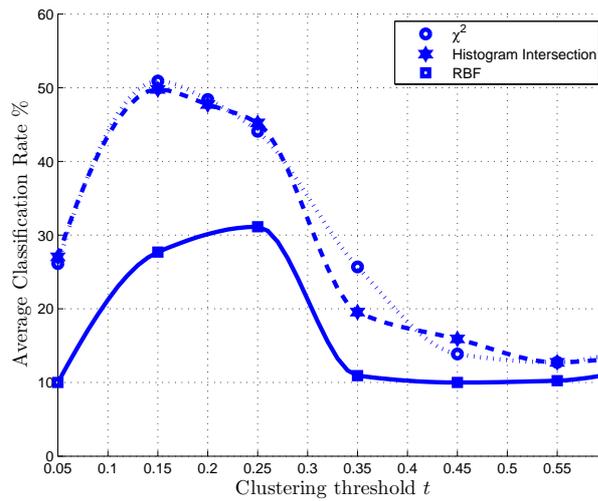
The results obtained with a traditional BoW approach, or building the codebook with an RNN clustering algorithm and the CPM methodology, can be improved further by the meta-clustering algorithm described in section 4.3.

For the experiments, we first run the correlation clustering based refinement for reducing the number of singletons. The algorithm is initialised with the codebook obtained with an RNN with threshold 0.25, *i.e.* the codebook that obtained the highest CP in the previous section. The threshold is increased from 0.25, in steps of 0.05, until a maximum of 0.9 is reached. After each iteration the average recognition rate is measured. For this experiment we use the  $\chi^2$ -Kernel which reported the best results in (van de Sande et al. 2008, Zhang et al. 2007).

Testing with the Caltech-101 subset of images, this refinement step finds that the best average recognition rate is obtained when the cut-off threshold varies from 0.25 to 0.9. This means that some clusters have been merged by the correlation clustering algorithm, and that the maximum threshold for some of them is now 0.9. The refined codebook is now made up of clusters with different cut-off thresholds, that is, these thresholds have been



(a)



(b)

Figure 4.8: (a) CPM performance. The threshold resulting in the maximum CP is 0.25 for the Caltech-101 subset. A comparison with the cluster precision measures of Mikolajczyk et al. (2005) and Stark and Schiele (2007) is shown too. (b) Average Classification Rate recognition vs. Clustering threshold. For the RBF kernel, the higher the CP the better the classification rate. Both the HIK and the  $\chi^2$ -Kernel obtain the best results for thresholds under the bound identified by the CPM algorithm. This Figure also shows that for thresholds over the bound the classification rate drastically drops.

(a)											(b)										
	ewer	sunflower	kangaroo	starfish	trilobite	menorah	helicopter	butterfly	brain	gran piano		ewer	sunflower	kangaroo	starfish	trilobite	menorah	helicopter	butterfly	brain	gran piano
ewer	<b>5</b>	3	1	4	0	0	1	2	1	1	ewer	<b>9</b>	2	6	1	0	2	3	2	0	1
sunflower	4	<b>12</b>	0	2	2	0	0	1	2	2	sunflower	5	<b>22</b>	2	2	2	0	0	5	4	2
kangaroo	4	4	<b>16</b>	10	2	1	3	6	0	0	kangaroo	3	4	<b>22</b>	9	4	3	7	8	3	0
starfish	0	0	0	<b>11</b>	1	1	0	1	0	0	starfish	4	3	1	<b>18</b>	3	1	1	1	1	0
trilobite	5	3	2	3	<b>24</b>	2	1	8	3	0	trilobite	2	2	3	4	<b>24</b>	0	0	6	1	0
menorah	9	2	7	4	5	<b>31</b>	6	4	4	11	menorah	5	0	0	3	3	<b>28</b>	5	2	1	5
helicopter	8	8	11	4	6	4	<b>28</b>	11	3	1	helicopter	3	1	2	0	0	3	<b>16</b>	2	1	0
butterfly	3	7	2	1	0	0	2	<b>4</b>	0	3	butterfly	10	8	7	3	2	3	7	<b>17</b>	1	0
brain	4	3	4	2	1	2	3	8	<b>36</b>	0	brain	1	0	0	1	2	1	3	2	<b>37</b>	0
gran piano	0	0	0	1	0	2	0	0	0	<b>31</b>	gran piano	0	0	0	1	1	2	2	0	0	<b>41</b>

Table 4.2: Confusion matrix before the adaptive clustering refinement process (a) and after (b). The entry in the  $i$ th row and  $j$ th column is the number of images from class  $i$  that were classified as class  $j$ .

separately optimised for each cluster.

The average classification rate has been increased from 43.4% to 52.5%, while the codebook size has been drastically reduced from 17575 to 6614, *i.e.* by 63%. This huge reduction is due to the high number of initial singletons the codebook had. Table 4.2 presents the confusion matrix for the classifier trained with the codebook before and after this first adaptive threshold refinement. It can be observed how the misclassifications decrease. So the refinement step obtains more discriminative codebooks reducing their size, both of which are desirable aspects.

After this first refinement process the resulting codebook has 6614 clusters. It consist of 8 singletons, 1571 clusters of size 2, 1223 clusters of size 3, and the rest of clusters. We again run the correlation clustering based refinement process, but this time to try to merge the clusters of size up to 3. The algorithm is initialised with the codebook obtained in the previous iteration. The threshold is increased from 0.5, in steps of 0.1, until a maximum of 0.9 is reached. The best average recognition rate is obtained in the third iteration, when the cut-off threshold varies from the minimum until 0.7. The resulting codebook has 3818 clusters, and it allows an average recognition rate of 58.33%. That is, the average classification rate has been increased from 52.5% to 58.33%, while the codebook size has been reduced from 6614 to 3818.

Figure 4.9 shows the class recognition rates before and after these two iterations of the correlation clustering based refinement. It is important to note that not all the class rates vary in the same way. In the experiments, in 80% of the classes the recognition rate is increased.

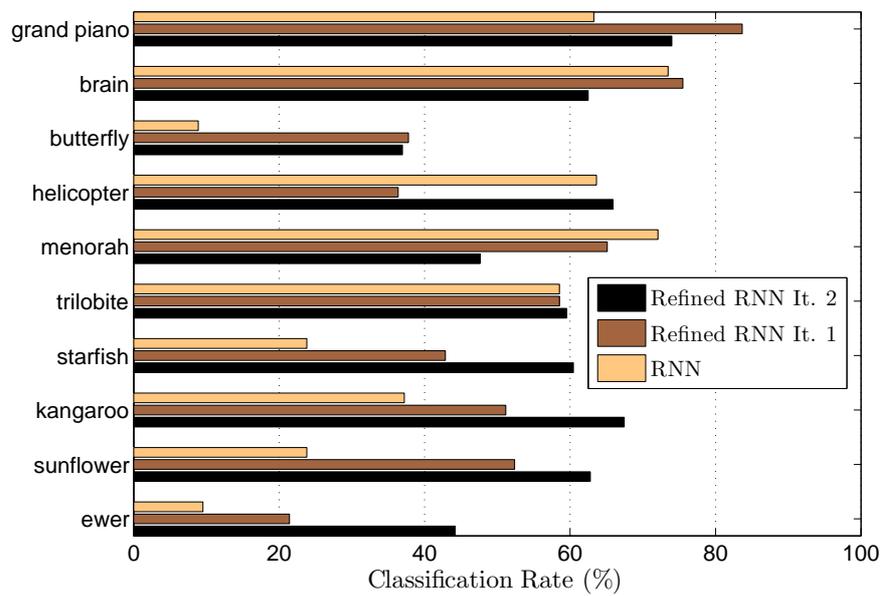


Figure 4.9: The Figure shows how the correlation clustering refinement iterations get to increase the average recognition rate. We show the class rate allowed by the original RNN vocabulary and the refined RNN codebooks after the first and the second iteration.

#### 4.4.4 A comparison with $K$ -means codebooks

So far we have shown how both the CPM algorithm and the correlation clustering based refinement perform with RNN codebooks. However, the most common clustering algorithm used in BoW approaches is  $K$ -means. An experimental comparison with vocabularies obtained with  $K$ -means clustering has also been carried out. Our objective is to evaluate: how the CP performs in combination with each clustering algorithm; what recognition performance the codebook allows; and the computational complexity.

##### 4.4.4.1 CPM performance

Using the same Caltech-101 subset described above,  $K$ -means codebooks of similar sizes to those obtained with the RNN clustering have been obtained. For each vocabulary we first measure its cluster precision and then the average classification rate each codebook casts. Figure 4.10 shows a comparison of the CP obtained for different  $K$ -means and RNN codebooks. Note the CP is similar at the beginning when the number of clusters is near the number of features. Once the number of clusters decreases, the CP for  $K$ -means codebooks decreases too. Note that there is no maximum for the CP when  $K$ -means codebooks are used. This is due to the fact that the RNN produces visually compact clusters, with a high proportion of singletons, which makes the CP increase when they merge during the first iterations. With  $K$ -means there is neither any guarantee that the clusters are visually compact nor that there are as many singletons. Because of the fixed value of  $K$ , some cluster centres may lie in-between several *real* clusters. That is,  $K$ -means has the defect that, in high-dimensional spaces, it tends to focus on the high-density area, resulting in clusters that extend quite far into the less dense areas, so not compact at all. However, with an agglomerative clustering algorithm, such as RNN, where the clusters aggregation process is heavily dominated by the similarity of the vectors they contain, the CPM is able to identify the threshold limit above which the average classification rate drastically decrease. This capacity is important, especially in high-dimensional spaces where the distances (similarities) tend to concentrate (Francois et al. 2007).

##### 4.4.4.2 Classification performance

We compare the average classification obtained with  $K$ -means, RNN and refined RNN codebooks. The most important results in classification are shown in Figure 4.11. We have used the HIK and the  $\chi^2$ -Kernel for this comparison. It is clear that  $K$ -means codebooks perform better than RNN codebooks, but after the correlation clustering based refinement, the refined

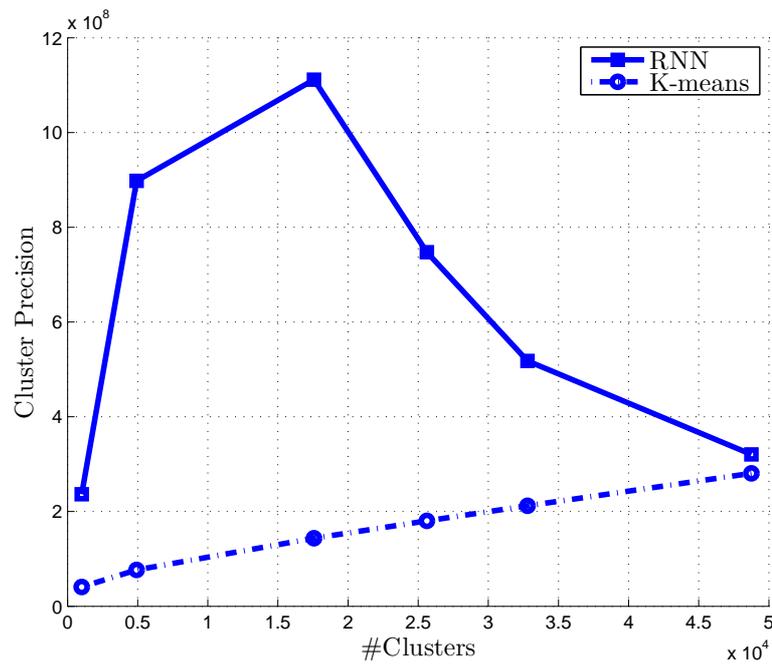


Figure 4.10: Comparison of the CP obtained with RNN and  $K$ -means codebooks. Note that there is no maximum for the CP when using  $K$ -means codebooks.

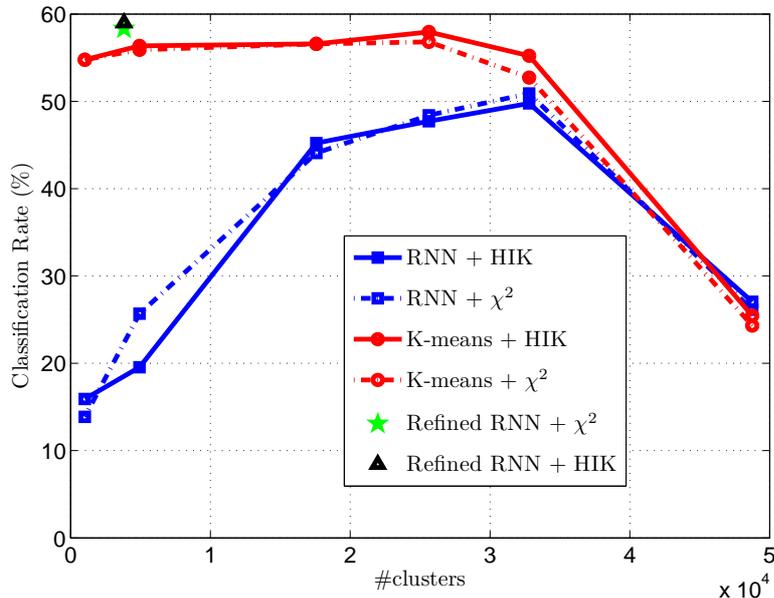


Figure 4.11: Average classification rate for different clustering algorithms ( $K$ -means, RNN, refined RNN), different vocabulary sizes and different kernels.

RNN codebooks obtain better results than  $K$ -means codebooks for both types of kernels. The results also indicate that the new CP measure is not correlated with the classification rate for  $K$ -means codebooks.

#### 4.4.4.3 Computational Complexity

Consider a classical BoW approach using  $K$ -means as clustering algorithm. To find the number of  $K$  clusters with which the codebook allows the best classification rate implies empirically determining the parameters of the whole system, including those parameters that typical classifiers, *e.g.* SVM. It is possible to define the run-time of an iteration  $i$  for a classical BoW as  $t_{BoW}^{(i)} = t_{Kmeans}^{(i)} + t_{train}^{(i)} + t_{val}^{(i)}$ , where  $t_{Kmeans}^{(i)}$  is the run-time of  $K$ -means clustering, and  $t_{train}^{(i)}$  and  $t_{val}^{(i)}$  are respectively the time spent training the classifier and testing with a validation set. Note that every time the number of clusters is changed, the  $K$ -means algorithm must be run from the beginning, although it is possible to use hierarchical approaches, *e.g.* vocabulary trees (Nister and Stewenius 2006), to speed up this process. Let  $M$  be the number of iterations until the desired classification rate is achieved, then  $t_{BoW} = \sum_{i=1}^M t_{BoW}^{(i)}$ .

On the other hand we have the CPM with RNN clustering scenario. The

CPM is an efficient scheme, which avoids the need to train the full system for different settings to evaluate a validation set. A CPM approach requires the clustering algorithm to be run with varying parameters, *e.g.* the threshold for the RNN clustering algorithm. It is possible to perform a full RNN clustering (*i.e.* with the maximum threshold) so as to save both the indices of clusters merged in every step and the similarities between them. With this information it is possible to rebuild a visual vocabulary for a different threshold at almost no computational cost. The run-time for a CPM approach with RNN can then be written as  $t_{CPM+RNN} = t_{RNN} + Mt_{CP} + (M - 1)t_{RNN_{new}}$ , where  $t_{CP}$  is the time used to measure the cluster precision in each CPM iteration,  $M$  is the number of iterations,  $t_{RNN}$  is the run time of a full RNN clustering algorithm, and  $t_{RNN_{new}}$  is the time taken by the algorithm to compute the new clusters when the threshold changes using the saved indices and similarities. Furthermore,  $t_{RNN} \gg (Mt_{CP} + (M - 1)t_{RNN_{new}})$ , so  $t_{CPM+RNN} \approx t_{RNN}$ .

The RNN clustering algorithm has  $O(N^2d)$  time complexity, which is high but almost independent of the number of clusters. The complexity of  $K$ -means is  $O(NKdl)$ . Recall  $l$  is the number of iterations until the algorithm converges. Within the context of category-level object recognition, normally the number of clusters  $K$  is high (sometimes it is in almost the same order as the number of features  $N$ , *e.g.* (Leibe et al. 2008b)). This implies that the run-time for  $K$ -means can exceed the one for RNN when  $K$  is large, *e.g.* (Leibe et al. 2006). In that case,  $t_{BoW} > t_{CPM+RNN}$ .

However, it is possible to use efficient implementations of  $K$ -means, such as (Kanungo et al. 2002, Elkan 2003), in order to reduce  $t_{BoW}$ . Furthermore, to speed up the RNN clustering algorithm we propose to use the accelerated version detailed in Chapter 5.

It is also important to note that RNN vocabularies do not obtain better classification results than  $K$ -means codebooks until the correlation clustering based refinement is processed. That is, we have to add to the time  $t_{CPM+RNN}$ , the time spent in this refinement, *i.e.*  $t_R$ . The run-time complexity of the correlation clustering approach used for clustering refinement is  $O(K^2I)$ , where  $K$  is the number of clusters and  $I$  is the maximum number of iterations the algorithm needs to get the final solution. Note that the number of clusters drastically decreases during the refinement process, so the refinement run-time decreases too. In our experiments, we have used  $I = 10^7$ , and we have confirmed that  $t_R + t_{CPM+RNN} < t_{BoW}$ .

## 4.5 Conclusions and future work

In this chapter we consider the problem of measuring and increasing the representativeness of visual words within the context of category-level object recognition. First, we have introduced an improved measure for CP, as well as a procedure to optimise the parameters during codebook construction without cross-validation, by maximising this CP measure. The CPM method measures the cluster precision for each class and automatically finds the thresholds for an RNN clustering algorithm that cast the best average recognition rates. A complete description of the method has been given, showing how to integrate an RNN clustering algorithm in it. CPM evaluates the intrinsic quality of the clusters for a classification task and as a result, allows us to compare the quality of clusters computed with different thresholds. Results confirm that the vocabularies obtained with CPM obtain better results.

On the other hand, a meta-clustering refinement algorithm has been presented. To the best of our knowledge, this is the first correlation clustering based approach for improving the class representativeness of visual words. The proposed methodology increases the average recognition rate and also dramatically compacts the size of the codebook.

Even though RNN is faster than  $K$ -means (for large values of  $K$ ), when large amounts of data are given a more efficient implementation is highly desirable. Chapter 5 is devoted to describe a new speeded up RNN algorithm via slicing the high dimensional space. Experimenting with other descriptors, detectors and datasets (*e.g.* PASCAL VOC Challenge datasets and ICARO) will also be considered.

Another line of research involves incorporating local information into the clustering process to improve the construction of a *semantic* visual codebook. Chapter 6 introduces a new approach, based on clustering aggregation techniques, to continue bridging the gap between visual features and semantic concepts.



# Chapter 5

## Fast Reciprocal Nearest Neighbours Clustering

*Machines take me by surprise with great frequency.*

Alan M. Turing, *Computing Machinery and Intelligence.*

---

This chapter presents a novel approach for accelerating the Reciprocal Nearest Neighbours (RNN) clustering algorithm, *i.e.* the fast-RNN. We speed up the nearest neighbour chains construction via the projection search paradigm. We describe an efficient implementation of the clustering algorithm and show extensive experimental results that illustrate their performance with high-dimensional local descriptors. A C++ implementation of our algorithm is made publicly available.

### 5.1 Introduction

The most time-consuming component for many computer vision problems consists in the Nearest Neighbour (NN) search in high-dimensional spaces. Two examples of such problems are: searching for the the best matches for local image features in large datasets (Lowe 2004, Philbin et al. 2007) and computing normalised cross-correlation for comparing image patches in gigantic image collections (Torralba et al. 2008a). The high-dimensional vector quantisation problem is also related to searching for NN in high-dimensional spaces, and it is a common problem for various applications in computer vision.

For instance, BoW approaches (see Section 2.1.1.1) are the most effective methods for category-level object recognition developed to date (Everingham et al. 2008, 2009b). In specific terms, these approaches extract local visual descriptors, quantise them, build histograms of the resulting visual words over the images and apply a discriminative classifier such as SVMs. Some examples are (Csurka et al. 2004, Khan et al. 2009, Sivic and Zisserman 2003, van de Sande et al. 2008, Zhang et al. 2007).

$K$ -means clustering is still the most widely used vector quantisation scheme within this context, even though more efficient and stable alternatives have been proposed, *e.g.* (Philbin et al. 2007, Nister and Stewenius 2006). However, the clustering solution depends on a random initialisation. Furthermore, if the number of outliers in the data distribution is large, the clustering solution is suboptimal. Finally, it requires the user to specify the number of clusters in advance.

To overcome  $K$ -means limitations, Mean-shift clusterers, such as (Jurie and Triggs 2005), have been also proposed. Moreover, other authors use agglomerative clustering techniques (Agarwal et al. 2004, Leibe et al. 2008a). In an agglomerative approach the number of clusters is determined automatically. However, both the runtime and memory requirements are often significantly higher for these agglomerative methods. As a result, given the large amounts of data that have to be processed, an efficient implementation of an agglomerative clustering algorithm needs to be obtained. Leibe et al. (2006) propose using the efficient RNN clustering algorithm (first described by de Rham (1980)) for local descriptors quantisation. In this chapter, we consider their work and present an accelerated version of this clustering algorithm: the fast-RNN.

The work of Nene and Nayar (1997) has inspired our approach to accelerating the RNN algorithm. They present an efficient method for searching for the NN within distance  $\epsilon$  in a high-dimensional space. However, a direct application of their approach to the RNN algorithm does not achieve better results than the standard RNN with simple linear NN search. One of the reasons is that the dataset of points where we need to find the NN of a query point  $\mathbf{q}$  changes in every iteration of the main loop of the RNN algorithm. Which means that we have to continuously update (or rebuild) the data structure defined in (Nene and Nayar 1997) and this is the main drawback of a direct application.

In this chapter, we propose a method for accelerating the RNN algorithm and not just the NN search. The idea is simple. Via *slicing* the high-dimensional space in one dimension, we speed up the NN chains construction and therefore accelerate the entire RNN clustering. Moreover, we use an efficient data structure for space partitioning. The time complexity

for building this data structure is on average  $O(n \log n)$ , with  $n$  being the number of vectors to quantise.

Section 5.2 introduces a detailed description of the standard RNN clustering algorithm, and in Section 5.3 we present the novel approach for accelerating it: the fast-RNN. Section 5.4 presents the results obtained in a comparison between RNN and fast-RNN when quantising high-dimensional vectors. Finally, Section 5.5 concludes the chapter.

## 5.2 Reciprocal Nearest Neighbours Clustering

Clustering algorithms can be broadly divided into two groups (Jain 2009): partitional and hierarchical. Hierarchical clustering algorithms recursively find nested clusters either in divisive mode or in agglomerative mode. All hierarchical agglomerative methods follow the same principle. Starting with each vector as a separate cluster, the two most similar clusters are merged until a similarity criterion between their constituent vectors is fulfilled. Typically, these agglomerative approaches are classified by the way the similarity of a newly built cluster to all others is computed. Useful clustering metrics are described using the updating formula proposed by Lance and Williams (1967):

$$\begin{aligned} \text{sim}(C_{u \cup v}, C_i) = & a_u \text{sim}(C_u, C_i) + a_v \text{sim}(C_v, C_i) + b \text{sim}(C_u, C_v) \\ & + c |\text{sim}(C_u, C_i) - \text{sim}(C_v, C_i)|, \end{aligned} \quad (5.1)$$

where  $C_u, C_v$  and  $C_i$  represent any clusters and  $C_{u \cup v}$  the cluster after merging clusters  $C_u$  and  $C_v$ . Lance and Williams (1967) show the choice for the parameters  $a_u, a_v, b$  and  $c$ . In this study, we focus on the Unweighted Pair Group Method with Arithmetic mean (UPGMA) approach, also known as the *average-link* method, where:  $a_u = \frac{n_u}{n_u + n_v}$ ,  $a_v = \frac{n_v}{n_u + n_v}$ ,  $b = 0$ ,  $c = 0$ .  $n_u$  and  $n_v$  are the sizes of the respective clusters.

The main drawback of the standard average-link algorithm is its  $O(n^2 \log n)$  runtime and  $O(n^2)$  space complexity (with  $n$  being the number of vectors to be quantised). The second drawback is the result of the input for the clustering algorithm being  $n \times n$  distance/similarity matrix, and hence the memory requirements impose a practical limit. In other words, since an  $O(n^2)$  similarity matrix has to be stored, the main memory typically available on modern machines has already been exceeded by  $n > 25000$ . These algorithms are therefore used with small data sets. Fortunately, a more efficient algorithm is available. It runs in  $O(n^2 d)$  (where  $d$  is the dimensionality of data) and

needs only  $O(n)$  space: the RNN clustering.

The RNN algorithm for agglomerative clustering was first described by de Rham (1980), and recovered for the computer vision community by Leibe et al. (2006). It is based on the construction of RNN pairs of vectors  $\mathbf{x}_i$  and  $\mathbf{x}_j$ , so that  $\mathbf{x}_i$  is the NN to  $\mathbf{x}_j$ , and vice versa. As soon as an RNN pair is found it can be agglomerated. Benzécri (1982) developed an efficient implementation that ensures that RNN can be found with as little re-computation as possible. This is achieved by building an NN chain, which consists of an arbitrary vector, followed by its NN, which is again followed by its NN among the remaining points and so on.

An NN chain of length  $l$  can be hence defined as the sequence of vectors

$$\{\mathbf{x}_1, \mathbf{x}_2 = \text{NN}(\mathbf{x}_1), \dots, \mathbf{x}_{l-1} = \text{NN}(\mathbf{x}_l), \mathbf{x}_l = \text{NN}(\mathbf{x}_{l-1})\} ; \quad (5.2)$$

where  $\text{NN}(\mathbf{x}_i)$  is the NN of  $\mathbf{x}_i$ . The properties of this type of chain are as follows:

1. The distances between adjacent vectors in the NN chain are monotonically decreasing.
2. The NN chains can not contain a circuit.
3. The chain must end at some point, and the last pair of nodes are NNs of each other, *i.e.* RNN.

The algorithm starts with an arbitrary centroid (vector). An NN chain is then built. When an RNN pair is found, *i.e.* no more centroids can be added to the current chain, the corresponding clusters are merged if their similarity is above a fixed cut-off threshold, and otherwise the algorithm discards the whole chain. This way of merging clusters can be applied whenever the distance matrix  $D$  satisfies the reducibility property

$$D(C_i, C_j) \leq \min\{D(C_i, C_k), D(C_j, C_k)\} \leq D(C_{i \cup j}, C_k) , \quad (5.3)$$

where  $D(C_i, C_j)$  is the distance between clusters  $C_i$  and  $C_j$ , and  $C_{i \cup j}$  is the cluster after merging  $C_i$  and  $C_j$ . Property (5.3) guarantees that when clusters are merged in this way, the NN relations for the remaining chain members are unaltered, and they can be therefore used for the next iteration. When the current chain is empty or discarded, a new arbitrary point is selected, and a new NN chain is started.

Benzécri (1982) demonstrates that a full clustering requires at most  $3(n-1)$  iterations of the main loop. The run-time is thus bound by the time required to find NNs, which in the simplest case is  $O(nd)$ . The key point

is how to recompute the similarity between a new cluster (after merging an RNN pair) and the others, *i.e.* how to efficiently measure the distances between the new centroid and the rest. Day and Edelsbrunner (1984) and Leibe et al. (2006) show this can be done efficiently if the cluster similarity can be expressed in terms of centroids, which holds for a group average criteria based on correlation or Euclidean distances. Let  $C_x = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}$  and  $C_y = \{\mathbf{y}_1, \dots, \mathbf{y}_m\}$  be two clusters. The group average criteria is defined as

$$\text{similarity}(C_x, C_y) = \frac{1}{nm} \sum_{i=1}^n \sum_{j=1}^m \text{similarity}(\mathbf{x}_i, \mathbf{y}_j). \quad (5.4)$$

Considering the similarity based on Euclidean distances, then

$$\text{similarity}(C_x, C_y) = -\frac{1}{nm} \sum_{i=1}^n \sum_{j=1}^m (\mathbf{x}_i - \mathbf{y}_j)^T (\mathbf{x}_i - \mathbf{y}_j). \quad (5.5)$$

Let  $\mu_x, \mu_y$  and  $\sigma_x^2, \sigma_y^2$  be the means and variances of clusters  $C_x$  and  $C_y$  respectively. Equation (5.5) can be written as

$$\text{similarity}(C_x, C_y) = -((\sigma_x^2 + \sigma_y^2) + (\mu_x - \mu_y)^2). \quad (5.6)$$

By using equation (5.6), when merging RNN pairs, the new distances can be computed in a constant time and only the mean and variance of each cluster need to be stored. Moreover, both parameters can be incrementally computed. Let  $C_x$  and  $C_y$  be an RNN pair of clusters, and  $C_{x \cup y}$  the new cluster after merging  $C_x$  and  $C_y$ . The mean and variance of  $C_{x \cup y}$  are incrementally computed as

$$\mu_{x \cup y} = \frac{n\mu_x + m\mu_y}{n + m}, \quad (5.7)$$

$$\sigma_{x \cup y}^2 = \frac{1}{n + m} (n\sigma_x^2 + m\sigma_y^2 + \frac{nm}{n + m} (\mu_x - \mu_y)^2). \quad (5.8)$$

In conclusion, this efficient implementation described in (Leibe et al. 2006) has  $O(n^2d)$  time and  $O(n)$  space complexity. However, when considering the use of the RNN clustering algorithm with high-dimensional data, such as SIFT (Lowe 1999) or Speeded Up Robust Features (SURF) (Bay et al. 2006) descriptors, the time complexity is still high. This is to be expected since the algorithm relies heavily on the search for NNs. In order to further improve the run-time of the algorithm, we present an efficient technique for speeding up NNs chain construction in the following section.

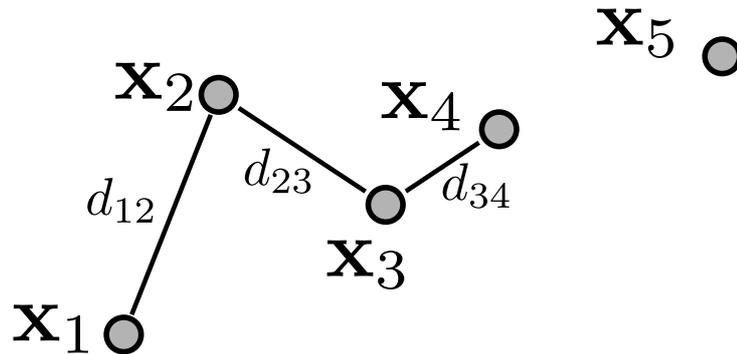


Figure 5.1: NN chain example. The NN chain starts with  $\mathbf{x}_1$ , and contains  $\{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4\}$ . Note that  $d_{12} > d_{23} > d_{34}$ .  $\mathbf{x}_5$  is not added to the NN chain because the distance  $d_{45} > d_{34}$ .

### 5.3 Fast-RNN

As stated in Section 5.2, the efficient implementation of the RNN algorithm is based on two principles:

- the construction of NN chains (Benzécri 1982).
- an efficient mechanism for measuring the distances between a new centroid and the other centroids (Day and Edelsbrunner 1984, Leibe et al. 2006).

We present an approach for accelerating the NN chain construction, thereby speeding up the RNN clustering. Furthermore, this new algorithm is able to efficiently search for NN in dynamic sets of points with high dimensionality.

Let us assume a set  $S$  of 5 vectors,  $S = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5\}$ . We run the RNN clustering algorithm on this set  $S$ , with a fixed cut-off threshold  $t$ . If we assume that the RNN randomly starts from the  $\mathbf{x}_1$  vector, then the NN chain shown in Figure 5.1 is constructed. The distances between adjacent elements are monotonically decreasing, and the last pair of vectors, *i.e.*  $\mathbf{x}_3$  and  $\mathbf{x}_4$ , are RNN. If  $d_{34} \leq t$ , then the algorithm agglomerates vectors  $\mathbf{x}_3$  and  $\mathbf{x}_4$  in a unique cluster, computes its new centroid and adds the new centroid to the set of points  $S$ . However, if  $d_{34} > t$ , then the whole chain is discarded, and each of its elements is considered to be a separate cluster.

In both cases, we can see that the set of points  $S$  is continuously changing. For instance, every vector added to a NN chain is extracted from  $S$ , and when two clusters are agglomerated, the new centroid has to be inserted in  $S$ . In other words,  $S$  is a dynamic set.

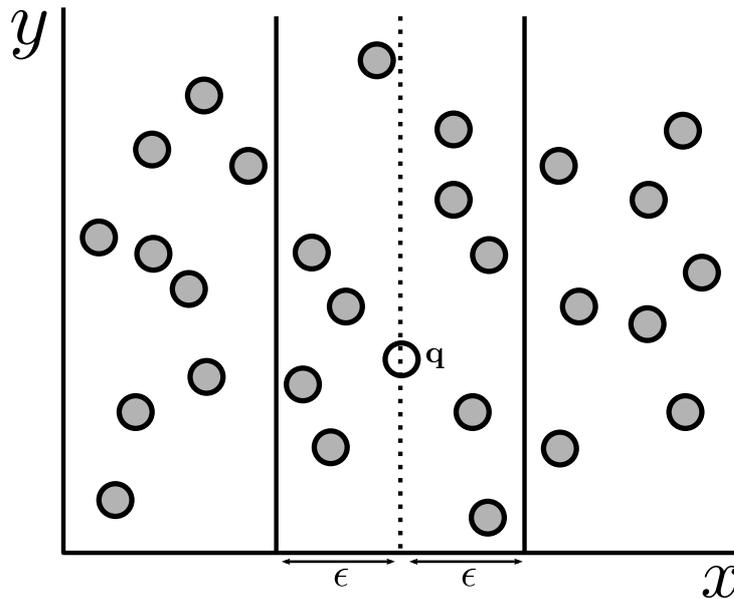


Figure 5.2: Slicing a 2-dimensional space.

Since we have to deal with such a dynamic set of points with high dimensionality, we propose the following algorithm, which is based on the projection search paradigm first described in (Friedman et al. 1975).

As in (Nene and Nayar 1997), the objective is to find the point in the set of points  $S$  that is closest to a query point  $\mathbf{q}$  and within a distance  $\epsilon$ . Instead of building a hypercube with sides  $2\epsilon$ , we propose finding all the points that lie within a slice of the  $d$ -dimensional space of width  $2\epsilon$  centred at point  $\mathbf{q} = (q_1, q_2, \dots, q_d)$ . That is, the  $i$ -slice is defined as the region confined by two parallel planes, perpendicular to the  $i$ th coordinate axis, separated a distance  $2\epsilon$  and centred at  $q_i$ . The NN search is done just with the points inside the slice. We illustrate this procedure in Figure 5.2 for a 2-dimensional space.

As a result, unlike (Nene and Nayar 1997), we merely construct a slice of the high-dimensional space. This aspect is critical when dealing with high-dimensional and dynamic datasets of points. First, the data structures, which are described in Section 5.3.3, are easy and fast to build and maintain. When points are extracted or inserted the data structure can be updated at a low cost. Furthermore, the size of the data structure does not grow with the dimensionality, which is desirable. Before describing the implemented data structure, we first discuss how to use this slicing approach efficiently when building NN chains.

### 5.3.1 Building NN chains via *slicing*

We are interested in building NN chains, *i.e.* chains of NN in which the distances between adjacent elements monotonically decrease. Although the main concepts concerning NN chains have been stated in Section 5.2, we provide a formal description here.

We first assume there is a set of  $N$  points  $S = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}$  which belong to a  $d$ -dimensional space. We assume that a metric  $d(\mathbf{x}_i, \mathbf{x}_j)$  is defined between points in  $S$ .

**Definition.** A metric  $d(\mathbf{x}_i, \mathbf{x}_j)$  is defined for any  $\mathbf{x}_i$  and  $\mathbf{x}_j$  in  $S$ . For all  $\mathbf{x}_i$ ,  $\mathbf{x}_j$  and  $\mathbf{x}_k$  in  $S$ , this metric is required to satisfy the following properties:

1.  $d(\mathbf{x}_i, \mathbf{x}_j) \geq 0$ .
2.  $d(\mathbf{x}_i, \mathbf{x}_j) = 0$  if and only if  $\mathbf{x}_i = \mathbf{x}_j$ .
3.  $d(\mathbf{x}_i, \mathbf{x}_j) = d(\mathbf{x}_j, \mathbf{x}_i)$ .
4.  $d(\mathbf{x}_i, \mathbf{x}_k) \leq d(\mathbf{x}_i, \mathbf{x}_j) + d(\mathbf{x}_j, \mathbf{x}_k)$ .

**Definition.** A point  $\mathbf{x}_j$  is the nearest neighbour of  $\mathbf{x}_i$ , denoted by  $\mathbf{x}_j = NN(\mathbf{x}_i)$ , in the set of points  $S$ , if  $d(\mathbf{x}_i, \mathbf{x}_j) \leq d(\mathbf{x}_i, \mathbf{x}_k) \forall \mathbf{x}_k \in S / \mathbf{x}_k \neq \mathbf{x}_i$ .

**Definition.** Two points  $\mathbf{x}_i$  and  $\mathbf{x}_j$  are RNN if  $\mathbf{x}_j = NN(\mathbf{x}_i)$  and  $\mathbf{x}_i = NN(\mathbf{x}_j)$ .

With these definitions we can easily define a NN chain as follows.

**Definition.** An NN chain  $C$  is a sequence of points  $C = \{\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k, \dots, \mathbf{x}_p, \mathbf{x}_q\}$ , such that  $\mathbf{x}_j = NN(\mathbf{x}_i)$ ,  $\mathbf{x}_k = NN(\mathbf{x}_j)$ ,  $\dots$ ,  $\mathbf{x}_q = NN(\mathbf{x}_p)$ , and  $\mathbf{x}_p = NN(\mathbf{x}_q)$ .

It is straightforward to note that the distances between adjacent elements in an NN chain are monotonically decreasing.

**Proposition.** If  $C = \{\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k, \dots, \mathbf{x}_p, \mathbf{x}_q\}$  is a NN chain then,  $d(\mathbf{x}_i, \mathbf{x}_j) \leq d(\mathbf{x}_j, \mathbf{x}_k) \leq \dots \leq d(\mathbf{x}_p, \mathbf{x}_q)$ .

In this context, we can use the slicing approach presented in the previous section to build NN chains in the following way. Any NN chain starts with a random point  $\mathbf{x}_i$ . Our first task is to determine the nearest neighbour of  $\mathbf{x}_i$  in  $S$ , *i.e.*  $\mathbf{x}_j = NN(\mathbf{x}_i)$ . To that end, we obtain the first slice of width  $2\epsilon$  centred at  $\mathbf{x}_i$ . All the points inside this slice are included on a candidate list. We perform the search for the NN of  $\mathbf{x}_i$  considering only the points on the

candidate list. Once  $\mathbf{x}_j$  is identified, we search for its NN, *i.e.*  $\mathbf{x}_k = NN(\mathbf{x}_j)$ , via slicing again, and so on.

What it is known is that the distances between adjacent elements in a NN chain are decreasing. For this reason, in each iteration we can assign the value of the last distance between nearest neighbours in the NN chain to  $\epsilon$ . If there are not points within the slice for that  $\epsilon$ , we can stop building the NN chain. If we proceed in this way, as the NN chain grows, the slices get thinner, and the NN searches run faster.

This procedure for updating  $\epsilon$  is adequate when we have to deal with points in low-dimensional spaces. However, building NN chains for high-dimensional data using this approach is not efficient at all, if we recall the concentration of distances problem (Beyer et al. 1999, Francois et al. 2007): in high-dimensional spaces, the norm used to define the distance has the strange property of concentrating. The theorem by Beyer et al. (1999) can be formally stated as follows.

**Theorem.** *Let  $D_{max_d}$  and  $D_{min_d}$  be the maximum and the minimum distance of a data point of dimensionality  $d$  to origin, respectively. Then,*

$$\lim_{d \rightarrow \infty} \frac{D_{max_d} - D_{min_d}}{D_{min_d}} \rightarrow 0.$$

*Proof.* See (Beyer et al. 1999). □

This theorem states that under certain general conditions, the difference between the distances from the nearest and farthest points, *i.e.*  $D_{max_d} - D_{min_d}$ , does not increase with dimensionality as fast as the distance from the nearest point, *i.e.*  $D_{min_d}$ . This means that the minimum and maximum distances from a query point to points in the dataset become increasingly closer as dimensionality increases. For instance, in a set of SIFT vectors, most of the descriptors are at almost the same distance. As a result, if we update  $\epsilon$  with the last distance in the NN chain, we will not trim the number of vectors we have for comparison. A different approach for fixing  $\epsilon$  is needed within this context.

Nene and Nayar (1997) describe a method for determining the minimum value of  $\epsilon$  necessary to ensure that at least one point is found within a hypercube of size  $2\epsilon$  with high probability. It is possible to follow a similar procedure when using the proposed slicing procedure, and this is described in Section 5.3.2.

However, an exhaustive search within the candidate list does not always find the NN of  $x_i$  in  $S$ . The metric used in the RNN clustering algorithm is defined by the  $L_2$  norm. However, the slicing approach presented does not find points within  $L_2$ . When performing the linear search with the points

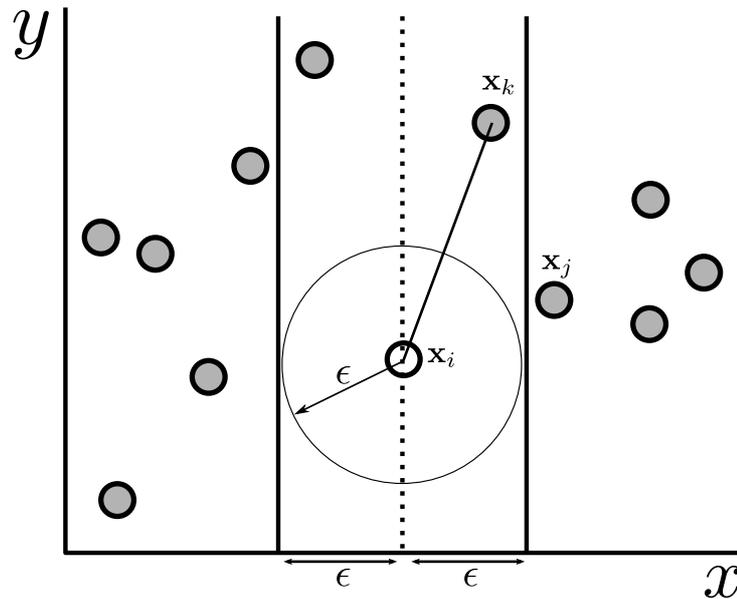


Figure 5.3: Slicing a 2-dimensional space.  $\mathbf{x}_j$  is closer to  $\mathbf{x}_i$  than  $\mathbf{x}_k$ , but it lies outside the slice.

in the candidate list, we must therefore check whether the distance to the nearest point inside the slice, *i.e.*  $d(\mathbf{x}_i, \mathbf{x}_j)$ , satisfies this condition  $d(\mathbf{x}_i, \mathbf{x}_j) \leq \epsilon$ . If not, we can not guarantee that  $\mathbf{x}_j = NN(\mathbf{x}_i)$ . Figure 5.3 illustrates this problem with an example.

In such a case, when no NN is found within the slice, bigger slices must be generated until  $\epsilon > d_{last}$  (with  $d_{last}$  being the distance between the last two elements in the NN chain) or we find a NN. To avoid this iterative process, we propose building only two slices. The first is built using an adequate  $\epsilon$  that guarantees a significant trim in the number of points and a high probability of success. The second slice is built fixing  $\epsilon$  to  $d_{last}$ . If the NN is not found in the first slice, we search in the second one. See Figure 5.4.

### 5.3.2 Determining $\epsilon$

Because the number of points inside the slice depends on the value of  $\epsilon$ , the efficiency of the proposed algorithm critically depends on  $\epsilon$  too. In Nene and Nayar (1997), a method for determining the minimum value of  $\epsilon$  necessary to ensure that at least one point is found within a hypercube of size  $2\epsilon$  with high probability is given.

We can use a similar procedure simply to ensure that our slice of width

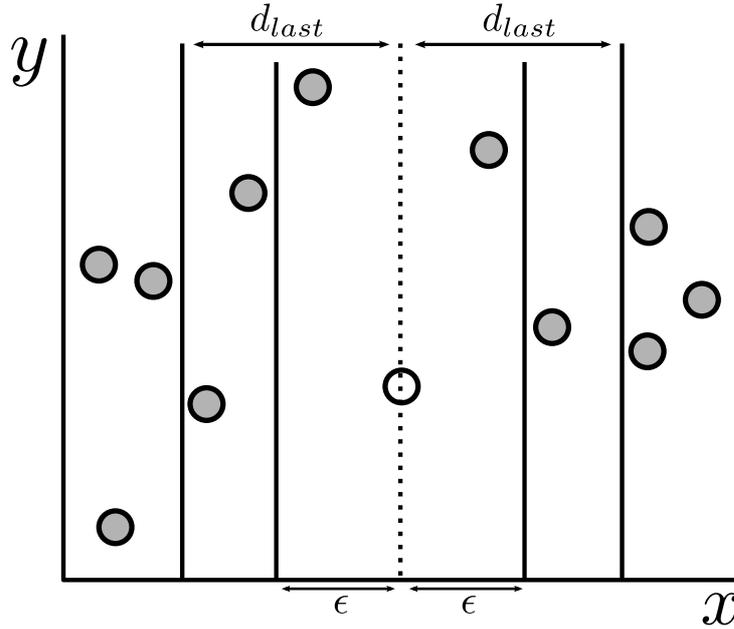


Figure 5.4: Slicing a 2-dimensional space. We build two slices, the interior one of width  $2\epsilon$ , and the exterior with width  $2d_{last}$ .

$2\epsilon$  contains at least one point. We focus our study on the specific case in which the set of points along each dimension is normally distributed. This assumption can be made, if we use SIFT, SURF or PCA-SIFT (Ke and Sukthankar 2004) descriptors (see the normal probability plots in Figure 5.5).

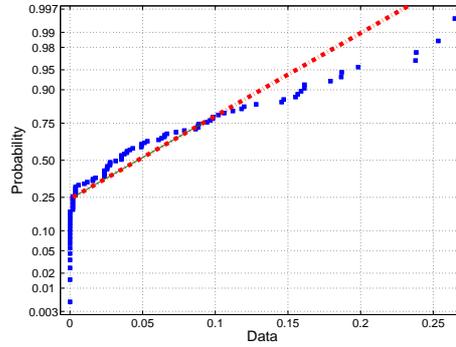
Our aim is to analytically compute the width of the thinnest slice, given that we want to be able to guarantee that it is not empty with probability  $p$ . Let  $N_s$  be the number of points within an slice of width  $2\epsilon_{min}$ . In order to determine the average number of points that lie within the slice, we have to compute  $E[N_s]$ . We can define  $Z_i$  as the distance between  $\mathbf{q}_i$  and any point in the slice. We define  $P_c$  as the probability that any projected point in the set of points is within distance  $\epsilon$  from  $\mathbf{q}_i$ , *i.e.*,

$$P_c = P\{-\epsilon \leq Z_i \leq \epsilon | \mathbf{q}_i\} . \quad (5.9)$$

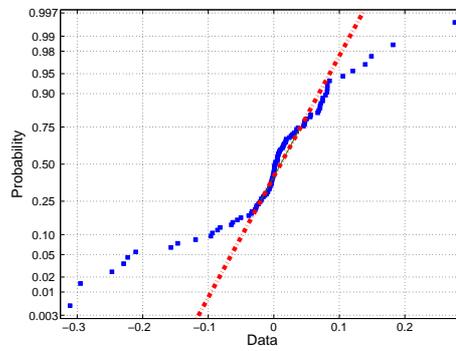
Regardless of the distribution of points,  $N_s$  is binomially distributed:

$$P\{N_s = k | \mathbf{q}_i\} = P_c^k (1 - P_c)^{n-k} \binom{n}{k} . \quad (5.10)$$

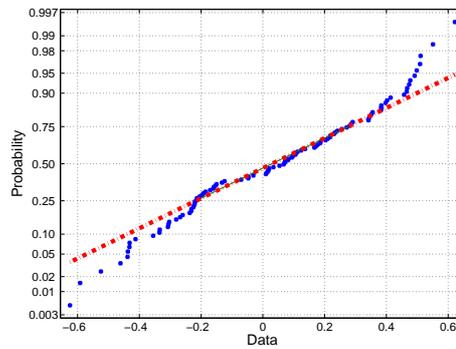
We focus on the scenario where the set of points is normally distributed,



(a) SIFT



(b) SURF



(c) PCA-SIFT

Figure 5.5: Normal probability plot for a random coordinate of a random selection of (a) SIFT, (b) SURF and (c) PCA-SIFT descriptors.

*i.e.*

$$f_{Z_i|\mathbf{q}_i}(z) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(z - q_i)^2}{2\sigma^2}\right) \quad (5.11)$$

$P_c$  can then be written as

$$P_c = P\{-\epsilon \leq Z_i \leq \epsilon | \mathbf{q}_i\} = \int_{-\epsilon}^{\epsilon} f_{Z_i|\mathbf{q}_i}(z) dz, \quad (5.12)$$

$$P_c = \frac{1}{2} \left( \operatorname{erf}\left(\frac{\epsilon - \mathbf{q}_i}{\sigma\sqrt{2}}\right) + \operatorname{erf}\left(\frac{\epsilon + \mathbf{q}_i}{\sigma\sqrt{2}}\right) \right). \quad (5.13)$$

The probability  $p$  that the slice contains at least one point is now

$$\begin{aligned} p &= P\{N_s > 0 | \mathbf{q}_i\} = 1 - P\{N_s = 0 | \mathbf{q}_i\} \\ &= 1 - (1 - P_c)^n \\ &= 1 - \left(1 - \frac{1}{2} \left( \operatorname{erf}\left(\frac{\epsilon - \mathbf{q}_i}{\sigma\sqrt{2}}\right) + \operatorname{erf}\left(\frac{\epsilon + \mathbf{q}_i}{\sigma\sqrt{2}}\right) \right)\right)^n \end{aligned} \quad (5.14)$$

We can vary the probability  $p$  between 0.1 and 0.9 to estimate the corresponding values of  $\epsilon$ . Using Equation 5.14,  $\epsilon$  is plotted against  $p$  in Figure 5.6. For this purpose we obtained a set of 100.000 SIFT vectors extracted from random images from the database ICARO (see Section 3.3). We set the number of descriptors  $n$  at different values between 1000 and 100.000. Note that the  $\epsilon$  required for building non-empty slices is very low for probabilities of success near 0.9, *e.g.* an  $\epsilon = 0.012$  guarantees a probability of success of 0.9 when  $n = 1000$ .

However, this study only guarantees that the slice contain at least one point, and what we want to guarantee is that within a slice of width  $2\epsilon$  we find the closest point to the query point  $\mathbf{q}$  at distance less than or equal to  $\epsilon$ . In practise, one has to set  $\epsilon$  to larger values, but guaranteeing that the number of points within the slice always remains small.

### 5.3.3 Data Structure

In this section we briefly introduce the data structure used for the implementation of this fast-RNN algorithm.

Our implementation uses a simple dynamic structure and 1D binary searches to efficiently find points inside the region defined by two parallel planes. First, we assume that the set of points we are dealing with is dynamic. For this reason, the data structure needs to be updated whenever the

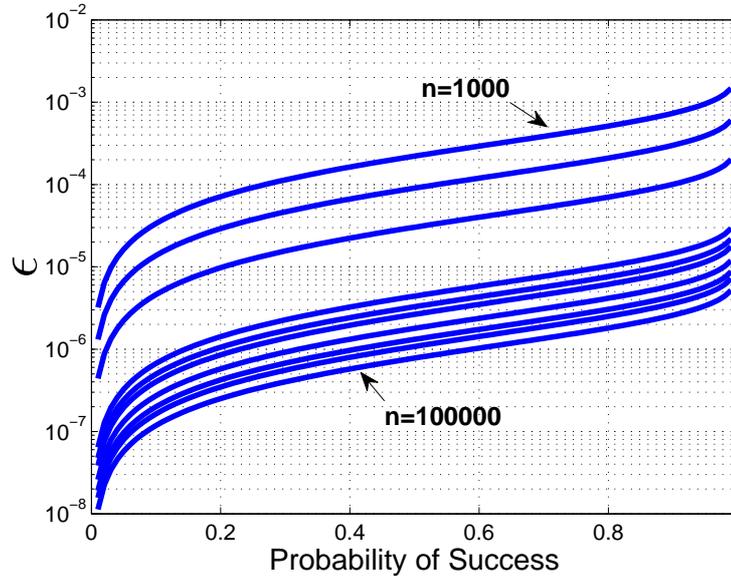


Figure 5.6:  $\epsilon$  vs. Probability of success. These results are obtained using a set of SIFT descriptors. We set the number of descriptors  $n$  to different values between 1000 and 100000.

set changes. In an RNN clustering algorithm we typically have to deal with:

- extractions of vectors from the set of points.
- insertions of new vectors when a pair of centroids is merged.

If the dimensionality of data is  $d$ , only coordinate  $j$ , where  $0 < j < d$ , is stored as an 1D array. This is called the Sorted Coordinate Array (SCA). In other words, we construct only one slice, which is perpendicular to the  $j$ th coordinate axis. Let us assume that our objective is to find the NN, in a set  $S$  of  $n$  points, of a given point  $\mathbf{x}_i$ , with coordinates  $\mathbf{x}_i = (x_{i_1}, \dots, x_{i_j}, \dots, x_{i_d})$ . In order to construct the candidate list efficiently, we have to search for those points that lie between two parallel planes perpendicular to the  $j$ th coordinate axis, centred at  $x_{i_j}$  and separated by a distance  $2\epsilon$ ; *i.e.*, our aim is to identify the points with  $j$ th coordinate in the SCA between the limits  $x_{i_j} - \epsilon$  and  $x_{i_j} + \epsilon$ .

The SCA is sorted in order to build the candidate list of points with two binary searches, and every binary search has a complexity in the worst case of  $O(\log n)$ .

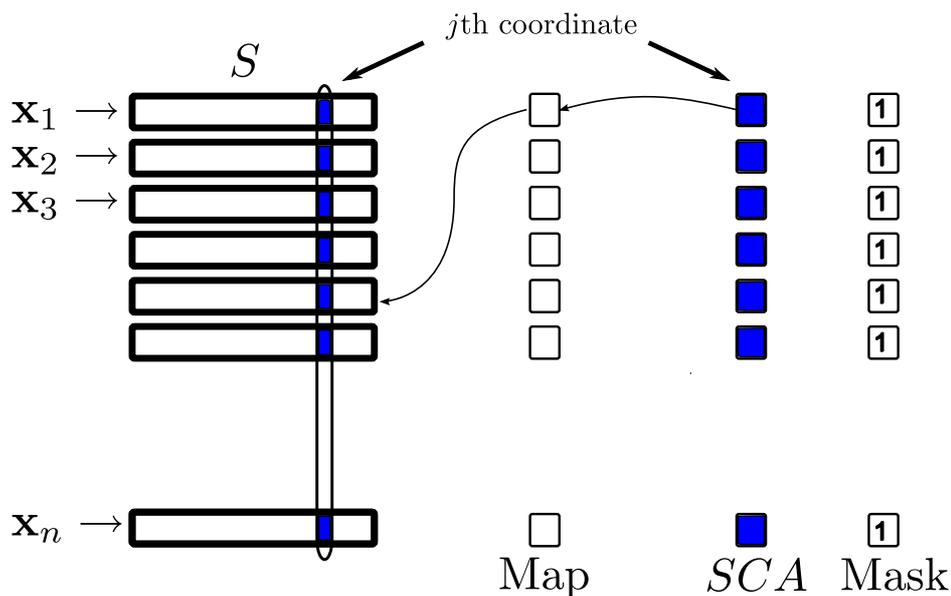


Figure 5.7: Data structures of Fast-RNN.

In our C++ implementation, the set of points  $S$  is built as a *list* (Stroustrup 2000) of vectors, where efficient insertions and removals of elements can take place anywhere in the list. In order to map a coordinate in the SCA to its corresponding point in the set of points  $S$  we maintain an array of *iterators* (Stroustrup 2000) where each element points to its corresponding vector in the *list*  $S$ .

We maintain a 1D array of boolean elements to deal with the insertions and deletions of points. Every element acts as a mask, indicating whether its position has been deleted (*false*) or not (*true*). When deleting an element, we first mark its mask to *false*, and then we delete the element from the list  $S$ . When a new vector has to be inserted, we first insert it in the list  $S$ , then with a binary search we determine the corresponding position of its  $j$ th coordinate in the SCA, and finally, we update the SCA and the mask. Figure 5.7 shows the detailed data structures.

Note that the SCA does not grow when we insert elements in  $S$ . Every insertion is associated with an agglomeration of two vectors that have been extracted beforehand. When an element is inserted, there are therefore at least two elements in SCA marked as deleted, and the algorithm simply updates the SCA by moving the elements within the array.

### 5.3.4 Discussion

We are interested in accelerating the RNN clustering algorithm, and an efficient NN search algorithm can help to speed up the NN chains construction. However, a direct application of the algorithm of Nene and Nayar (1997) does not help to speed up the RNN clustering process.

Nene and Nayar (1997) present an approach for searching for NN based on the projection search paradigm of Friedman et al. (1975). In specific terms, their method is based on using the  $L_\infty$  norm to quickly find all points within the hypercube of size  $2\epsilon$  around a query point  $\mathbf{q}$ . Once the hypercube is built, they search for the NN of  $\mathbf{q}$  within it. The points within the hypercube are what they call the *candidate list*.

Given a set of points  $S$  (with dimensionality  $d$ ), their goal is to find the closest point to a query point  $\mathbf{q} = (q_1, q_2, \dots, q_d)$  in  $S$ . The method first builds a candidate list with the points that lie inside the hypercube of side  $2\epsilon$  centred at  $\mathbf{q}$ . The points within the cube are found as follows. The set of points  $S$  is stored as a collection of  $d$  1D arrays, where the  $i$ th array contains the  $i$ th coordinate of all vectors in  $S$ .

Then, the algorithm starts with the first coordinate array, and identifies the points in  $S$  that lie between a pair of parallel hyperplanes perpendicular to the first coordinate axis, separated by a distance of  $2\epsilon$ , and centred at  $q_1$ . In other words, they find those points that their first coordinate lies in the interval  $[q_1 - \epsilon, q_1 + \epsilon]$ . The algorithm iterates on  $i = 2, 3, \dots, d$  as follows. In iteration  $i$ , it checks every point on the candidate list to see if its  $i$ th coordinate lies in the interval  $[q_i - \epsilon, q_i + \epsilon]$ . Points with  $i$ th coordinate that lies outside this range are discarded from the candidate list.

Nene and Nayar (1997) define a data structure for building this candidate list efficiently. First they propose sorting each  $i$ th-coordinate array in order to identify the points within intervals  $[q_i - \epsilon, q_i + \epsilon]$  with just two binary searches. Next, they build two maps: a forward map to relate a point in  $S$  to a point in the ordered set, and a backward map to relate a coordinate in the ordered set to the corresponding coordinate in the set of points  $S$ .

Nene and Nayar (1997) assume that the set of points  $S$  is static and hence for a given  $S$ , the data structures only need to be constructed once. This is not clearly the case when building NN chains in an RNN clustering algorithm: the set of points changes continuously. Moreover, Nene and Nayar (1997) design a data structure based on simple integer arrays. This means that the operations used for trimming the high-dimensional space to generate the candidate list are only integer comparisons and memory lookups. This feature is critical to the efficiency of the proposed algorithm in (Nene and Nayar 1997). However, when dealing with a dynamic set of points, a static

structure like this is not desirable.

The experiments in (Nene and Nayar 1997) only deal with datasets with dimensionality up to 35. The most commonly used local descriptors in computer vision are currently SIFT and SURF, both of which have dimensionality 128. In (Nene and Nayar 1997), as we have seen, computational efficiency is achieved by limiting the search to a small region around the query point. In their implementation, the region around each point is in the shape of a hypercube, but in order to conform to the properties of the Euclidean distance, only points within a hypersphere inscribed within the hypercube are considered valid, and the rest are discarded. However, in a 128-dimensional space the volume of the hypersphere of radius  $\epsilon$  is much smaller than the volume of the enclosing hypercube. Only a small fraction of the returned points will therefore actually be within the search distance required.

Moreover, we have the problem of the concentration of distances in high-dimensional spaces. This concentration is problematic when building the hypercube of size  $2\epsilon$  to generate the candidate list. An inadequate  $\epsilon$  can include almost all the points inside the cube, thereby not reducing the NN search time.

## 5.4 Experimental Evaluation

In this section we present an experimental comparison between the standard RNN and our new implementation, the fast-RNN. Two groups of datasets have been used. For the first group, we used the ICARO database for computing SIFT (Lowe 1999), PCA-SIFT (Ke and Sukthankar 2004) and SURF (Bay et al. 2006) descriptors from randomly selected images. The second group of datasets includes 5 sets of vectors, with dimensionality between 3 and 128, generated from the normal distribution. Table 5.1 details the datasets used in the experiments.

Following the practise of past research, *e.g.* (Elkan 2003), we measure the performance of an algorithm as the number of distance calculations required. However, fast-RNN may incur overhead to build and update the auxiliary data structure detailed in Section 5.3.3. We also include the speedup of the total execution time ( $t_{speedup} = t_{RNN}/t_{fast-RNN}$ ) to show that the time of the fast-RNN approach is always less than the time required by an standard RNN algorithm.

Dataset	#vectors	Dimensionality
100K-SIFT	100.000	128
100K-PCA-SIFT	100.000	36
100K-SURF-128	100.000	128
100K-SURF-64	100.000	64
100K-SURF-36	100.000	36
100K-SURF-16	100.000	16
50K-3D-NORM	50.000	3
50K-16D-NORM	50.000	16
50K-36D-NORM	50.000	36
50K-64D-NORM	50.000	64
50K-128D-NORM	50.000	128

Table 5.1: Datasets for the experiments.

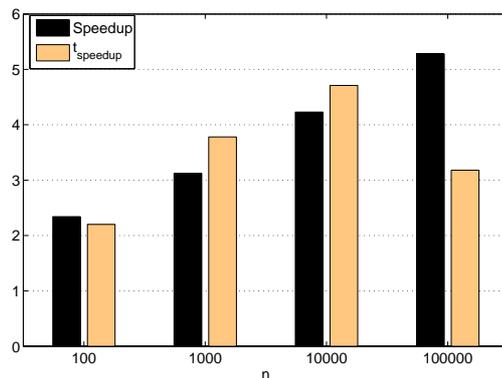
Dataset	Algorithm	$n = 100$	$n = 1000$	$n = 10000$	$n = 100000$
100K-PCA-SIFT	fast-RNN ( $t = 0.6, \epsilon = 0.1$ )	2.76e+03	3.44e+05	3.35e+07	2.81e+09
	RNN ( $t = 0.6$ )	6.47e+03	10.76e+05	14.21e+07	1.49e+10
100K-SIFT	fast-RNN ( $t = 0.9, \epsilon = 0.1$ )	5.20e+03	7.34e+05	1.00e+08	1.13e+10
	fast-RNN ( $t = 0.9, \epsilon = 0.05$ )	4.29e+03	5.66e+05	7.45e+07	–
	RNN ( $t = 0.9$ )	6.44e+03	9.11e+05	1.25e+08	1.44e+10

Table 5.2: Fast-RNN vs. RNN with the 100K-PCA-SIFT and 100K-SIFT sets. The table details the average number of distance calculations for each algorithm.

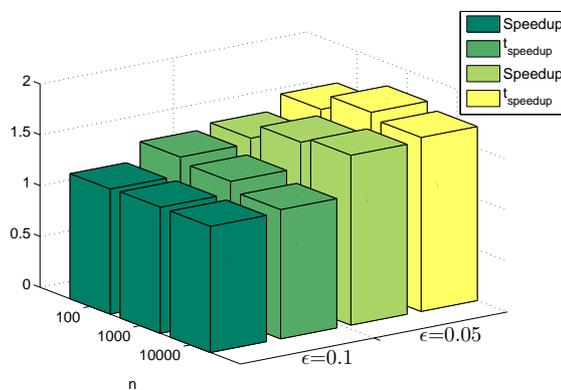
### 5.4.1 Experiments with local descriptors

Table 5.2 shows the results obtained when we use the sets 100K-PCA-SIFT and 100K-SIFT. We report the average number of distance calculations after 10 executions of the clustering algorithms on these datasets while the size of the set of points increases from  $n = 100$  to  $n = 100000$ . With both PCA-SIFT and SIFT descriptors, the results show that the number of distance calculations decreases when using the fast-RNN algorithm.

In Figure 5.8, the speedup shows how many times faster the new algorithm is, when the unit of measurement is distance calculations. Figure 5.8(a) shows the results obtained when using PCA-SIFT descriptors. Both the speedup and the  $t_{speedup}$  are always greater than 1 when  $\epsilon = 0.1$ . One can see how the speedup increases with  $n$ . The  $t_{speedup}$  also increases with  $n$ , but only until  $n = 10000$ . For  $n = 100000$ ,  $t_{speedup}$  drastically decreases to



(a) PCA-SIFT



(b) SIFT

Figure 5.8: Speedup for SIFT and PCA-SIFT descriptors.

3.18. Figure 5.8(b) shows the results for SIFT descriptors and two different values of  $\epsilon$ , 0.1 and 0.05. When  $\epsilon = 0.1$  the speedup monotonically increases with  $n$ , but  $t_{speedup}$  monotonically decreases. However, when  $\epsilon = 0.05$  both the speedup and  $t_{speedup}$  augment. Furthermore, the  $t_{speedup}$  does not monotonically decrease with  $n$ .

The very high-dimensionality of SIFT vectors can explain the results obtained. We can observe that the speedup is lower for SIFT descriptors than for PCA-SIFT vectors. All SIFT vectors are concentrated (recall the distance concentration problem described in Section 5.3), hence a more discriminative  $\epsilon$  is needed to considerably reduce the number of distance calculations.

We have also experimented with SURF descriptors with dimensionality between 16 and 128. See Table 5.3 for the results. The number of distance

Dataset	Algorithm	$n = 100$	$n = 1000$	$n = 10000$	$n = 100000$
100K-SURF-16	fast-RNN ( $t = 0.6, \epsilon = 0.1$ )	6.48e+03	6.63e+05	6.44e+07	6.25e+09
	fast-RNN ( $t = 0.6, \epsilon = 0.05$ )	5.20e+03	3.79e+05	3.62e+07	–
	RNN ( $t = 0.6$ )	13.60e+03	1.47e+06	1.49e+08	1.49e+10
100K-SURF-36	fast-RNN ( $t = 0.6, \epsilon = 0.1$ )	8.17e+03	8.30e+05	8.29e+07	8.27e+09
	fast-RNN ( $t = 0.6, \epsilon = 0.05$ )	5.27e+03	5.50e+05	4.92e+07	–
	RNN ( $t = 0.6$ )	1.34e+04	1.47e+06	1.49e+08	1.49e+10
100K-SURF-64	fast-RNN ( $t = 0.6, \epsilon = 0.1$ )	9.36e+03	1.05e+06	1.08e+08	1.07e+10
	fast-RNN ( $t = 0.6, \epsilon = 0.05$ )	7.53e+03	7.45e+05	7.14e+07	–
	RNN ( $t = 0.6$ )	1.20e+04	1.44e+06	1.49e+08	1.49e+10
100K-SURF-128	fast-RNN ( $t = 0.6, \epsilon = 0.1$ )	4.45e+03	8.18e+05	1.02e+08	1.11e+10
	fast-RNN ( $t = 0.6, \epsilon = 0.05$ )	6.78e+03	7.89e+05	7.98e+07	–
	RNN ( $t = 0.6$ )	5.47e+03	1.02e+06	1.33e+08	1.45e+10

Table 5.3: Fast-RNN vs. RNN with the different SURF datasets.  $n$  is the number of vectors. The table details the average number of distance calculations for each algorithm.

calculations decreases when using the fast-RNN algorithm.

Moreover, Figures 5.9(a) and 5.9(b) show that the speedup increases with  $n$  for all dimensionalities and for the two different values of  $\epsilon$ . In Figures 5.9(c) and 5.9(d), one can see that  $t_{speedup}$  depends on  $\epsilon$ . Specifically, for SURF descriptors of dimensionality 128 and 64 we should use a more restrictive  $\epsilon$  to try to reduce the number of distance calculations. For high dimensions,  $t_{speedup}$  only increases when small values of  $\epsilon$  are used.

### 5.4.2 Normally Distributed Random Datasets

Table 5.4 details the results obtained with 5 datasets of random numbers generated from the normal distribution. Again, the fast-RNN implementation obtains better results than the standard RNN algorithm, *i.e.*  $speedup > 1$  and  $t_{speedup} > 1$ . With these datasets, bigger values of  $\epsilon$  obtain better results than with the SIFT and SURF descriptors.

### 5.4.3 Determining the best $\epsilon$

As we have seen, the speedup of the proposed algorithm depends critically on  $\epsilon$ . We have measured the speedup for PCA-SIFT, SIFT and SURF descriptor varying  $\epsilon$  between 0.001 and 0.1. Figure 5.10 shows the results obtained. For PCA-SIFT descriptors, with dimensionality 36, the speedup we obtain does not depend critically on  $\epsilon$  within the interval  $\epsilon \in [0.001, 0.1]$ . However,

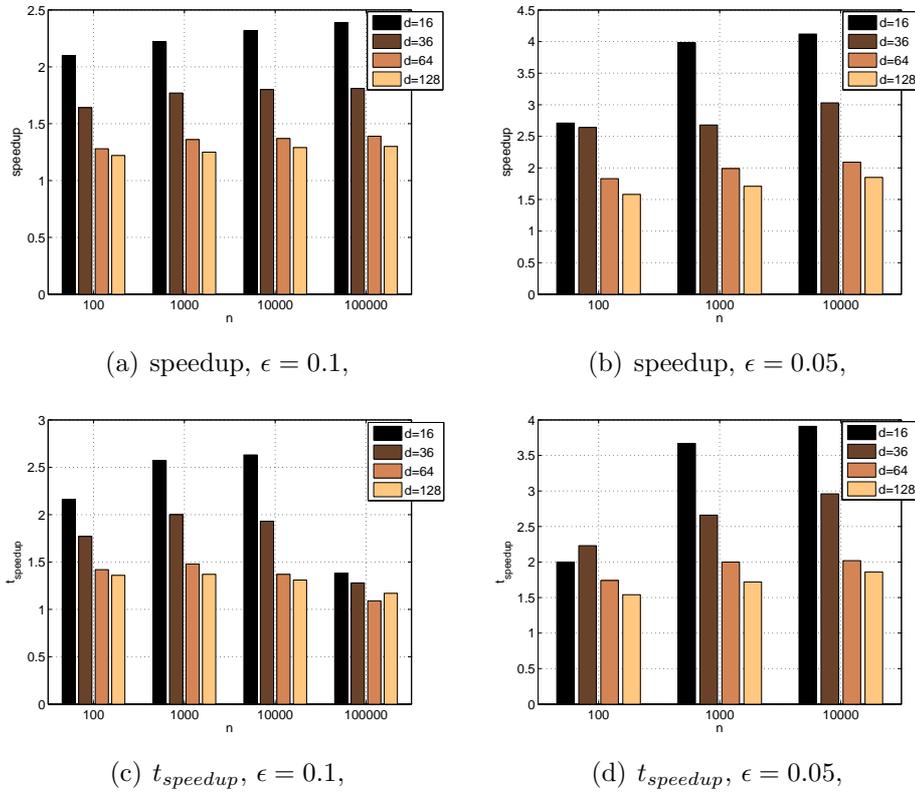


Figure 5.9: Speedup for SURF descriptors.

Dataset		$n = 100$	$n = 1000$	$n = 10000$	$n = 50000$	Parameters
50K-3D-NORM	Speedup	2.96	3.15	4.19	4.15	$t = 0.4, \epsilon = 0.1$
	$t_{speedup}$	1.95	3.41	4.94	5.91	
50K-16D-NORM	Speedup	2.51	2.37	2.44	2.60	$t = 1.0, \epsilon = 0.1$
	$t_{speedup}$	1.89	2.61	2.79	2.30	
50K-36D-NORM	Speedup	2.07	2.12	2.66	2.44	$t = 1.1, \epsilon = 0.1$
	$t_{speedup}$	1.73	2.29	2.69	2.11	
50K-64D-NORM	Speedup	2.07	2.27	1.71	1.83	$t = 1.14, \epsilon = 0.1$
	$t_{speedup}$	1.84	1.78	1.76	1.80	
50K-128D-NORM	Speedup	1.35	1.75	1.45	1.79	$t = 1.16, \epsilon = 0.1$
	$t_{speedup}$	1.42	1.9	1.49	1.64	

Table 5.4: Fast-RNN vs. RNN with the normally distributed datasets.

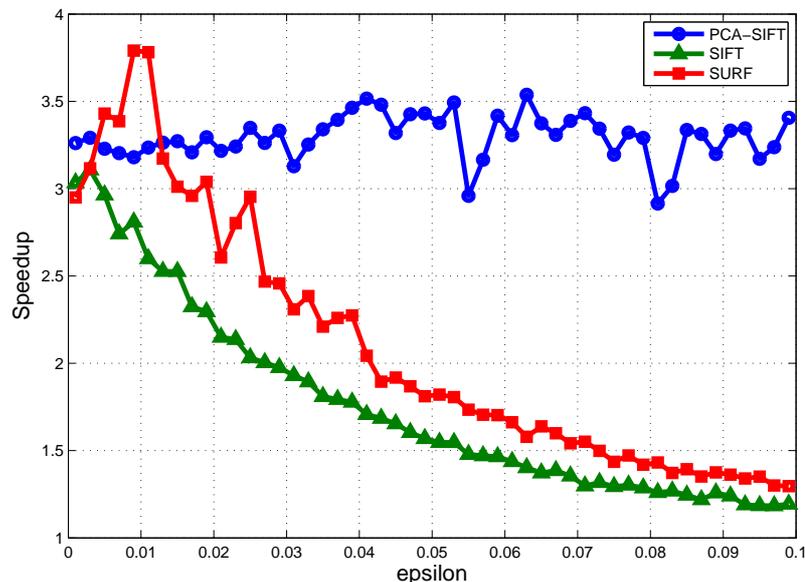


Figure 5.10: Epsilon vs. Speedup for PCA-SIFT, SIFT and SURF descriptors.

for SURF descriptors we obtain the maximum speedup when  $\epsilon = 0.01$ . Figure 5.10 also shows that for both SIFT and SURF descriptors the speedup monotonically decreases with  $\epsilon$ .

## 5.5 Conclusion

This chapter provides a detailed description of the implementation of the fast-RNN clustering algorithm. Our novel approach is based on the projection paradigm for accelerating the search for NN when we build NN chains. We have analysed how to deal with the problem efficiently when our data lie in a high-dimensional space. The results show how the fast-RNN is faster than the standard approach when an adequate value of  $\epsilon$  is chosen. We would like to emphasise that the algorithm presented does not provide an approximate solution; the solution obtained by a standard RNN and this new fast-RNN are identical. Furthermore, with the aim of making our research reproducible, both the set of descriptors and a C++ implementation of the fast-RNN clustering are publicly available and can be downloaded from

<http://agamenon.tsc.uah.es/Personales/rlopez/data/fastrnn>.

# Chapter 6

## Aggregating Visual Words

*“And now, gentlemen,” said d’Artagnan, without stopping to explain his conduct to Porthos, “All for one, one for all—that is our motto, is it not?”*

Alexandre Dumas, *The Three Musketeers*.

---

Most recent category-level object recognition systems work with visual words, *i.e.* vector quantised local descriptors. These visual vocabularies are usually constructed by using a single method such as  $K$ -means for clustering the descriptor vectors of patches sampled either densely or sparsely from a set of training images. Instead, in this chapter we propose a novel methodology for creating efficient codebooks for visual recognition using clustering aggregation techniques. Our model aims to increase the stability of the visual vocabulary construction process by combining several clusterings. A rigorous approach for incorporating meaningful spatial coherency among the local features into the visual vocabulary construction is described too. Results on image classification are presented on the PASCAL VOC Challenge 2007 and 2009 datasets.

### 6.1 Introduction

The BoW approach (Csurka et al. 2004, Sivic and Zisserman 2003) is a popular strategy for representing images within the context of category-level object recognition. The basic idea behind this type of representation is to characterise an image by the histogram of its visual words, *i.e.* vector quantised local features. Popular candidates for these local features are local

descriptors (Mikolajczyk and Schmid 2005) that can be extracted either at specific interest points (Csurka et al. 2004, Fergus et al. 2003, Leibe and Schiele 2003) or densely sampled over the image (Jurie and Triggs 2005, Perronnin et al. 2006).

There are various clustering methods for creating these visual words.  $K$ -means, approximate  $K$ -means (Philbin et al. 2007) and vocabulary trees (Nister and Stewenius 2006) are currently the most common.

Subsequently, each local feature in an image is mapped to a cluster so as to represent any image as a histogram over the clusters. Such a representation has been shown to characterise the images and objects within them in a robust yet descriptive manner, in spite of the fact that it ignores the spatial configuration between visual words. Moreover, these Bag-of-Words (BoW) systems have shown impressive results lately (van de Sande et al. 2008, Zhang et al. 2007). Variations on this scheme won the recent PASCAL VOC Challenges on object classification (Everingham et al. 2007, 2008).

Although such ideas appear to be quite exciting, there are 2 main challenges that need to be overcome. Since the clustering into visual words is unsupervised, this representation often does not group semantically meaningful object parts (*e.g.* wheels or eyes). That is, visual words tend to be much more ambiguous than texts. In practise, if the dataset is sufficiently coherent (*e.g.* images of only one particular class), only a reduced number of visual words represent semantic object parts. Moreover, when an unsupervised quantisation is applied to a more diverse dataset, synonyms and polysemies are the norm rather than the exception (Quelhas et al. 2005, Yuan et al. 2007). Typically, the spatial context of the local features is lost during the visual vocabulary construction, *i.e.* the clustering algorithms ignore the semantic relationship between local features that normally co-occur.

On the other hand, there are the limitations of the clustering algorithms themselves. In general, data clustering has usually associated the stability problem:

- the absence of ground truth, against which the clustering result can be validated, makes impossible to use cross validation for tuning the clustering parameters.
- the dependence on the initialisation is a common problem of most of the iterative methods.
- the objectives that each clustering approach pursues are different, so very different structures in data may be discovered with different algorithms.

Specifically,  $K$ -means clustering output depends on the initialisation as the procedure only undertakes the search for a local optimum and it requires the user to specify the number of clusters in advance. Moreover, there is no guarantee that the clusters obtained are visually compact and it is computationally expensive for big values of  $K$ .

Other approaches use efficient hierarchical clustering schemes (Leibe et al. 2006) where it is possible to fix a cut-off threshold on the cluster compactness. This threshold determines the number of clusters by successively merging clusters until it is reached. However, it may happen that some *real* clusters are split in several clusters, so that the visual words are not representative of all features. Furthermore, both the runtime and the memory requirements are often significantly higher for hierarchical methods.

The contribution this chapter makes is twofold. First, we introduce a new framework for obtaining visual words that tries to overcome the problem of clustering stability. We propose to apply the clustering aggregation techniques (Gionis et al. 2007) to the visual vocabulary construction process. The problem of clustering aggregation can be formulated as follows: given a set of clusterings that have been obtained for a particular data set, find a clustering that agrees as much as possible with the given clusterings. To the best of our knowledge, this is the first work to describe a clustering aggregation approach within this context. We analyse how these techniques perform in discovering visual words using different combinations of quantisation algorithms.

Second, we incorporate meaningful spatial coherency among the local descriptors into the visual vocabulary to narrow the semantic gap. The basic idea behind our approach is that local features in neighboring regions with similar appearance/colour should be characterised by the same visual word. In short, our approach consist of the following steps. We first quantise the dense local descriptors following traditional clustering algorithms such as  $K$ -means. Additionally, we quantise the position and appearance of these local features by using regular grids and over-segmenting the images respectively. For the latter, we first discretise the colour space using a regular lattice. Each local feature is assigned to the bin representing the average colour of the region it belongs to. These regions are generated by the segmentation algorithms. These additional quantisations are added to the input of the clustering aggregation approach hence incorporate the semantic information into the vocabulary construction process.

The rest of this chapter is organised as follows. First, we introduce the clustering aggregation theory (Section 6.2). In Section 6.3 we give a detailed description of the framework we propose for integrating the clustering aggregation techniques into the visual vocabulary construction process. Ex-

periments in image categorisation are described in Section 6.4 and Section 6.5 concludes the chapter.

## 6.2 Clustering Aggregation

The problem of clustering aggregation has been considered under a variety of names (consensus clustering, clustering combination, clustering ensemble). Many approaches have been proposed, *e.g.*, the graph cut method (Fern and Brodley 2004), the information theoretic method (Strehl and Ghosh 2002) and the Bayesian method (Wang et al. 2009). Recently, clustering aggregation has been widely applied in many areas such as bio-informatics (Hu and Yoo 2004) and computer vision (Yu et al. 2008).

Clustering aggregation is defined as the optimisation problem where, given a set of  $m$  clusterings, we want to find the clustering that minimises the total number of disagreements with the  $m$  clusterings. Clustering aggregation can be considered as a metaclustering method to improve stability and robustness of clustering by combining the results of many clusterers. Moreover, it can determine the appropriate number of clusters while detects outliers.

Gionis et al. (2007) propose an approach to this problem based on the concept of aggregation. We are given a set of  $m$  clusterings  $C_1, C_2, \dots, C_m$ . Our objective is to obtain a single clustering  $C$  that agrees as much as possible with the  $m$  input clusterings. It is possible to define a disagreement between two clusterings  $C_i$  and  $C_j$  as a pair of objects  $u$  and  $v$  such that  $C_i$  places them in the same cluster, while  $C_j$  places them in different clusters or vice versa. If  $d(C_i, C_j)$  defines the number of disagreements between  $C_i$  and  $C_j$ , then the objective is to find a clustering  $C$  that minimises  $\sum_{i=1}^m d(C_i, C)$ . Figure 6.1 shows an example of clustering aggregation. The rightmost column is the clustering  $C$  where the number of disagreements with the clusterings  $C_1, C_2$  and  $C_3$  has been minimised.

The clustering aggregation approach described by Gionis et al. (2007) is related to a problem which is known as correlation clustering (Bansal et al. 2004). See Section 4.3.1 for further details.

Before we go into the details of clustering aggregation techniques, some notations will be introduced. Consider a set of  $n$  objects  $V = \{v_1, v_2, \dots, v_n\}$ . A clustering  $C_i$  of  $V$  is a partition of  $V$  into  $k$  disjoint sets  $\{S_1, S_2, \dots, S_k\}$ , *i.e.*  $\bigcup_i^k S_i = V$  and  $S_i \cap S_j = \emptyset$  for all  $i \neq j$ . The clusters of  $C_i$  are the  $k$  sets  $S_1, S_2, \dots, S_k$ . For each  $v \in V$ , we use  $C_i(v)$  to denote the label of the cluster to which the object  $v$  belongs to, *i.e.*  $C(v) = j$  if and only if  $v \in S_j$ . The task of the clustering aggregation is to find a clustering that minimises

	$C_1$	$C_2$	$C_3$	$C$
$v_1$	1	1	1	<b>1</b>
$v_2$	1	2	2	<b>2</b>
$v_3$	2	1	1	<b>1</b>
$v_4$	2	2	2	<b>2</b>
$v_5$	3	3	3	<b>3</b>
$v_6$	3	4	3	<b>3</b>

Figure 6.1: Example of clustering aggregation following the approach of Gionis et al. (2007). Consider the dataset of 6 objects  $\{v_1, v_2, \dots, v_6\}$ . Each column corresponds to a clustering, and a value  $i$  denotes that the object in that row belongs in the  $i$ -th cluster of the clustering in that column.  $C$  is the clustering that minimises the total number of disagreements with the clusterings  $C_1, C_2$  and  $C_3$ .

the disagreements with a set of input clusterings.

Formally, the clustering aggregation can be defined as follows. Given a set of objects  $V$  and  $m$  clusterings  $C_1, C_2, \dots, C_m$ , the distance between two objects  $u$  and  $v$  is defined as

$$d(u, v) = \frac{1}{m} |\{i \mid 1 \leq i \leq m \text{ and } C_i(u) \neq C_i(v)\}|, \quad (6.1)$$

*i.e.*  $d(u, v)$  is the fraction of clusterings that assign the pair  $(u, v)$  into different clusters. The clustering aggregation has to find a partition  $C$  that minimises the function

$$d(C) = \sum_{C(u)=C(v)} d(u, v) + \sum_{C(u) \neq C(v)} (1 - d(u, v)). \quad (6.2)$$

For a candidate solution  $C$ , if  $C$  places  $u$  and  $v$  in the same cluster, it will disagree with  $md(u, v)$  of the original clusterings, whilst if  $C$  places  $u$  and  $v$  in different clusters, it will disagree with the remaining  $m(1 - d(u, v))$  clusterings. Gionis et al. (2007) demonstrate that the distance measure  $d(u, v)$  satisfies the triangle inequality. A toy example to illustrate how the clustering aggregation works is depicted in Figure 6.2.

Several algorithms are described to solve the problem of clustering aggregation, we refer to (Gionis et al. 2007) for further details. It is worth emphasising that most algorithms are parameter-free, therefore, we do not need to specify the number of clusters. However, within the context of visual codebooks, we have to apply vector quantisation methods for extremely large

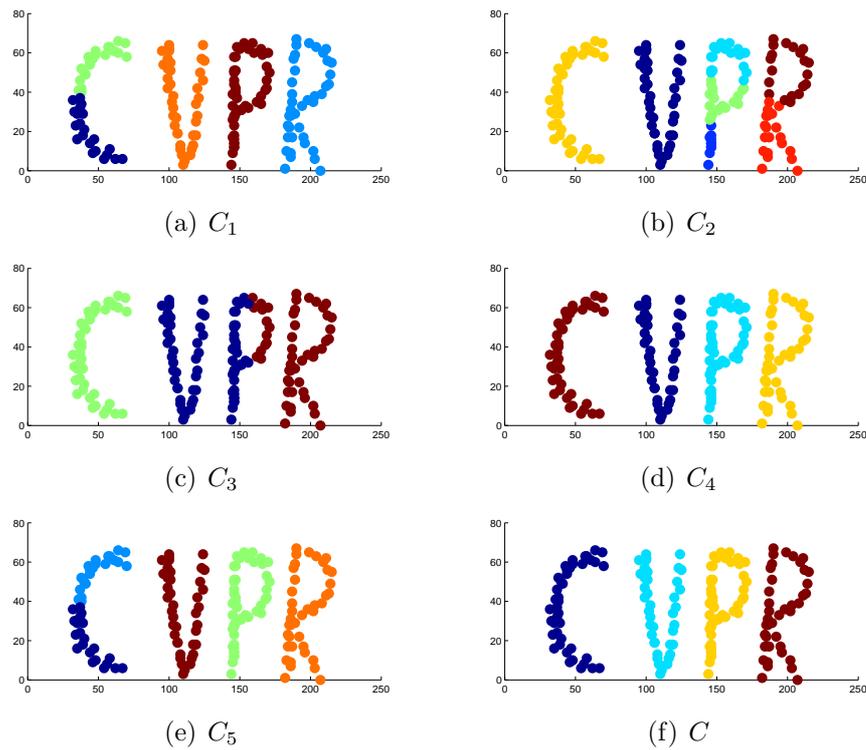


Figure 6.2: Toy example.(a)-(e) are five clustering  $C_1, C_2, \dots, C_5$  over the CVPR dataset of two dimensional points. (f) depicts the results of the clustering aggregation algorithm. We have used different colours to denote different clusters.

sets of vectors in high dimensional spaces. The quadratic complexity is inherited in the correlation clustering problem since a complete graph is the input to the problem. This makes the algorithms inapplicable to large datasets. Gionis et al. (2007) present a sampling algorithm to overcome this problem. The algorithm samples a set of nodes,  $S$ , from the dataset. This set is the input for the clustering aggregation algorithm. Once the aggregation has finished, the algorithm goes through the nodes not in  $S$  and decides whether to place it on one of the existing clusters or to create a singleton. In our case this sampling algorithm is not enough due to the high complexity of the aggregation algorithm. In order to achieve more reduction in the running time, we do not apply the sampling over the entire set of objects, but we perform first a uniformly and random sampling of objects candidates which are inserted in the set  $R$ . This set is the input for the sampling algorithm described before. In the post-processing, we inspect the nodes not in  $R$  and assign them to the nearest cluster.

## 6.3 Visual Words Aggregation

Before describing our approach, we discuss why clustering aggregation techniques can be useful for large sets of vectors in high dimensional spaces.

Such spaces are extremely sparse with the data points far away from each other. Furthermore, the norm used to define the distance has the strange property to concentrate (Francois et al. 2007). As a consequence, all pairwise distances in a high-dimensional data set seem to be equal or at least very similar. This may lead to problems when searching for clusters. The phenomenon is known in the statistical literature as the *curse of dimensionality*.  $K$ -means is a popular algorithm for its simplicity. Unfortunately, centres tend towards denser regions, with the result that they tend to be tightly clustered near dense regions and sparsely spread in sparse ones. Furthermore, if these high dimensional descriptors have been densely extracted, there do not exist separate clusters. Mean-shift based approaches (*e.g.* (Jurie and Triggs 2005)) can be used to overcome the limitations of  $K$ -means. Moreover, agglomerative clustering approaches (*e.g.* (Leibe et al. 2006)) handle the unbalanced density problem well, but they can not be applied to large dataset due to their algorithmic complexity.

Within this context, the clustering aggregation techniques let us to increase the stability of the visual vocabulary construction process. Furthermore, they are able to combine the properties from different clustering algorithms, which is something desirable in such a high-dimensional space where the local descriptors reside.

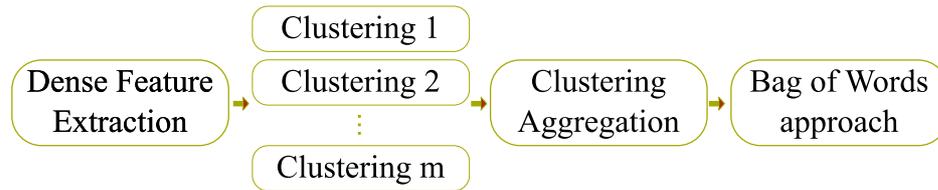


Figure 6.3: Flowchart of our approach for constructing a visual vocabulary using clustering aggregation.

### 6.3.1 Simple Aggregation

Figure 6.3 depicts the major steps for constructing a visual vocabulary using clustering aggregation.

Images are represented using local features. To extract these, we experiment with a dense sampling of images patches using a regular grid. However, it is also possible to extract local features using scale-invariant feature detectors. As a first step of our algorithm, we describe the content of each patch with a local descriptor (*e.g.* SIFT (Lowe 1999)). Then the vector quantisation process starts. A total of  $m$  clustering algorithms are executed. We experiment with different algorithms and with different runs of the same one so as to increase the stability of the solution. The clustering algorithms used in the experiments are  $K$ -means, RNN (Leibe et al. 2006) and the mean-shift based algorithm proposed by Jurie and Triggs (2005). Once these clusterings have been obtained we proceed with the clustering aggregation step. Using the sampling strategy to handle large datasets, we let the aggregation clustering converges into a final codebook. The resulting features are collected to form a BoW image representation.

### 6.3.2 Incorporating Semantic Information

Clustering aggregation approach is able to incorporate spatial coherency among the local descriptors into the visual vocabulary. Figure 6.3 shows that before the clustering aggregation step, several codebooks have to be obtained. These codebooks are clusterings that organise the local descriptors associated to the local features. It is also possible to cluster these local features in the image and colour space, *i.e.* to cluster features that are near in the image and that belong to image regions with similar average colour (or similar appearance). In addition to the codebooks obtained in the descriptors domain, the spatial and colour quantisation can be introduced as inputs in the clustering aggregation process. This will lead the aggregation process

to obtain more semantic vocabularies.

### 6.3.2.1 Via Regular Grids

This first approach starts with the extraction of local features, either at interest points or densely sampled. We proceed with the local descriptors as it was described in Section 6.3.1, *i.e.* vector quantising them with traditional clustering algorithms. The objective of this first step is to obtain  $n$  different codebooks  $C_i$  where  $i = 1 \dots n$ . See Figure 6.4 upper box.

The second step has been designed with the aim of incorporating into a traditional clustering aggregation approach a factor to lead the algorithm to merge those visual words that are near in the image and not only in the descriptor high-dimensional space. So as to get this, a simple approach is to superimpose a grid over all the images in the database (all the images in the database have been previously scaled to the same size). Each cell can be considered a cluster. So a grid of  $r \times c$  cells defines a clustering of size  $r \times c$  where the local features are quantised. Furthermore, it is possible to use grids of either random or fixed cell sizes. Within this context, we assign to each feature the label of the cell where it falls. Then, if  $l$  different grids are used, we obtain  $l$  different local quantisations  $CL_i$  where  $i = 1 \dots l$ .

Figure 6.4 depicts how we incorporate the regular grids to the clustering aggregation process. The  $n$  codebooks  $\{C_1, C_2, \dots, C_n\}$  plus the  $l$  local quantisations  $\{CL_1, CL_2, \dots, CL_l\}$  are the input of the clustering aggregation algorithm.

### 6.3.2.2 Via over-segmentation

Unfortunately, using regular grids we do not often group image regions of homogeneous appearance, so the local features may not be in clusters with semantic coherency. For instance, in an image with different object instances at very different scales a simple regular grid fails to semantically group the local features.

In order to assess this issue, we propose a second approach inspired by Russell et al. (2006). We first perform an over-segmentation of an image by partitioning it into multiple homogeneous regions. We and Russell et al. (2006) use an image segmentation process to group related visual words. However, our segmentation step plays a fundamentally different role. In (Russell et al. 2006) each segment is assumed to be a potential object in its entirety. Our representation searches for over-segmented regions than can group local features which belong to semantic object parts, as Cao and Fei-Fei (2007) do. Furthermore, our approach is not tied to a specific segmentation algo-

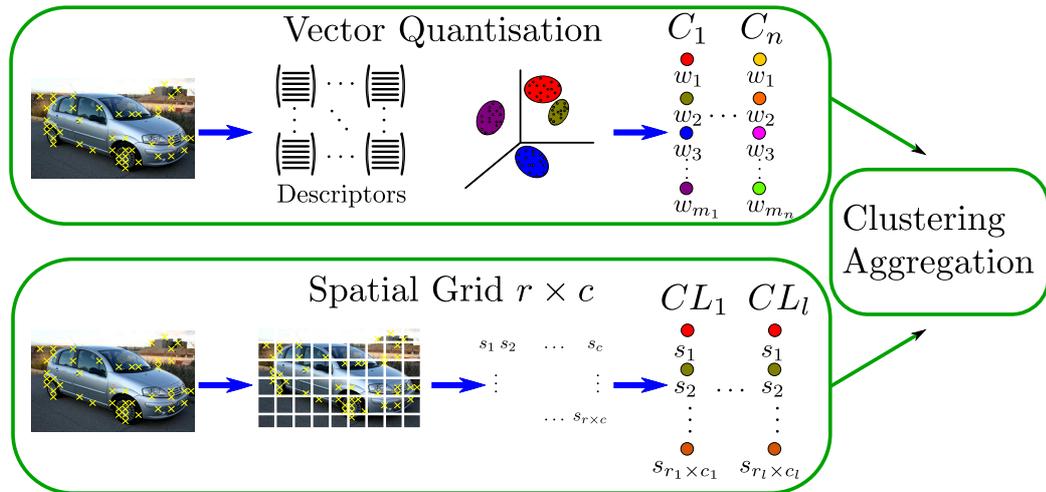


Figure 6.4: Using regular grids to add spatial information to the vocabulary construction process.

rithm. We use the segmentation algorithm described in (Felzenszwalb and Huttenlocher 2004a). We choose colour (in RGB space) to describe the appearance of a region. Figure 6.5 shows one example of our over-segmentation approach. Note that we let the segmentation discover regions that identify *semantic* parts.

The first step of this second approach coincides with the first step described in Section 6.3.2.1. So, to build  $n$  different codebooks  $C_i$  where  $i = 1 \dots n$  is the first objective. See Figure 6.6 upper box. The second step implies to over-segment the images to establish regions of neighbouring appear-

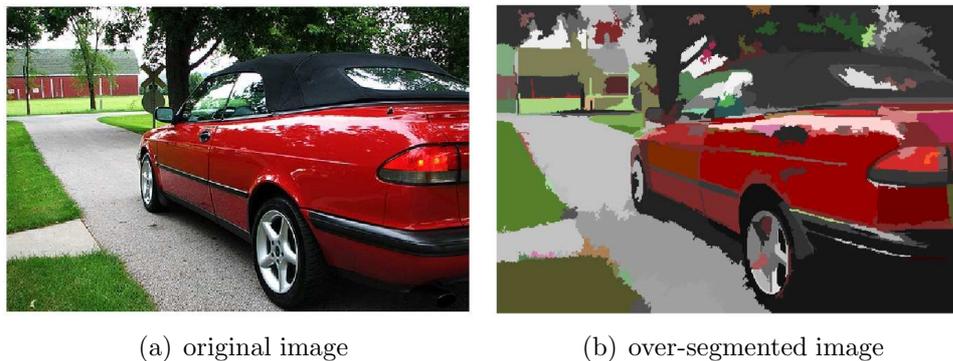


Figure 6.5: Image over-segmented.

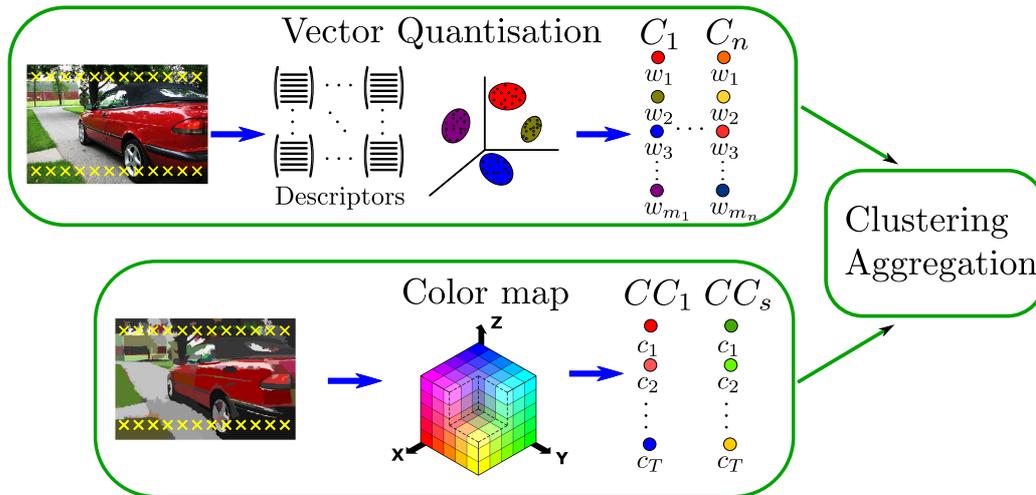


Figure 6.6: Using over-segmentation for adding spatial/semantic information to the vocabulary construction process.

ances/colours. We then discretise the colour space using a regular lattice of  $T$  bins, *i.e.* a colour map of  $T$  values, and each local feature is assigned to the bin representing the average colour of the region it belongs to. So, if  $s$  different segmentations are applied, we obtain the set of colour quantisation  $\{CC_1, CC_2, \dots, CC_s\}$ . The  $n$  codebooks  $\{C_1, C_2, \dots, C_n\}$  plus the  $s$  colour quantisations  $\{CC_1, CC_2, \dots, CC_s\}$  are the input of the clustering aggregation algorithm.

Within the described approach it is also possible to use different colour maps and different colour spaces. That is how any obtained appearance quantisation can be incorporated into the clustering aggregation algorithm.

## 6.4 Results

### 6.4.1 Experimental Setup

Our aim is to evaluate within the context of image classification the performance of the different visual vocabulary construction approaches proposed in this chapter. So as to obtain reliable results, we use the PASCAL VOC Challenge 2007 and 2009 datasets (Everingham et al. 2007, 2009b). We emphasise that this challenge is widely acknowledged as a difficult testbed for both object detection and image categorisation.

For image representation, we follow the procedure described by Lazebnik

et al. (2006). Specifically, we use SIFT (Lowe 1999) descriptors of  $16 \times 16$  pixel patches computed over a grid with spacing of 8 pixels. With these descriptors we perform the vocabulary construction using the clustering aggregation techniques described in Section 6.3. Specifically, we use the *BALLS* algorithm for clustering aggregation described in (Gionis et al. 2007) due to its low time and space complexity. The sampling algorithm is run over a subset of the local descriptors. Finally, once the clustering aggregation obtains the codebook, each image is represented by a spatial pyramid. Typical pyramids levels ( $L$ ) values for our experiments are  $L = 2, 3$ .

We use SVM for classification. The decision function of an SVM classifier for a test sample with feature vector  $\mathbf{x}$  has the form

$$g(\mathbf{x}) = \sum_i \alpha_i y_i k(\mathbf{x}_i, \mathbf{x}) - b, \quad (6.3)$$

where  $y_i$  is the class label of  $\mathbf{x}_i$  ( $-1$  or  $+1$ ),  $\alpha_i$  is the learnt weight of train sample  $\mathbf{x}_i$ ,  $b$  is a learnt threshold parameter and  $k(\mathbf{x}_i, \mathbf{x})$  is the value of a kernel function. We experiment with a kernel function which has shown good results in object recognition: the HIK (Maji et al. 2008). The HIK applied to two feature vectors  $\mathbf{x}$  and  $\mathbf{x}'$  of dimension  $D$  is defined as

$$k(\mathbf{x}, \mathbf{x}') = \sum_{i=1}^D \min(\mathbf{x}(i), \mathbf{x}'(i)). \quad (6.4)$$

Specifically, we use libSVM (Chang and Lin 2001) and the built-in one-versus-one approach for multi-class classification. A 10-fold cross-validation on the *trainval* set (of both the PASCAL VOC challenge 2007 and 2009 datasets) to tune SVM parameters is conducted to train each classifier.

### 6.4.2 Evaluation criteria

We closely follow the image classification evaluation procedure proposed by Everingham et al. (2009a) using the precision/recall curve for each class. Recall is defined as the proportion of all positive examples ranked above a given rank. Precision is the proportion of all examples above that rank which are from a positive class. The interpolated average precision  $AP$  is measured as the mean precision in a set of 11 equally spaced recall levels ( $[0, 0.1, \dots, 1]$ ):

$$AP = \frac{1}{11} \sum_{r \in \{0, 0.1, \dots, 1\}} p_{\text{interp}}(r). \quad (6.5)$$

The interpolated precision at each recall level  $r$ , *i.e.*  $p_{\text{interp}}(r)$ , is defined as the highest precision found for any recall level  $r' \geq r$  (for a method)

$$p_{\text{interp}}(r) = \max_{r' \geq r} p(r'), \quad (6.6)$$

where  $p(r')$  is the measured precision at recall  $r'$ .

When performing experiments over multiple object classes, the average precisions of the individual classes can be aggregated. This aggregation is called Mean Average Precision *MAP*, which is computed by taking the mean of the average precisions.

### 6.4.3 Codebooks Performance in Image Classification

#### 6.4.3.1 PASCAL VOC Challenge 2007 dataset

The PASCAL VOC Challenge 2007 dataset contains 9,963 annotated images, with the number of annotated objects being 24,640. In the experiments we select the *trainval* and *test* set for training and testing the classifier respectively. For further details we refer to Chapter 3 and (Everingham et al. 2007).

**Simple Aggregation** Based on the visual vocabulary construction using clustering aggregation over different codebooks (Section 6.3.1) we have performed the experiments summarised in Table 6.1. The results per object category are shown in Figure 6.7. The *MAP* for each experiment is presented in Table 6.2. The experiment CA-5, where *K*-means and Jurie and Triggs (2005) clusterings are aggregated, obtains the best results for image classification. Moreover, a clustering aggregation with several executions of RNN over SIFT descriptors decreases the average precision in image categorisation. Experiment CA-2 shows that aggregating different runs of *K*-means algorithm, with *K* fixed to 200, does not increase the average precision, *i.e.* it seems that the problem of stability of *K*-means codebooks does not affect too much the performance of the SVM classifiers. Furthermore, experiment CA-7 shows that when we vary *K* from 200 to 600 and aggregate the resulting *K*-means codebooks, the performance decreases.

**Incorporating Semantic Information** We have experimented with the incorporation into the clustering aggregation of spatial relations among local features. Table 6.3 summarises the different combinations of clusterings and spatial and colour quantisations we have used in the experiments. The reported results have been obtained building a pyramid of 3 levels. From

Experiment	Description
CA-1	Single $K$ -means clustering ( $K = 200$ )
CA-2	Aggregation of 5 $K$ -means clusterings ( $K = 200$ for all runs)
CA-3	Single RNN clustering (Leibe et al. 2006) ( $t = 0.4$ )
CA-4	Aggregation of 5 RNN clusterings (Leibe et al. 2006) ( $t = 0.4$ for all runs)
CA-5	Aggregation of 2 different clusterings: $K$ -means ( $K = 400$ ) and Jurie and Triggs (2005) ( $r = 0.8, N = 3000$ )
CA-6	Single clustering Jurie and Triggs (2005) ( $r = 0.8, N = 3000$ )
CA-7	Aggregation of 5 $K$ -means clusterings ( $K = 200, 400, 600$ and 800)

Table 6.1: Experiments descriptions for simple clustering aggregation.

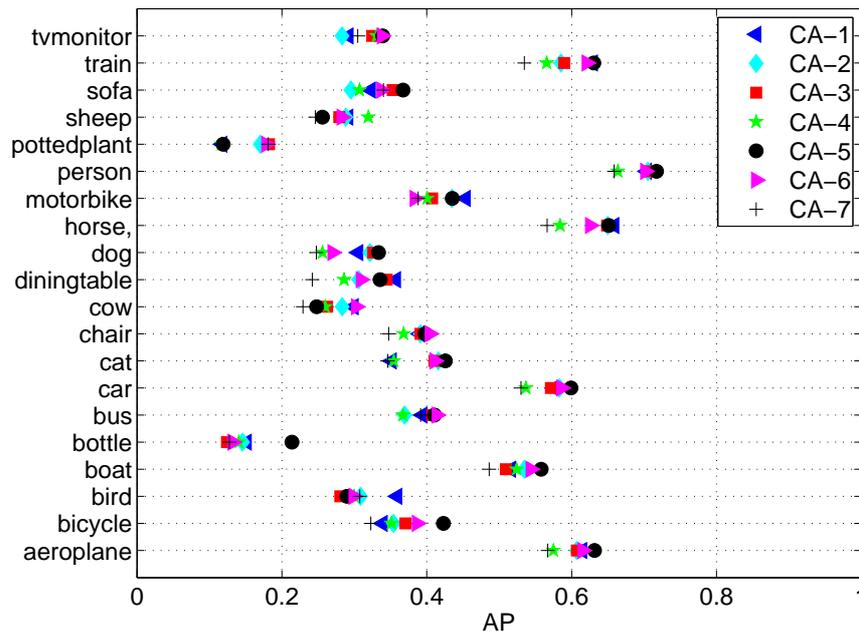


Figure 6.7: Evaluation of codebooks on image categorisation over the PASCAL VOC 2007 Challenge. Average precision per class for each method is shown.

	CA-1	CA-2	CA-3	CA-4	CA-5	CA-6	CA-7
<i>MAP</i>	40.60	40.16	40.48	38.03	<b>41.91</b>	40.71	36.8

Table 6.2: *MAP* obtained for the experiments CA-*i*.

Clustering	Parameters	SCA-1	SCA-2	SCA-3	SCA-4	SCA-5	SCA-6
<i>K</i> -means	$K = 200$	✓	✓	✓			
Jurie and Triggs (2005)	$(r, N) = (0.8, 2000)$				✓	✓	✓
Fixed Grid	$16 \times 8$	✓		✓	✓		✓
Over Segmentation	$(\sigma, k) = (0.5, 300)$ $mcs = 100$ Colourmap = 500		✓	✓		✓	✓

Table 6.3: Clustering combinations for the experiments SCA-*i*. See (Felzenszwalb and Huttenlocher 2004a) for a more detailed description of the parameters  $\sigma$ ,  $k$  and  $mcs$ .  $mcs$  stands for minimum component size.

the results in Table 6.4, it is observed that the combination of the clustering algorithm of Jurie and Triggs (2005) with the codebook obtained through the over-segmentation of images (*i.e.* experiment SCA-5) performs better than the rest of combinations. On the other hand, *K*-means clusterings based approaches (*i.e.* experiments SCA-1, SCA-2 and SCA-3) do perform slightly worse. Furthermore, they have not been benefited from the incorporation of the over-segmentation of images into the clustering aggregation.

**Discussion** In general, the results reveal that *K*-means codebooks do not satisfactorily aggregate. That is the average precision does not increase after the aggregation algorithms. On the other hand, mean-shift based clustering algorithms (*e.g.* Jurie and Triggs (2005)) seem to be more suitable for this type of approach.

The average precisions obtained when we add semantic information have decreased too. One of the reason for this worsening in the performance is that the images in the PASCAL VOC Challenge 2007 dataset present high variability in terms of object size, orientation, pose, illumination, position and occlusion. All of these aspects difficult the incorporation of semantic information into the vocabulary construction process with our approaches. To

	SCA-1	SCA-2	SCA-3	SCA-4	SCA-5	SCA-6
<i>MAP</i>	37.5	34.8	36.3	39.7	<b>39.8</b>	39.4

Table 6.4: *MAP* obtained for for the experiments SCA-*i*.



	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
HP	88.1	68.6	68.1	72.9	44.2	79.5	72.5	70.8	59.5	53.6	57.5	59.3	73.1	72.3	85.3	40.8	56.9	57.9	86.0	68.6
AVW	<b>70.9</b>	<b>39.0</b>	43.1	<b>53.0</b>	14.7	56.3	38.2	<b>47.7</b>	<b>40.2</b>	<b>28.2</b>	<b>29.2</b>	<b>34.5</b>	39.1	<b>39.7</b>	<b>68.5</b>	18.5	<b>33.7</b>	<b>34.7</b>	<b>65.2</b>	<b>44.4</b>
Rank	33	34	33	29	43	32	38	30	37	29	37	33	34	34	36	34	28	31	30	32
LAVW	70.4	34.8	<b>43.4</b>	49.2	<b>16.6</b>	<b>57.3</b>	<b>39.7</b>	45.6	39.7	26.7	22.6	30.3	<b>41.0</b>	39.6	67.5	<b>18.6</b>	32.5	27.3	64.9	42.6
Rank	34	36	32	31	42	30	35	31	38	30	39	39	33	35	38	33	30	33	31	34
LP	6.8	6.0	16.8	6.7	8.9	4.1	12.4	8.5	13.7	2.6	4.7	12.5	5.9	5.4	49.5	6.1	3.4	6.9	7.9	5.6

Table 6.5: PASCAL VOC 2009 Results. Average precision scores of our methods. We also show the rank of our methods in the competition, and the highest and lowest precisions, HP and LP respectively, for each class.

ted two approaches to the competition *comp1*<sup>1</sup>. The first one was labelled as ALCALA\_AVW. In ALCALA\_AVW we developed a simple aggregation of visual words obtained from different runs of the  $K$ -means algorithm over SIFT descriptors. Specifically, we used three  $K$ -means codebooks with  $K=2000$  ( $C_1$ ),  $K=2500$  ( $C_2$ ) and  $K=3000$  ( $C_3$ ). After the clustering aggregation, each image was represented by a spatial pyramid (Lazebnik et al. 2006) of 2 levels.

ALCALA\_LAVW is the label of the second submitted method. This time we used a regular grid to incorporate spatial/semantic information within the vocabulary construction process. Specifically, we first obtained a codebook using a  $K$ -means algorithm ( $K = 2000$ ). Then we superimposed a regular grid of  $16 \times 8$  cells to obtain the second codebook. Again, after the clustering aggregation, each image was represented by a spatial pyramid (Lazebnik et al. 2006) of 2 levels.

Table 6.5 summarises the results for our two submissions. We show the average precision (%) of our methods and their ranks among the 48 submitted methods. The highest (HP) and lowest (LP) average precisions for each class are showed too.

ALCALA\_AVW approach performs better than ALCALA\_LAVW in 14 classes, and obtains a median average precision of 57.7%. Figure 6.9 shows the best result by group, where ALCALA\_AVW holds the position 14 of 20.

Figure 6.10 depicts the maximum average precision, the median precision of all methods for every class, and the average precision of both ALCALA\_AVW and ALCALA\_LAVW. For Precision/Recall curves for each of the 20 classes we refer to Everingham et al. (2009b).

Within the PASCAL VOC Challenge 2009 it was also possible to train on VOC 2009 data and to test on the VOC 2008 images. Table 6.6 shows the average precision for competition *comp1* within this context. Slightly worse results are obtained by all methods with VOC 2008 data. However, Figure

<sup>1</sup>See <http://pascallin.ecs.soton.ac.uk/challenges/VOC/voc2009/results/index.html> for further details.

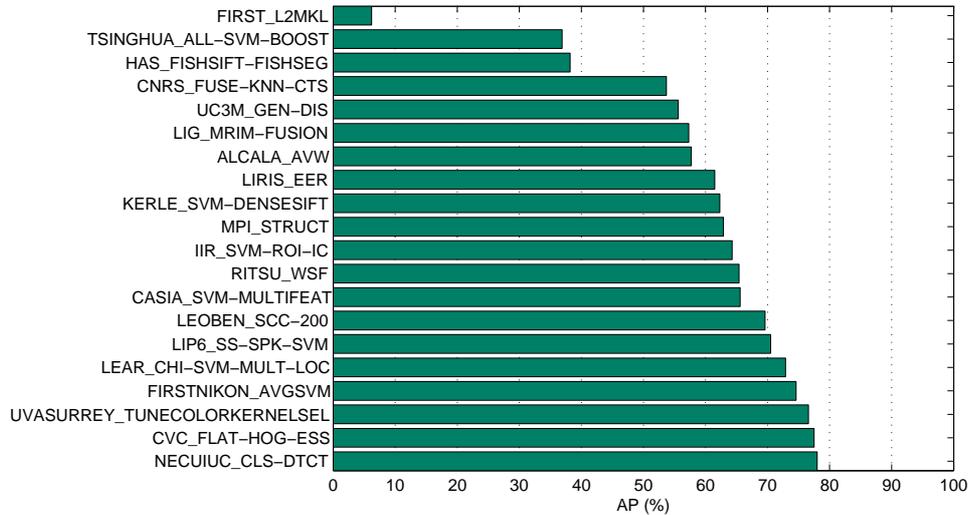


Figure 6.9: PASCAL VOC Challenge 2009: Median Average Precision. Best result by group.

	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
AVW	66.1	34.6	43.0	57.0	20.0	35.3	40.5	47.3	37.8	26.6	17.8	36.3	38.5	36.2	74.7	10.9	26.1	30.7	58.0	43.8
LAVW	65.7	31.3	44.0	53.8	16.7	39.3	41.7	43.7	35.0	20.0	19.0	31.9	40.9	37.1	73.6	19.0	23.4	22.0	56.9	45.6

Table 6.6: PASCAL VOC 2008 Results. Average precision scores of our methods.

6.11 shows there is correlation in the results.

It is also worth noting that all methods, including ours, have no bias toward larger objects. Although there is a moderate evidence for some classes, *e.g.* bicycle and car (see Figure 6.12), for most classes, correlation with object area is zero or negative (See Figure 6.13).

## 6.5 Conclusion

In this chapter, we have introduced the clustering aggregation techniques into the process of creating visual codebooks for image categorisation. To the best of our knowledge, this is the first work to describe such a clustering aggregation methodology within this context. Given a set of clusterings over the local descriptors, it is possible to obtain a clustering which agrees as much as possible with the given codebooks. The results show that the ave-

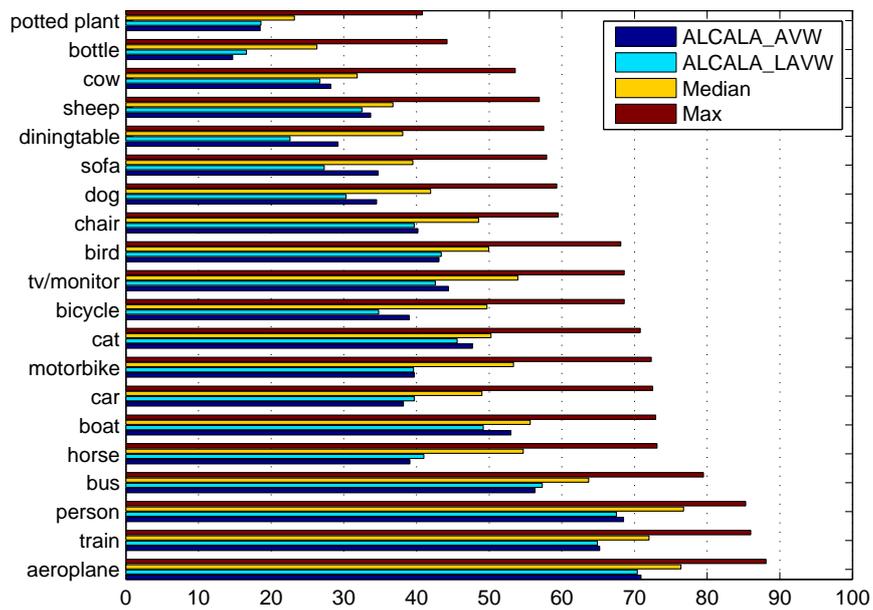


Figure 6.10: PASCAL VOC Challenge 2009: average precision per class (max, median, ALCALA\_AVW and ALCALA\_LAVW).

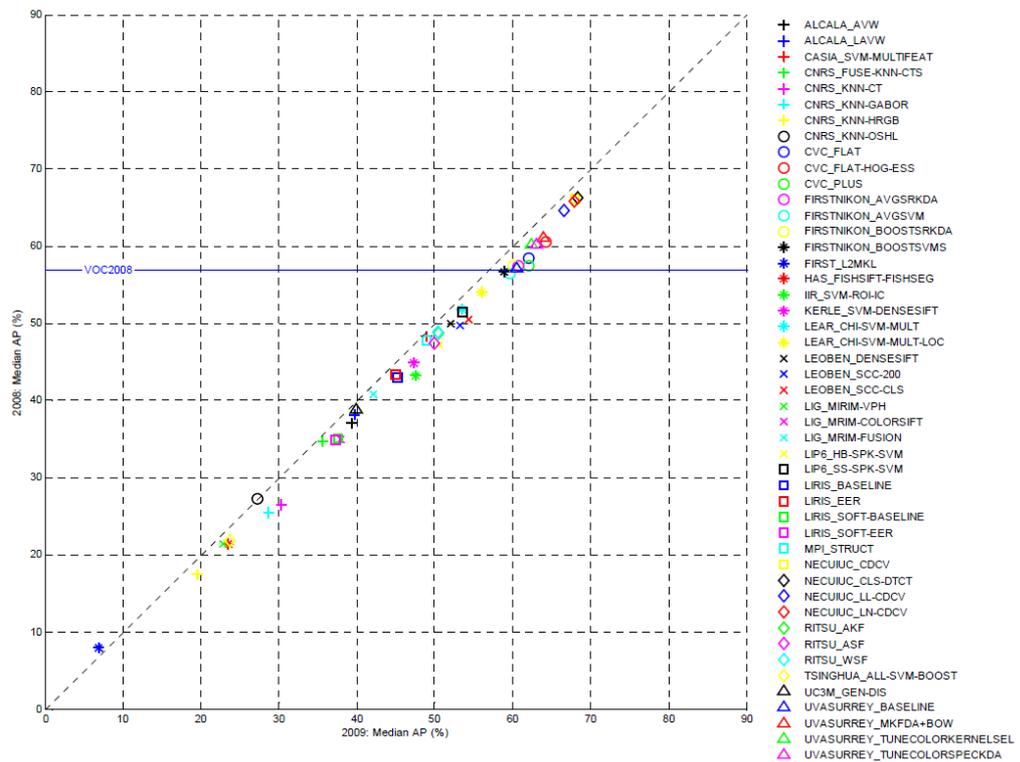
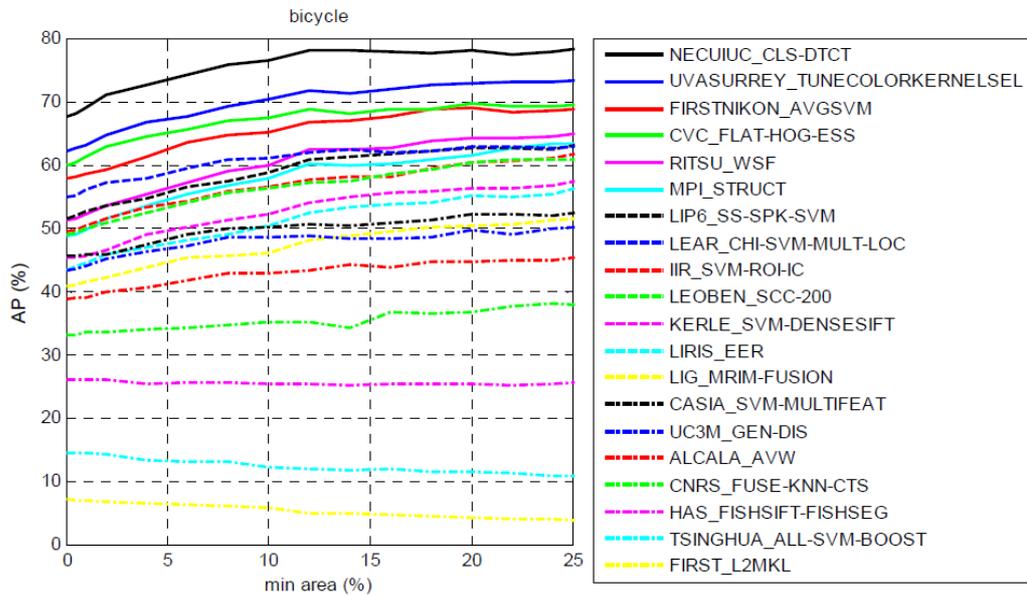
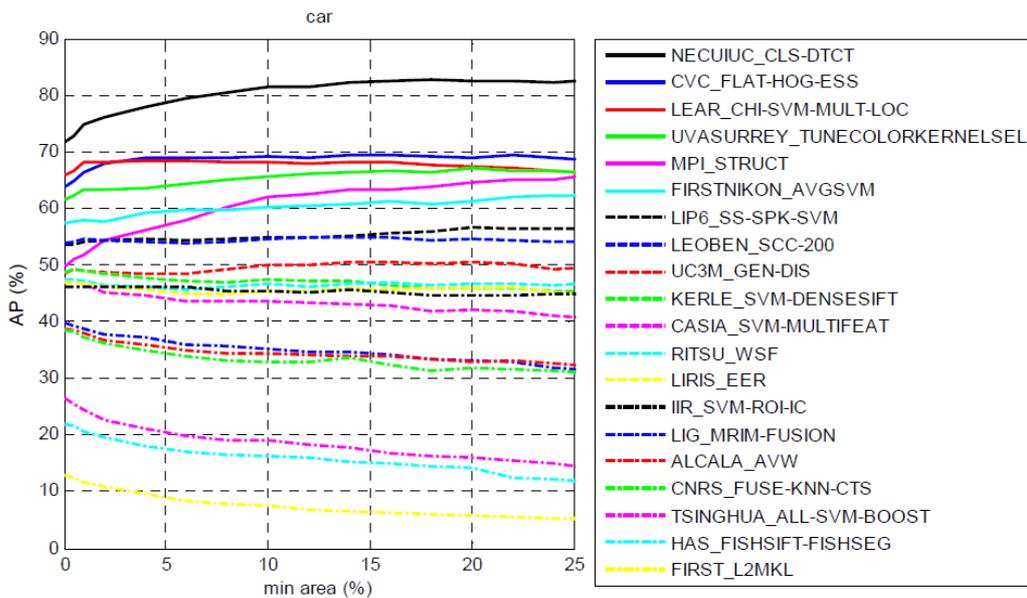


Figure 6.11: PASCAL VOC Challenge 2009 vs. PASCAL VOC Challenge 2008. Plot reproduced from (Everingham et al. 2009b).

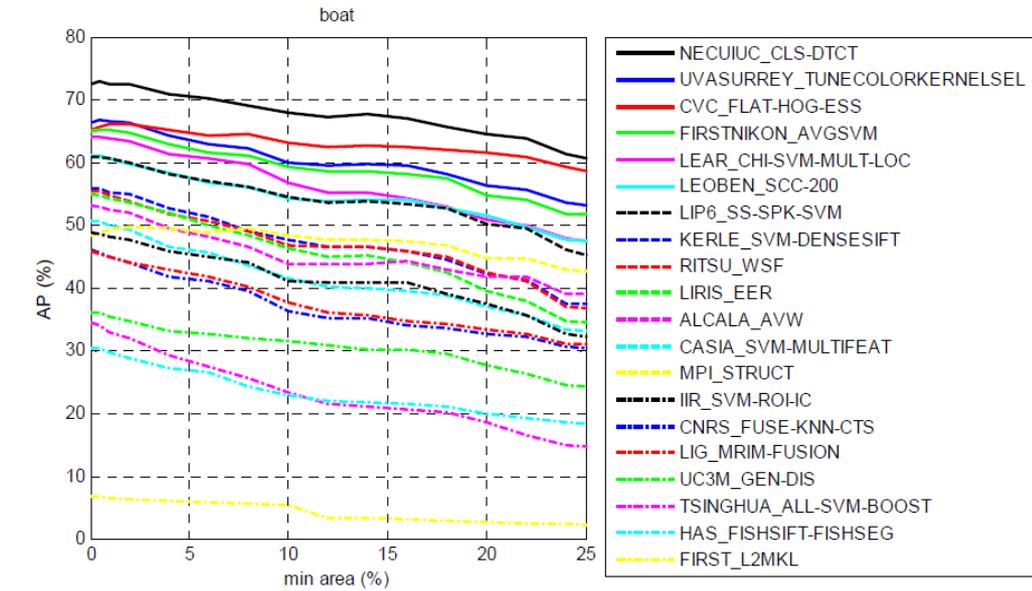


(a) bicycle

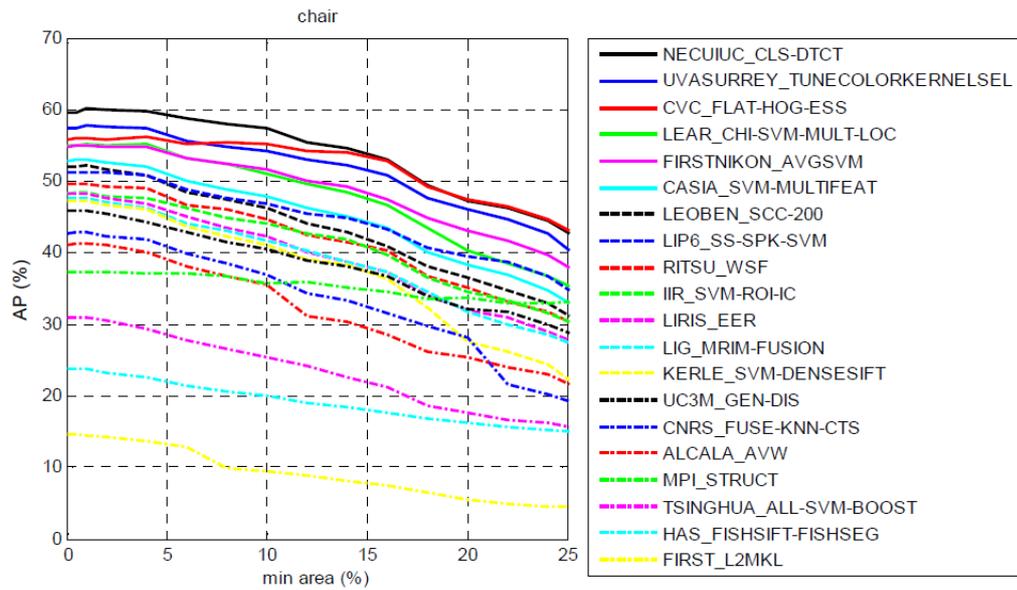


(b) car

Figure 6.12: Average Precision vs. Object Class Area: (a) bicycle, (b) car. Plots reproduced from (Everingham et al. 2009b).



(a) boat



(b) chair

Figure 6.13: Average Precision vs. Object Class Area: (a) boat, (b) chair. Plots reproduced from (Everingham et al. 2009b).

rage precision increases when aggregating  $K$ -means and mean-shift clustering algorithms.

We also propose a novel framework for incorporating meaningful spatial coherency among the local features into the visual vocabulary construction through clustering aggregation algorithms. Results on the PASCAL VOC 2007 dataset confirm that this approach does not obtain the best results. The high variability in terms of object size, orientation, pose, illumination, position and occlusion, difficult the incorporation of semantic information into the vocabulary construction process with our approaches. We propose, for future work, to train with more uniform datasets, like Caltech-101 (Fei-Fei et al. 2004), to study whether with them the clustering aggregation approaches obtain better results in categorisation.

It is worth mentioning that our approaches are not tied to any specific clustering algorithm or segmentation process. Exploring other clusterings as well as other datasets is another interesting avenue of future research.



# Chapter 7

## Conclusions

*And if you find her poor, Ithaca won't have fooled you.  
Wise as you will have become, so full of experience,  
you will have understood by then what these Ithakas mean.*

Constantine P. Cavafy, *Ithaca*.

---

### 7.1 Summary and main contributions

In this thesis, we have stated models and methods for object categorisation. Specifically, we have explored new approaches for building visual vocabularies for category-level object recognition. Our aim has been not just to obtain more discriminative and more compact visual codebooks, but to bridge the gap between visual features and semantic concepts. Furthermore, we have proposed efficient learning methodologies and efficient vector quantisation methods for high-dimensional data. Here we briefly summarise the main contributions of our research work:

- We have released a new database called ICARO. It has been especially designed to evaluate category-level object recognition approaches which can work both in 2D and 3D. An exhaustive comparison of ICARO with other well-known datasets used within the same context has been carried out. Furthermore, a benchmark for both object classification and detection has been established. However, our aim with ICARO is not only to provide a new benchmark for computer vision, but also to offer a challenging dataset that will grow under high quality annotation directives. The ICARO database is publicly available for scientific research purposes. All the images and the annotations can be downloaded from

<http://agamenon.tsc.uah.es/Personales/rlopez/data/icaro/>

- A novel approach for obtaining class representative visual words has been stated. First, we have introduced an improved measure for Cluster Precision, as well as a procedure to optimise the parameters during codebook construction without crossvalidation, by maximising this Cluster Precision measure. The CPM method measures the cluster precision for each class and automatically finds the thresholds for an RNN clustering algorithm that cast the best average recognition rates.
- A meta-clustering refinement algorithm has been presented. To the best of our knowledge, this is the first correlation clustering based approach for improving the class representativeness of visual words. The proposed methodology increases the average recognition rate and also dramatically compacts the size of the codebook.
- An speeded up version of the traditional RNN clustering algorithm based on the projection search paradigm has been described. This novel implementation, *i.e.* the fast-RNN, presents better run time results than the standard RNN algorithm. Results have been obtained with sets of PCA-SIFT, SIFT and SURF descriptors.
- We have stated novel clustering aggregation based approaches for building efficient and semantic visual codebooks. A novel framework for incorporating neighboring appearances of local descriptors into the vocabulary construction through clustering aggregation algorithms has been described. The aim of this new methodology is to increase the stability of the visual vocabulary construction process by combining several clusterings.
- We have proposed a rigorous approach for grouping semantically meaningful object parts, hence narrowing the semantic gap. With a clustering aggregation approach we are able to incorporate spatial coherency among the local descriptors into the visual vocabulary. The idea behind our novel approach is that local features in neighboring regions with similar appearance/colour should be characterised by the same visual word. In addition to the codebooks obtained in the descriptors domain, the spatial and colour quantisation can be introduced as inputs in the clustering aggregation process. This leads the aggregation process to obtain more semantic vocabularies.
- With the aim of making our research reproducible, we have used only publicly available datasets. Furthermore, a detailed specification of

the experimental setup is described in each chapter and reference implementations of the developed algorithms to replicate results are available too.

## 7.2 Future directions

There is too much work to do if we compare the state of the art approaches in object recognition with the human categorisation capabilities. However, the rapid development in categorisation is a fact, and the category-level object recognition community has a bright future ahead of itself.

In this thesis, we have presented several models and algorithms that can be extended in several ways. We briefly describe them below:

- Benchmarks for object detection and image categorisation have been established for the novel ICARO dataset. But a benchmark for 3D categorisation and detection is also needed. Others datasets have been used to that end, but the objects of interest neither appear in (very) different positions within the images, nor with other object classes in real world-scenes. A similar experimental setup to the one described by Savarese and Fei-Fei (2007) could be followed with ICARO in order to obtain a framework to compare results.
- ICARO has to increase in terms of the number of annotated classes and images, but always following a high quality annotation procedure.
- The CPM approach has been tested with the particular SIFT descriptor and a subset of the Caltech-101 dataset. To experiment with other descriptors (*e.g.* SURF), detectors and datasets (*e.g.* PASCAL VOC Challenge datasets and ICARO) has to be considered.
- The correlation clustering technique described in Section 4.3 has been only applied to an agglomerative clustering algorithm. However, it is a meta-clustering algorithm that can be used with vocabularies obtained with  $K$ -means or the meanshift based approach described in (Jurie and Triggs 2005). Further experiments with other clustering algorithms can be done to increase the class representativeness of the visual words and to refine the obtained clusters.
- Chapter 5 presents an speeded up version of the RNN clustering algorithm. Our approach is not based on approximate nearest neighbours

techniques. As a future line of research, we plan to apply such a techniques to hierarchical clustering approaches for quantising high dimensional vectors. To explore how the quality of the clusters and visual words affects the results of category-level object recognition systems has to be considered as well.

- The clustering aggregation framework presented in Section 6.3.1 can be extended too. For instance, more experiments with different vector-quantisation algorithms and different descriptors (*e.g.* colour descriptors (van de Sande et al. 2008)) can be done.
- It is also possible to extend the approach proposed from Section 6.3.2 to incorporate meaningful spatial coherency among the local features into the visual vocabulary construction. An obvious extension is to study how other (over-)segmentation algorithms (*e.g.* (Shi and Malik 2000, Comaniciu and Meer 2002)) and colour spaces (*e.g.* HSV, HSL) perform. Arbeláez et al. (2009) describes an algorithm that produces a hierarchical segmentation from the output of a contour detector. To study how to integrate such an approach into our framework can be interesting as well. Furthermore, it is also interesting to explore how colour attention maps (Khan et al. 2009) can be used to add semantic information to the vocabulary construction stage.

# Appendix A

## Evolution of cluster precision

Figure 4.4 depicts the evolution of  $P^{(i)}$  through iterations in the CPM approach. In this appendix we analytically analyse that  $P$  cannot stay the same if any clusters are merged.

For each iteration  $i$ , let  $\{P_m^{(i)}\}$  be the finite sequence of real numbers defined as

$$P_m^{(i)} = \frac{K^{(i)}}{M} \sum_{j_m=1}^{K^{(i)}} s_{j_m}^{(i)} n_{j_m}^{(i)}, \quad (\text{A.1})$$

where  $K^{(i)}$ ,  $M$ ,  $s_{j_m}^{(i)}$  y  $n_{j_m}^{(i)} \in \mathbb{N}$ .

Let  $i + 1$  be the iteration, where we consider that at least two clusters, say  $C_u$  and  $C_v$ , have been merged. This implies that the number of clusters, namely  $K^{(i+1)}$ , has decreased by one; *i.e.*  $K^{(i+1)} = K^{(i)} - 1$ . Therefore, one obtains

$$P_m^{(i+1)} = \frac{(K^{(i)} - 1)}{M} \sum_{j_m=1}^{K^{(i)}-1} s_{j_m}^{(i+1)} n_{j_m}^{(i+1)}. \quad (\text{A.2})$$

It is possible to develop equation (A.2) as follows

$$P_m^{(i+1)} = \frac{(K^{(i)} - 1)}{M} \left\{ \left( \sum_{j_m=1}^{K^{(i)}-2} s_{j_m}^{(i)} n_{j_m}^{(i)} \right) + (s_{u_m}^{(i)} + s_{v_m}^{(i)}) n_{u \cup v_m}^{(i+1)} \right\}, \quad (\text{A.3})$$

where  $n_{u \cup v_m}^{(i+1)}$  is the new number of class  $m$  object instances represented in the new cluster  $C_{u \cup v}$ .  $n_{u \cup v_m}^{(i+1)}$  will depend on whether the clusters  $C_u$  and  $C_v$  have features which belong to the same object instances. So to analyse equation

(A.3), we must consider two possible situations when merging clusters  $C_u$  and  $C_v$ : the clusters contain features which come from the same object instances or from different object instances.

**S1 – The same object instances in  $C_u$  and  $C_v$**  Let us assume that no new instance is added to cluster  $C_u$  when merging with  $C_v$ , and vice versa. Then  $n_{u \cup v_m}^{(i+1)} = n_{u_m}^{(i)} = n_{v_m}^{(i)}$ , therefore

$$\begin{aligned} P_m^{(i+1)} &= \frac{(K^{(i)} - 1)}{M} \left\{ \left( \sum_{j_m=1}^{K^{(i)}-2} s_{j_m}^{(i)} n_{j_m}^{(i)} \right) + (s_{u_m}^{(i)} + s_{v_m}^{(i)}) n_{u_m}^{(i)} \right\} \\ &= \frac{(K^{(i)} - 1)}{M} \left( \sum_{j_m=1}^{K^{(i)}} s_{j_m}^{(i)} n_{j_m}^{(i)} \right) = P_m^{(i)} \left( 1 - \frac{1}{K^{(i)}} \right). \end{aligned} \quad (\text{A.4})$$

Due to  $K^{(i)} > 1$ , then  $P_m^{(i+1)} < P_m^{(i)}$ . So, if no new view is added when merging the clusters, the cluster precision for each class decreases.

**S2 – Different object instances in  $C_u$  and  $C_v$**  Let  $\delta_{u_m}^{(i+1)}$  and  $\delta_{v_m}^{(i+1)}$  be the number of new class  $m$  object instances added to cluster  $C_u$  when merging with cluster  $C_v$  and vice versa, respectively. Then,

$$n_{u \cup v_m}^{(i+1)} = n_{u_m}^{(i)} + \delta_{u_m}^{(i+1)} = n_{v_m}^{(i)} + \delta_{v_m}^{(i+1)}. \quad (\text{A.5})$$

Once this merging is done the cluster precision for class  $m$  can be expressed as follows

$$P_m^{(i+1)} = \frac{(K^{(i)} - 1)}{M} \left\{ \left( \sum_{j_m=1}^{K^{(i)}-2} s_{j_m}^{(i)} n_{j_m}^{(i)} \right) + (s_{u_m}^{(i)} + s_{v_m}^{(i)}) (n_{u_m}^{(i)} + \delta_{u_m}^{(i+1)}) \right\}, \quad (\text{A.6})$$

and by algebraic manipulation

$$P_m^{(i+1)} = P_m^{(i)} - \frac{P_m^{(i)}}{K^{(i)}} + \frac{(K^{(i)} - 1)}{M} (s_{u_m}^{(i)} \delta_{u_m}^{(i+1)} + s_{v_m}^{(i)} \delta_{v_m}^{(i+1)}). \quad (\text{A.7})$$

It is straightforward to note that  $P_m^{(i+1)} \geq P_m^{(i)}$  when

$$-\frac{P_m^{(i)}}{K^{(i)}} + \frac{(K^{(i)} - 1)}{M} (s_{u_m}^{(i)} \delta_{u_m}^{(i+1)} + s_{v_m}^{(i)} \delta_{v_m}^{(i+1)}) \geq 0, \quad (\text{A.8})$$

which implies

$$(s_{v_m}^{(i)} \delta_{v_m}^{(i+1)} + s_{u_m}^{(i)} \delta_{u_m}^{(i+1)}) > \frac{MP_m^{(i)}}{K^{(i)}(K^{(i)} - 1)} \rightarrow P_m^{(i+1)} > P_m^{(i)}, \quad (\text{A.9})$$

$$(s_{v_m}^{(i)} \delta_{v_m}^{(i+1)} + s_{u_m}^{(i)} \delta_{u_m}^{(i+1)}) < \frac{MP_m^{(i)}}{K^{(i)}(K^{(i)} - 1)} \rightarrow P_m^{(i+1)} < P_m^{(i)}. \quad (\text{A.10})$$

So when there are different object instances in  $C_u$  and  $C_v$ , the cluster precision for each class can either increase or decrease, depending on equations (A.9) and (A.10).

Iteration by iteration, while the threshold increases and the RNN clustering is executed,  $K$  decreases but clusters grow in terms of size. When clusters are big the likelihood of having more different object instances represented in a cluster is greater. This implies that while the threshold increases cluster precision tends to decrease as situation **S1** is more probable than **S2**.

On the other hand, during the first iterations, when the threshold  $t$  is small and the number of clusters  $K$  is high, situation **S2** is more probable. When the threshold increases,  $s_{u_m}$  and  $s_{v_m}$  increase for sure. Moreover,  $\delta_{u_m}$  and  $\delta_{v_m}$  increase during the first iterations until a threshold after which no new object instances are added when clusters are merged. In the beginning we can also consider  $K \sim S$ . All this concludes in that condition (A.9) is more probable during the first iterations and condition (A.10) for the last ones.

Taken together, these considerations show what happens to  $P_m$  when clusters merge. As shown in Figure 4.4  $P$  has a maximum. Where this maximum is depends on given data. Nonetheless, it is possible to prove analytically that  $P$  cannot stay the same if any clusters are merged.

For any  $\Delta t > 0$ , however small,  $t_{i+1} = t_i + \Delta t$ . If any clusters are merged then  $P^{(i+1)} \neq P^{(i)}$ . Let us consider situation **S1**, where no new views are added when clusters are merged. As proved in equation (A.4),  $P_m^{(i+1)} < P_m^{(i)}$ , so  $P^{(i+1)} < P^{(i)}$ .

Suppose now that new views are added when clusters merge, *i.e.* situation **S2**. Recall that  $K, s_{u_m}, s_{v_m}, \delta_{u_m}, \delta_{v_m} \in \mathbb{N}$  and  $P_m^{(i)} \in \mathbb{Q}$ . Now, the following multivariate rational function is considered

$$f(x_1, \dots, x_6) = -\frac{x_1}{x_2} + \frac{(x_2 - 1)}{M} (x_3x_5 + x_4x_6) , \quad (\text{A.11})$$

Then, equality (A.7) can be expressed as

$$P_m^{(i+1)} = P_m^{(i)} + f(P_m^{(i)}, K^{(i)}, s_{u_m}^{(i)}, s_{v_m}^{(i)}, \delta_{u_m}^{(i+1)}, \delta_{v_m}^{(i+1)}) , \quad (\text{A.12})$$

note that  $K^{(i)} \neq 0$ , and hence the above specialisation of  $f$  is well-defined.  $f$  can be expressed as  $f = g/(x_2)$ , where  $g = -x_1 + x_2(x_2 - 1)(x_3x_5 + x_4x_6)$ . In this situation, we observe that the iterative sequence may stabilise, *i.e.*  $P_m^{(i+1)} = P_m^{(i)}$  for some  $i$ , if  $g(x_1, \dots, x_6) = 0$  has zeros over  $\mathbb{N}$ . In our case, since  $P_m^{(i)} \in \mathbb{Q}$  and  $K^{(i)}, s_{u_m}^{(i)}, \delta_{u_m}^{(i+1)}, s_{v_m}^{(i)}, \delta_{v_m}^{(i+1)} \in \mathbb{N}$ , this phenomenon might happen. Nevertheless, because of the nature of the experiment and data, in practise this situation does not occur. In order to formally guarantee that the sequence never stabilises, one can proceed as follows. It is possible to introduce a small perturbation  $\epsilon$  in the sequence so as to guarantee that  $P_m$  always varies when clusters merge. More precisely, instead of working with  $P_m^{(i)}$ , we take

$$\tilde{P}_m^{(i)} = \frac{K^{(i)}}{M} \left( \sum_{j_m=1}^{K^{(i)}} s_{j_m}^{(i)} n_{j_m}^{(i)} + \epsilon \right) , \quad (\text{A.13})$$

where  $\epsilon$  is taken as a real non-rational number; *e.g.* one might take  $\epsilon = \sqrt{K^{(i)}/(K^{(i)} + 1)}$  or a similar expression in case it turns to be a rational number. Now, the important fact is that  $\tilde{P}_m^{(i)} \notin \mathbb{Q}$ . Under these new conditions one can prove that  $\{\tilde{P}_m^{(i)}\}$  never stabilises. Indeed: if there exists  $i$  such that  $\tilde{P}_m^{(i)} = \tilde{P}_m^{(i+1)}$ , one has that  $g(\tilde{P}_m^{(i)}, K^{(i)}, s_{u_m}^{(i)}, s_{v_m}^{(i)}, \delta_{u_m}^{(i)}, \delta_{v_m}^{(i)}) = 0$ . This implies that  $\tilde{P}_m^{(i)}$  can be expressed rationally in terms of  $K^{(i)}, s_{u_m}^{(i)}, s_{v_m}^{(i)}, \delta_{u_m}^{(i)}, \delta_{v_m}^{(i)}$ :

$$\tilde{P}_m^{(i)} = \frac{K^{(i)}(K^{(i)} - 1)}{M} (s_{u_m}^{(i)} \delta_{u_m}^{(i)} + s_{v_m}^{(i)} \delta_{v_m}^{(i)}) . \quad (\text{A.14})$$

So,  $\tilde{P}_m^{(i)} \in \mathbb{Q}$  which is a contradiction. So to conclude, it has been proved that  $P_m$  will always vary if at least one pair of clusters is merged when the threshold changes, *i.e.*  $P_m$  does not stabilise.

# Bibliography

- S. Agarwal, A. Awan, and D. Roth. Learning to detect objects in images via a sparse, part-based representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(11):1475–1490, Nov. 2004.
- P. Arbeláez, M. Maire, C. Fowlkes, and J. Malik. From contours to regions: An empirical evaluation. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2009.
- N. Bansal, A. Blum, and S. Chawla. Correlation clustering. *Machine Learning*, 56:89–113, 2004.
- H. Bay and T. Quack. Kooaba. <http://www.kooaba.com/>, 2006.
- H. Bay, T. Tuytelaars, and L. Van Gool. Surf: Speeded up robust features. In *Proceedings of the European Conference on Computer Vision*, volume 3951 of *Lecture Notes in Computer Science*, pages 404–417, 2006.
- J.P. Benzécri. Construction d’une classification ascendante hiérarchique par la recherche en chaîne des voisins réciproques. *Cahiers de l’Analyse des Données*, 2(7):209–218, 1982.
- K S. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft. When is ”nearest neighbor” meaningful? In *Proceedings of the 7th International Conference on Database Theory*, 1999.
- I. Biederman. Recognition by components: a theory of human image understanding. *Psychological Review*, 94:115–147, 1987.
- A. Bosch, A. Zisserman, and X. Muñoz. Scene classification via pLSA. In *Proceedings of the European Conference on Computer Vision*, volume 4, pages 517–530, 2006.
- A. Bosch, A. Zisserman, and X. Muñoz. Representing shape with a spatial pyramid kernel. In *Proceedings of the 6th ACM international conference on Image and Video retrieval*, pages 401–408, 2007.

- J. Burianek, A. Ahmadyfard, and J. Kittler. Soil-47, the surrey object image library. <http://www.ee.surrey.ac.uk/Research/VSSP/demos/colour/soil47/>, 2000.
- L. Cao and L. Fei-Fei. Spatially coherent latent topic model for concurrent segmentation and classification of objects and scenes. In *Proceedings of the International Conference on Computer Vision*, 2007.
- Chih-Chung Chang and Chih-Jen Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>.
- M. Charikar, V. Guruswami, and A. Wirth. Clustering with qualitative information. In *IEEE Symposium on Foundations of Computer Science*, pages 524–533, 2003.
- O. Chum and A. Zisserman. An exemplar model for learning object classes. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2007.
- D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5):1–18, 2002.
- G. Csurka, C. R. Dance, L. Fan, J. Willamowski, and C. Bray. Visual categorization with bags of keypoints. In *Proceedings of the European Conference on Computer Vision*, 2004.
- CVPR. IEEE Computer Society Conference on Computer Vision and Pattern Recognition. <http://www.cvpr2009.org>, 2009.
- Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2005.
- W. H. Day and H. Edelsbrunner. Efficient algorithms for agglomerative hierarchical clustering methods. *Journal of Classification*, 1(1):7–24, 1984.
- C. de Rham. La classification hiérarchique ascendante selon la méthode des voisins réciproques. *Cahiers de l'Analyse des Données*, 2(5):135–144, 1980.
- E. D. Demaine, D. Emanuel, A. Fiat, and N. Immerlica. Correlation clustering in general weighted graphs. *Theoretical Computer Science*, 361(2):172–187, 2006.

- J. Deng, W. Dong, R. Socher, L-J Li, and L. Fei-Fei. ImageNet: A large-scale hierarchical image database. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2009.
- C. Elkan. Using the triangle inequality to accelerate k-means. In *Proceedings of the Twentieth International Conference on Machine Learning*, 2003.
- M. Everingham, A. Zisserman, C. K. I. Williams, and L. Van Gool. The PASCAL Visual Object Classes Challenge 2006 (VOC2006) Results. <http://www.pascal-network.org/challenges/VOC/voc2006/results.pdf>, 2006.
- M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results. <http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html>, 2007.
- M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2008 (VOC2008) Results. <http://www.pascal-network.org/challenges/VOC/voc2008/workshop/index.html>, 2008.
- M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes (VOC)challenge. *International Journal of Computer Vision*, page In press, 2009a.
- M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2009 (VOC2009) Results. <http://www.pascal-network.org/challenges/VOC/voc2009/workshop/index.html>, 2009b.
- L. Fei-Fei and P. Perona. A bayesian hierarchical model for learning natural scene categories. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, volume 2, pages 524–531, 2005.
- L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, Workshop on Generative-model Based Vision*, 2004.
- L. Fei-Fei, R. Fergus, and A. Torralba. Recognizing and learning object categories. <http://people.csail.mit.edu/torralba/shortCourseRLOC/>,

2009. Short course given at the International Conference on Computer Vision 2009.
- Christiane Fellbaum. *WordNet: An Electronic Lexical Database*. The MIT Press, 1998.
- P. Felzenszwalb and D. Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 59(2):167–181, 2004a.
- P. Felzenszwalb and D. Huttenlocher. Distance transforms of sampled functions. Technical report, Cornell Computing and Information Science, 2004b.
- P. Felzenszwalb and D. Huttenlocher. Pictorial structures for object recognition. *International Journal of Computer Vision*, 61(1):55–79, 2005.
- P. Felzenszwalb, D. McAllester, and D. Ramanan. A discriminatively trained, multiscale, deformable part model. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2008.
- R. Fergus, P. Perona, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, volume 2, pages 264–271, 2003.
- R. Fergus, Y. Weiss, and A. Torralba. Semi-supervised learning in gigantic image collections. In *Advances in Neural Information Processing Systems*, 2009.
- X. Z. Fern and C. E. Brodley. Solving cluster ensemble problems by bipartite graph partitioning. In *Proceedings of the twenty-first international conference on Machine learning*, 2004.
- V. Ferrari, T. Tuytelaars, and L. Van Gool. Simultaneous object recognition and segmentation from single or multiple model views. *International Journal of Computer Vision*, 67(2):159–188, 2006.
- M. A. Fischler and R. A. Elschlager. The representation and matching of pictorial structures. *IEEE Transactions on Computers*, 22(1):67–92, 1973.
- D. Francois, V. Wertz, and M. Verleysen. The concentration of fractional distances. *IEEE Transactions on Knowledge and Data Engineering*, 19(7):873–886, 2007.

- J. H. Friedman, F. Baskett, and L. J. Shustek. An algorithm for finding nearest neighbors. *IEEE Transactions on Computers*, 24(10):1000–1006, 1975.
- B. Fulkerson, A. Vedaldi, and S. Soatto. Localizing objects with smart dictionaries. In *Proceedings of the European Conference on Computer Vision*, pages 179–192, 2008.
- J. M. Geusebroek, G. J. Burghouts, and A. W. M. Smeulders. The amsterdam library of object images. *International Journal of Computer Vision*, 61(1):103–112, 2005.
- A. Gilbert, J. Illingworth, and R. Bowden. Fast realistic multi-action recognition using mined dense spatio-temporal features. In *Proceedings of the International Conference on Computer Vision*, 2009.
- A. Gionis, H. Mannila, and P. Tsaparas. Clustering aggregation. In *International Conference on Data Engineering*, pages 341–352, 2005.
- A. Gionis, H. Mannila, and P. Tsaparas. Clustering aggregation. *ACM Transactions on Knowledge Discovery from Data*, 1(1):4, 2007.
- Google. Google goggles. <http://www.google.com/mobile/goggles/>, 2010.
- K. Grauman and T. Darrell. The pyramid match kernel:discriminative classification with sets of image features. In *Proceedings of the International Conference on Computer Vision*, pages 1458–1465, 2005.
- G. Griffin, A. Holub, and P. Perona. Caltech-256 object category dataset. Technical Report 7694, California Institute of Technology, 2007.
- D. Hoiem, A. Efros, and M. Hebert. Putting objects in perspective. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2006.
- D. Hoiem, C. Rother, and J. Winn. 3D LayoutCRF for multi-view object class recognition and segmentation. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2007.
- X. Hu and I. Yoo. Cluster ensemble and its applications in gene expression analysis. In *Proceedings of the second conference on Asia-Pacific bioinformatics*, 2004.

- T. S. Jaakkola and D. Haussler. Exploiting generative models in discriminative classifiers. In *Proceedings of the 1998 conference on Advances in neural information processing systems II*, pages 487–493, 1999.
- A. K. Jain. Data clustering: 50 years beyond k-means. *Pattern Recognition Letters*, In press, 2009.
- H. Jegou, H. Harzallah, and C. Schmid. A contextual dissimilarity measure for accurate and efficient image search. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2007.
- F. Jurie and B. Triggs. Creating efficient codebooks for visual recognition. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2005.
- T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, and A. Y. Wu. An efficient K-means clustering algorithm: analysis and implementation. *IEEE Transactions on Pattern Recognition and Machine Intelligence*, 24(7):881–892, 2002.
- Y. Ke and R. Sukthankar. Pca-sift: A more distinctive representation for local image descriptors. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2004.
- F. S. Khan, Joost van de Weijer, and M. Vanrell. Top-down color attention for object recognition. In *Proceedings of the International Conference on Computer Vision*, 2009.
- A. Kushal and J. Ponce. Modeling 3d objects from stereo views and recognizing them in photographs. In *Proceedings of the European Conference on Computer Vision*, volume 2, pages 563–574, 2006.
- A. Kushal, C. Schmid, and J. Ponce. Flexible object models for category-level 3d object recognition. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2007.
- C. H. Lampert, M. B. Blaschko, and T. Hofmann. Efficient subwindow search: A branch and bound framework for object localization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(12):2129–2142, 2009.
- G. N Lance and W. T. Williams. A general theory of classificatory sorting strategies 1. hierarchical systems. *The Computer Journal*, 9(4):373–380, 1967.

- S. Lazebnik and M. Raginsky. Learning nearest-neighbor quantizers from labeled data by information loss minimization. In *International Conference on Artificial Intelligence and Statistics*, 2007.
- S. Lazebnik, C. Schmid, and J. Ponce. Semi-local affine parts for object recognition. In *Proceedings of the British Machine Vision Conference*, 2004.
- S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, volume 2, pages 2169–2178, 2006.
- B. Leibe and B. Schiele. Interleaved object categorization and segmentation. In *Proceedings of the British Machine Vision Conference*, 2003.
- B. Leibe, A. Leonardis, and B. Schiele. Combined object categorization and segmentation with an implicit shape model. In *ECCV'04 Workshop on Statistical Learning in Computer Vision*, pages 17–32, 2004.
- B. Leibe, K. Mikolajczyk, and B. Schiele. Efficient clustering and matching for object class recognition. In *Proceedings of the British Machine Vision Conference*, 2006.
- B. Leibe, A. Ettl, and B. Schiele. Learning semantic object parts for object categorization. *Image and Vision Computing*, 26(1):15–26, 2008a.
- B. Leibe, A. Leonardis, and B. Schiele. Robust object detection with interleaved categorization and segmentation. *International Journal of Computer Vision*, 77(1-3):259–289, 2008b.
- V. Lepetit and P. Fua. Keypoint recognition using randomized trees. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(9):1465–1479, Sept. 2006.
- T. Leung and J. Malik. Representing and recognizing the visual appearance of materials using three-dimensional features. *International Journal of Computer Vision*, 43(1):29–44, 2001.
- J. Liebelt, C. Schmid, and K. Schertler. Viewpoint-independent object class detection using 3d feature maps. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2008.
- Like.com. Like.com. <http://www.like.com>, 2005.

- J. Liu, Y. Yang, and M. Shah. Learning semantic visual vocabularies using diffusion distance. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2009.
- D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.
- D.G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the International Conference on Computer Vision*, September 1999.
- D.G. Lowe. Local feature view clustering for 3D object recognition. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, volume 1, pages 682–688, 2001.
- F. Lv and R. Nevatia. Single view human action recognition using key pose matching and viterbi path searching. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2007.
- S. Maji, A. C. Berg, and J. Malik. Classification using intersection kernel support vector machines is efficient. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2008.
- K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(10):1615–1630, 2005.
- K. Mikolajczyk, B. Leibe, and B. Schiele. Local features for object class recognition. In *Proceedings of the International Conference on Computer Vision*, 2005.
- F. Moosmann, B. Triggs, and F. Jurie. Fast discriminative visual codebooks using randomized clustering forests. In *Advances in Neural Information Processing Systems*, 2006.
- H. Murase and S.K. Nayar. Learning and recognition of 3D objects from appearance. In *Proceedings of IEEE Workshop on Qualitative Vision*, pages 39–50, 14 June 1993.
- S. Nene, S. Nayar, and H. Murase. Columbia object image library (coil-100). Technical Report CUCS-006-96, Columbia University, 1996.
- S. A. Nene and S. K. Nayar. A simple algorithm for nearest neighbor search in high dimensions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(9):989–1003, 1997.

- A. Y. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems 14*, pages 849–856. MIT Press, 2001.
- J. C. Niebles and L. Fei-Fei. A hierarchical model of shape and appearance for human action classification. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2007.
- D. Nister and H. Stewenius. Scalable recognition with a vocabulary tree. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 2161–2168, 2006.
- A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *International Journal of Computer Vision*, 42(3):145–175, 2001.
- A. Opelt, A. Pinz, M. Fussenegger, and P. Auer. Generic object recognition with boosting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(3):416–431, 2006.
- F. Perronnin and C. Dance. Fisher kernels on visual vocabularies for image categorization. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2007.
- F. Perronnin, C. Dance, G. Csurka, and M. Bressan. Adapted vocabularies for generic visual categorization. In *Proceedings of the European Conference on Computer Vision*, 2006.
- J. Philbin, O. Chum, J. Isard, M. and Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2007.
- A. Pinz. Object categorization. *Foundations and Trends in Computer Graphics and Vision*, 1(4):255–353, 2006.
- J. Ponce, T.L. Berg, M. Everingham, D.A. Forsyth, M. Hebert, S. Lazebnik, M. Marszalek, C. Schmid, B.C. Russell, A. Torralba, C.K.I. Williams, J. Zhang, and A. Zisserman. *Toward category-level object recognition*, chapter Dataset Issues in Object Recognition, pages 29–48. Springer, 2006.
- T. Quack, V. Ferrari, B. Leibe, and L. Van Gool. Efficient mining of frequent and distinctive feature configurations. In *Proceedings of the International Conference on Computer Vision*, 2007.

- P. Quelhas, F. Monay, J.-M. Odobez, D. Gatica-Perez, T. Tuytelaars, and L. Van Gool. Modeling scenes with local descriptors and latent aspects. In *Proceedings of the International Conference on Computer Vision*, 2005.
- P. Quelhas, F. Monay, J.-M. Odobez, D. Gatica-Perez, and T. Tuytelaars. A thousand words in a scene. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(9):1575–1589, 2007.
- F. Rothganger, S. Lazevnik, C. Schmid, and J. Ponce. 3d object modeling and recognition using local affine-invariant image descriptors and multi-view spatial constraints. *International Journal of Computer Vision*, 66(3): 231–259, 2006.
- B. C. Russell, A. A. Efros, J. Sivic, B. C. Freeman, and A. Zisserman. Using multiple segmentations to discover objects and their extent in image collections. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2006.
- B.C. Russell, A. Torralba, K.P. Murphy, and W.T. Freeman. Labelme: A database and web-based tool for image annotation. *International Journal of Computer Vision*, 77(1–3):157–173, 2008.
- J. Savarese, S. Winn and C. Criminisi. Discriminative object class models of appearance and shape by correlatons. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 2033–2040, 2006.
- S. Savarese and L. Fei-Fei. 3d generic object categorization, localization and pose estimation. In *Proceedings of the International Conference on Computer Vision*, pages 1 – 8, 2007.
- S. Savarese and L. Fei-Fei. View synthesis for recognizing unseen poses of object classes. In *Proceedings of the European Conference on Computer Vision*, pages 602–615, 2008.
- H. Schneiderman and T. Kanade. A statistical method for 3d object detection applied to faces and cars. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, volume 1, pages 746–75, 13-15 June 2000.
- J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.
- J. Shotton, J. Winn, C. Rother, and A. Criminisi. Textonboost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation. In *Proceedings of the European Conference on Computer Vision*, 2006.

- J. Sivic and A. Zisserman. Video google: A text retrieval approach to object matching in videos. In *Proceedings of the International Conference on Computer Vision*, 2003.
- J. Sivic and A. Zisserman. Video data mining using configurations of viewpoint invariant regions. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 488–495, 2004.
- J. Sivic, B. C. Russell, A. A. Efros, A. Zisserman, and W. T. Freeman. Discovering object categories in image collections. In *Proceedings of the International Conference on Computer Vision*, 2005.
- M. Stark and B. Schiele. How good are local features for classes of geometric objects. In *Proceedings of the International Conference on Computer Vision*, pages 1–8, 2007.
- A. Strehl and J. Ghosh. Cluster ensembles - a knowledge reuse framework for combining multiple partitions. *Machine Learning Research*, 3:583–617, 2002.
- Bjarne Stroustrup. *The C++ Programming Language*. Addison-Wesley Longman Publishing Co., Inc., 2000.
- H. Su, M. Sun, L. Fei-Fei, and S. Savarese. Learning a dense multi-view representation for detection, viewpoint classification and synthesis of object categories. In *Proceedings of the International Conference on Computer Vision*, 2009.
- E. B. Sudderth, A. Torralba, W. T. Freeman, and S. A. Willsky. Learning hierarchical models of scenes, objects, and parts. In *Proceedings of the International Conference on Computer Vision*, 2005.
- M. Sun, H. Su, S. Savarese, and L. Fei-Fei. A multi-view probabilistic model for 3d object classes. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2009.
- J. Tahir, M.A. Kittler, K. Mikolajczyk, F. Yan, K.E.A. van de Sande, and T. Gevers. Visual category recognition using spectral regression and kernel discriminant analysis. In *Proceedings of the International Conference on Computer Vision*, 2009.
- A. Thomas, V. Ferrari, B. Leibe, T. Tuytelaars, B. Schiele, and L. Van Gool. Towards multi-view object class detection. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, volume 2, pages 1589–1596, 2006.

- A. Thomas, V. Ferrari, B. Leibe, T. Tuytelaars, and L. Van Gool. Depth-from-recognition: Inferring meta-data by cognitive feedback. In *Proceedings of the International Conference on Computer Vision, Workshop on 3D Representation for Recognition (3dRR)*, 2007.
- A. Torralba, K. P. Murphy, and W. T. Freeman. Sharing features: efficient boosting procedures for multiclass object detection. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, volume 2, pages 762–769, 2004.
- A. Torralba, R. Fergus, and W. Freeman. 80 million tiny images: a large dataset for non-parametric object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(11):1958–1970, 2008a.
- A. Torralba, R. Fergus, and Y. Weiss. Small codes and large databases for recognition. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2008b.
- T. Tuytelaars and K. Mikolajczyk. Local invariant feature detectors: A survey. *Foundations and Trends in Computer Graphics and Vision*, 3(3): 177–280, 2008.
- T. Tuytelaars, C.H. Lampert, M.B. Blaschko, and W. Buntine. Unsupervised object discovery: A comparison. *International Journal of Computer Vision*, 2009.
- S. Ullman. Three-dimensional object recognition based on the combination of views. *Cognition*, 67:21–44, 1998.
- K. van de Sande, T. Gevers, and C. Snoek. Evaluation of color descriptors for object and scene recognition. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2008.
- K. van de Sande, T. Gevers, and C. Snoek. Evaluating color descriptors for object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, In press, 2010.
- J. C. van Gemert, J. M. Geusebroek, C. J. Veenman, and A. W. Smeulders. Kernel codebooks for scene categorization. In *Proceedings of the European Conference on Computer Vision*, 2008.
- M. Varma and D. Ray. Learning the discriminative power-invariance trade-off. In *Proceedings of the International Conference on Computer Vision*, 2007.

- J. Vogel and B. Schiele. Semantic modeling of natural scenes for content-based image retrieval. *International Journal of Computer Vision*, 72(2): 133–157, 2007.
- H. Wang, H. Shan, and A. Banerjee. Bayesian cluster ensembles. In *Proceedings of the 9th SIAM International Conference on Data Mining*, 2009.
- M. Weber, , M. Welling, and P. Perona. Towards automatic discovery of object categories. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2000a.
- M. Weber, W. Einhäuser, M. Welling, and P. Perona. Viewpoint-invariant learning and detection of human heads. In *Proceedings of the 4th IEEE International Conference on Automatic Face and Gesture Recognition*, 2000b.
- J. Winn and J. Shotton. The layout consistent random field for recognizing and segmenting partially occluded objects. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2006.
- J. Winn, A. Criminisi, and A. Minka. Object categorization by learned universal visual dictionary. In *Proceedings of the International Conference on Computer Vision*, 2005.
- J. Xiao, J. Chen, D.Y. Yeung, and L. Quan. Structuring visual words in 3d for arbitrary-view object localization. In *Proceedings of the European Conference on Computer Vision*, pages 725–737, 2008.
- P. Yan, S.M. Khan, and M. Shah. 3d model based object class detection in an arbitrary view. In *Proceedings of the International Conference on Computer Vision*, pages 1–6, 2007.
- L. Yang, R. Jin, R. Sukthankar, and F. Jurie. Unifying discriminative visual codebook generation with classifier training for object category recognition. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2008.
- Z. Yu, Z. Deng, H. S. Wong, and X. Wang. Fuzzy cluster ensemble and its application on 3D head model classification. In *Neural Networks, 2008. IJCNN 2008. (IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on*, pages 569–576, 2008.
- J. Yuan, Y. Wu, and M. Yang. Discovery of collocation patterns: from visual words to visual phrases. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2007.

- J. Zhang, M. Marszalek, S. Lazebnik, and C. Schmid. Local features and kernels for classification of texture and object categories: A comprehensive study. *International Journal of Computer Vision*, 73(2):213–238, 2007.

*What we cannot speak of we must pass over in silence.*  
L. Wittgenstein, *Tractatus Logico-Philosophicus*.