UNIVERSITY OF SOUTHAMPTON

Designing a Resource-Allocating Codebook for Patch-based Visual Object Recognition

by

Amirthalingam Ramanan

A thesis submitted in partial fulfillment for the
degree of Doctor of Philosophy

in the
Faculty of Engineering, Science and Mathematics
School of Electronics and Computer Science

June 2010

UNIVERSITY OF SOUTHAMPTON

ABSTRACT

FACULTY OF ENGINEERING, SCIENCE AND MATHEMATICS
SCHOOL OF ELECTRONICS AND COMPUTER SCIENCE

Doctor of Philosophy

by Amirthalingam Ramanan

The state-of-the-art approach in visual object recognition is the use of local information extracted at several points or image patches from an image. Local information at specific points can deal with object shape variability and partial occlusions. The underlying idea is that, in different images, the statistical distribution of the patches is different, which can be effectively exploited for recognition. In such a patch-based object recognition system, the key role of a visual codebook is to provide a way to map the low-level features into a fixed-length vector in histogram space to which standard classifiers can be directly applied. The discriminative power of a visual codebook determines the quality of the codebook model, whereas the size of the codebook controls the complexity of the model. Thus, the construction of a codebook plays a central role that affects the model's complexity. The construction of a codebook is an important step which is usually done by cluster analysis. However, clustering is a process that retains regions of high density in a distribution and it follows that the resulting codebook need not have discriminant properties. This is also recognised as a computational bottleneck of such systems.

This thesis demonstrates a novel approach, that we call resource-allocating codebook (RAC), to constructing a discriminant codebook in a one-pass design procedure inspired by the resource-allocation network family of algorithms. The RAC approach slightly outperforms more traditional approaches due to its tendency to spread out the cluster centres over a broader range of the feature space thereby including rare low-level features in the codebook than density-preserving clustering-based codebooks. Our algorithm achieves this performance at *drastically reduced* computing times, because apart from an initial scan through a small subset to determine length scales, each data item is processed only once.

We illustrate some properties of our method and compare it to a closely related approach known as the mean-shift clustering technique. A pruning strategy has been employed to tackle a few outliers when assigning each feature in images to the closest codeword to create a histogram representation for each image. Features whose distance from the closest codeword exceeds an empirical distance maximum are neglected. A recognition system that learns incrementally with training images and the output classifier accounting for class-specific discriminant features is also presented. Furthermore, we address an approach which, instead of clustering, adaptively constructs a codebook by computing Fisher scores between the classes of interest.

This thesis also demonstrates a novel sequential hierarchical clustering technique that initially builds a hierarchical tree from a small subset of the data, while the remaining data are processed sequentially and the tree adapted constructively. Evaluations performed with this approach show that the performance is comparable while reducing the computational needs. Finally, during the process of classification, we demonstrate a new learning architecture for multi-class classification tasks using support vector machines. This technique is faster in testing compared to directed acyclic graph (DAG) SVMs, while maintaining comparable performance to the standard multi-class classification techniques.

Contents

List of Figures

List of Tables

Acknowledgements

First and foremost, my heartily profound gratitude and appreciation are addressed to my supervisor Professor Mahesan Niranjan for taking me as a research student under his valuable supervision. His advice, discussions and guidance were the real encouragement to complete this work. I admire his creativity, simplicity, generosity, work ethic, and the ability to balance work and life. It has been an honour to work with him. I will always be thankful to him for the valuable times that he spent in supervising my progress, and the travel opportunities he gave me during this time. I would also like to thank Professor Mike Holcoumbe and Dr Joab Winkler who monitored my progress as the members of my MPhil/PhD committee at the University of Sheffield.

I am forever grateful to Dr S.Mahesan who started my career as a student in the stream of computer science. My sincere thanks to Dr E.Y.A.Charles and Professor R.Kumaravadivel for their guidance and support in all the administrative correspondence with the University of Jaffna, Sri Lanka.

I am very grateful to the University of Sheffield for financially supporting me for the first half of my MPhil/PhD programme and the University of Southampton for the second half. Also, I am very much grateful to the World Bank for financially supporting me throughout my PhD programme through the IRQUE project[1] implemented at the University of Jaffna. I would like to thank Professor K.Kandasamy, Dr R.Vigneswaran and other executive members of the IRQUE project who implemented all their administrative support through the project at the University of Jaffna. I am also very thankful to the PASCAL Network of Excellence[2] for the travel grants given to me during this time. I would also like to thank the academic and administrative members of the Department of Computer Science at the University of Sheffield and the School of Electronics and Computer Science at the University of Southampton for their support.

Joining the ISIS research group was a wonderful experience. I have been lucky to share an office with so many friendly people (in alphabetical order): Ali, Bassam, Daisy, Ke, Mustansar, Ni, Qasem, Salih, Somjet, SungUk, Tayyaba, Wei, Xin and Zoe who made it a convivial place to work. Thanks for all of the nice conversations and for being such an integral part of my postgraduate school experience.

Finally, I am forever indebted to my parents and my wife, who have supported and encouraged me through their kindness and affection, so that I could concentrate on my studies. They touched me more deeply than I could have ever expected.

[1] http://www.irque.lk/

[2] http://www.pascal-network.org/

To my beloved father, mother, wife and to my mentors ...

Chapter 1

Introduction

Visual object recognition and scene classification have caused a great challenge for computer vision with the long term goal of being able to achieve near human levels of recognition. Each three-dimensional object in the real world can cast an infinite number of different two-dimensional images onto the retina. Changes in pose, lighting, occlusion, clutter, intra-class differences, inner-class variances, deformations, background that varies relative to the viewer, large number of images and several object categories make the problem highly challenging. For example, in Figure 1.1, we see two images each of the London Tower Bridge, the BMW M3, the Sri Lankan Airlines A330-200, and the CN Tower, respectively. Despite changes in the background, lighting, and pose of the image, they are still the same object, but an artificial system might find this hard to recognise. However, the human brain solves this problem effortlessly as it can recognise about 10^4 to 10^5 objects. Advances in machine learning and image feature representations have led to great progress in pattern recognition approaches in recognising up to 10^2 visual object categories (Fei-Fei and Perona, 2005).

FIGURE 1.1: Two instances of four specific objects (images taken from Google-images). (a) the London Tower Bridge (b) the BMW M3 (c) the Sri Lankan Airlines A330-200 (d) the CN Tower.

Computer vision is concerned with the theory and technology for building artificial systems based on the information obtained from images. It shares topics such as image and signal processing, machine vision, visual cognition and pattern recognition. Some of the real world applications of computer vision include: content-based image retrieval, fingerprint recognition and Gait analysis for security systems, object or scene categorisation, image restoration, video tracking, motion estimation, and medical imaging. Computer vision in turn provides new avenues for machine learning techniques to explore.

Machine learning deals with the design and development of algorithms and techniques that allow computers to extract useful information from complex datasets, i.e. it is concerned with the evaluation of algorithms that simplify the process of classification, regression, recognition, and prediction based on models derived from example data or prior experience. Machine learning includes several application areas such as: pattern recognition, bioinformatics, neural networks, speech and handwriting recognition, spam filtering in emails, and web-based search engines.

This thesis can be divided into two parts: In the first part we look at the well-known problem in machine learning of clustering large scale data. We demonstrate two novel approaches tested on benchmark datasets. In the second part, we show how extending some previously known algorithms that are employed by support vector machines (SVMs) in multi-class classification allows us to solve the excessive computational needs while maintaining accuracy comparable to those standard algorithms tested.

Following are some terminologies that are used throughout this work.

- **Visual object class:** Is a collection of objects that share certain visual properties in common. Examples for a visual class would be 'airplane', 'bicycle', 'bird', 'tree', 'buildings' or 'person'.
- **Recognition:** Is the process of perceiving visual objects. In our experiments, the term 'recognition' is used in two contexts: (i) for visual object classes, the 'classification' is meant, (ii) for faces, this term is equal to 'identification'.
- **Classification:** In the area of object recognition, classification is the process of assigning images to a predefined number of classes. One image might be assigned to different classes, depending on the presence of multiple object categories in that image, such as in the case of the PASCAL Visual Object Classes (VOC) challenge image dataset[1]. An alternative term used in the literature is 'categorisation'.
- **Codebook:** Is used to quantize the continuous high dimensional space of local image descriptors such as scale-invariant feature transform (SIFT) (Lowe, 1999) into a fixed-length vector in histogram space to which standard classifiers can be directly applied. Usually a visual codebook is constructed by means of clustering techniques.

[1] http://pascallin.ecs.soton.ac.uk/challenges/VOC/

- **Codeword:** Each interest point cluster is treated as a 'codeword' in the visual codebook.
- **Bag-of-features:** An image is depicted as an orderless collection of patch-based features. For compact representation, a visual codebook is usually used to describe the bag-of-features.

1.1 Objective

A desired property of a codebook model-based visual object recognition system is that the time taken in constructing a visual codebook should scale sub-linearly with the number of object categories and/or visual descriptors. Otherwise, any system will become unmanageably slow once hundreds of categories or millions of descriptors are reached. Recently, many traditional approaches have gained favour as machines have become fast enough to make them practical in constructing a codebook on relatively large scale descriptors.

Our goal in this thesis is to address this issue by demonstrating a sequential algorithm that can handle a large number of patch-based descriptors in constructing visual codebooks in a drastically reduced time.

We also address the excess testing time incurred in multi-class classification tasks when large numbers of classes are involved. We do not find optimal features for solving a specific visual object recognition task, but rather focus on a particular, widely used, feature set (e.g. SIFT) and use this as the basis to address those problems.

1.2 Patch-based visual object recognition

As mentioned earlier, the important problem in computer vision is to determine the presence or absence of a specific object category under a wide variety of conditions. Most appearance-based approaches to visual object recognition characterise the objects by their global appearance, usually the entire image (Papageorgiou and Poggio, 2000). The popular approach in visual object recognition is to use local information extracted at several points or patches in the image. Such local patch-based approaches have been shown to have benefits over global methods (Leibe and Schiele, 2003). The assumption is, in different images, the statistical distribution of the patches is different. For instance, the patches showing spikes in the 'wheel' are more likely to appear in the images of vehicles than those of animals or persons. Although the bag-of-feature approach is a simple representation, it can handle viewpoint changes among the patch-based features.

The local patch-based visual object recognition has several advantages that we list below:

- Local patch-based descriptors can robustly detect regions up to some extent which are translation, rotation and scale invariants addressing the problem of viewpoint changes (Dorko et al., 2005; Quelhas et al., 2007).
- The viewpoint invariant local descriptors provide a wide baseline matching (Lowe, 2004).
- When objects to be recognised are partially occluded then the global method fails as it requires the outline of an object, but the patch-based method can cope well as the information is acquired at local point.
- Changes in the geometrical relation between image parts can be modelled to be flexible (Larlus and Jurie, 2006; Moosmann et al., 2007).
- The visual object classes do not need to be segmented prior to recognition (Leibe and Schiele, 2003).

Beside these advantages of the patch-based visual object recognition system, there are some known disadvantages.

- Although the interest points detected are significantly less than the number of pixels in the image, the feature space suffers from the 'curse of dimensionality', i.e. each interest point detected by SIFT is described by a 128 dimensional vector.
- When using the bag-of-features approach with the patch-based object recognition systems, the physical location from where the patches were extracted gets discarded. In image scene classification, e.g. classification of 'sand' and 'sky', the performance may achieve better rates when spatial locations are preserved, i.e. in natural scenes 'sand' always appears at the bottom, whereas 'sky' always appears at the top. However, the usage of latent information makes the training of object recognition models more difficult.
- Interest points are detected when sharp changes happens in the intensities at any resolution of the image regions, e.g. the Difference of Gaussians (DoG) (Lowe, 2004). This causes the problem that those relevant parts of the object that were detected in the testing images may be missed in the training images due to the resolution of images. If this is the case the classification which relies on these parts is likely to fail.

Several state-of-the-art visual object recognition systems (Csurka et al., 2004; Winn et al., 2005; Zhang et al., 2007; Li et al., 2008; Wu and Rehg, 2009a) fit into a general framework that is depicted in Figure 1.2 and is summarised in the following steps:

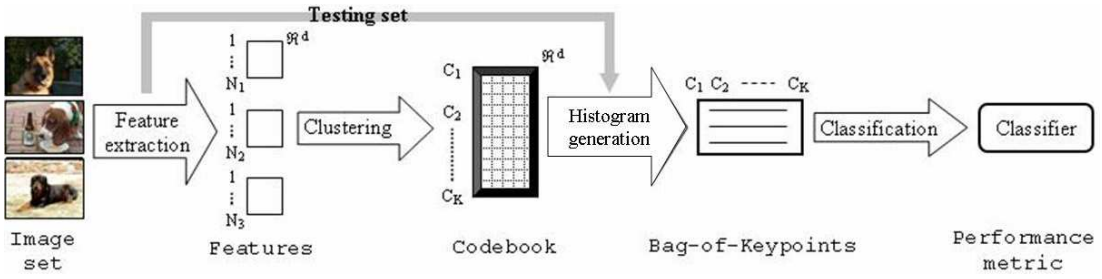


FIGURE 1.2: General framework of a visual object recognition system

1. *Feature extraction*: Detection and description of image patches from the image corpus.

2. *Cluster analysis*: Constructing a visual codebook by means of clustering techniques. The codebook is the set of centres of the learnt clusters.

3. *Histogram generation*: Mapping the extracted image patches from the image set into a feature vector by computing the frequency histograms with the learnt clusters. This mapping produces a bag-of-features representation, similar to the bag-of-words representation that was originally used in text classification (Joachims, 1998).

4. *Histogram classification*: Classifying the test set to predict which category or categories to assign to the image. The performance of the classifier is evaluated by an appropriate performance measure (e.g. classification rate, area under the receiver operating characteristic curve, and average precision).

This patch-based bag-of-features methodology has proved to yield state-of-the-art performance in large evaluations (e.g. the PASCAL VOC Challenge). This is why we restrict our attention to local patch-based features in this work.

The detection of image patches consists of detecting those image regions that are considered to be of interest. Many approaches have been proposed, such as the extraction of local-patches at keypoints, the extraction of local-patches on sliding grids or at random, and the segmentation of an image. These locally detected patches are then described by various local descriptors, such as the pixel gray values, the histograms of oriented gradient (HOG) (Dalal and Triggs, 2005), the SIFT and the speed-up robust features (SURF) (Bay et al., 2008). The construction of a codebook involves running a clustering algorithm, usually the traditional K-means clustering algorithm (Hartigan and Wong, 1979), over a large set of training patches. In K-means based methods, a codeword is presented by the centroid (average of all visual keypoints that belongs to this cluster) or, within a probabilistic framework, the codewords can be presented by the Gaussian mixture models (GMMs).

Features extracted from images are then coded using vector quantization against the learnt codebook, and the resulting 'votes' for each codeword are tallied to produce a fixed-length histogram, characterising the distribution of patches over the image or local regions. The process of assigning a single codeword to a single image features is referred to as hard-assignment. Instead of using hard-assignment, each patch can be assigned to all codewords in a probabilistic manner, i.e. assign weights to neighbouring visual words. The latter process is referred to as soft-assignment. Finally, a classifier (k-nearest neighbour or support vector machine) is trained on the histogram representation of the images to discriminate between the presence or absence of an object class.

The classification performance of such a visual object recognition system strongly depends on the effectiveness of the visual codebook as it plays a central role that affects the model's complexity (Wu and Rehg, 2009a; van Gemert et al., 2010). As mentioned above, the popular way of constructing a codebook is achieved by the traditional K-means clustering (Csurka et al., 2004; Winn et al., 2005; Zhang et al., 2007). There are several known difficulties with the use of K-means clustering in this context that we discuss in detail in section 4.2.1, and this thesis demonstrates a particular way of circumventing these difficulties.

In practice, the construction of a visual codebook is often performed from thousands of images and each image on average contains hundreds or even one thousand of patch-based interest points described in a higher dimensional space of one hundred, in order to capture sufficient information for efficient classification. While clustering algorithms and their performance characteristics have been studied extensively over recent years, a major bottleneck lies in handling the massive scale of the datasets. The Caltech[2] and PASCAL VOC Challenge image datasets are becoming gold standard for measuring recognition performance in recent vision papers, but the size of these datasets nearly grows exponentially over the years, which can be seen from the data statistics summarised in Table 1.1.

For example, the SIFT features extracted for constructing a visual codebook using the Caltech-101 (Fei-Fei et al., 2004) dataset by using 30 images per object category are approximately $2\text{M}\times\mathbb{R}^{128}$, while the Caltech-256 with 75 images per object category is around $14\text{M}\times\mathbb{R}^{128}$. Over the timelines, the number of object categories and the minimum number of images per object categories have been increased more than twice. As a consequence, the local-patch based features have become seven times larger in size. At these scales of data, classical clustering algorithms such as K-means clustering are not straightforward to apply and a need for novel approaches arises. Similarly during the classification step, an enormous amount of time is spent in tuning the parameters and classifying the unseen patterns.

[2] http://www.vision.caltech.edu/Image_Datasets/

TABLE 1.1: Summary of the Caltech and PASCAL VOC Challenge image datasets. There are actually 102 and 257 categories in the Caltech image datasets if the 'clutter' categories in each set are included. We provide the statistics of the PASCAL VOC Challenge image datasets where there is an increase in object categories.

Dataset	**Released**	**#Categories**	**#Images**
Caltech-101	2003	102	9144
Caltech-256	2006	257	30607
PASCAL VOC'05	2005	4	1373
PASCAL VOC'06	2006	10	5304
PASCAL VOC'09	2009	20	13704

In order to improve a few of these problems we demonstrate two novel sequential designs for clustering: One approach that we refer to is a resource-allocating codebook (RAC) method, inspired by the resource-allocation network (RAN) algorithms developed in the artificial neural networks literature (Platt, 1991; Kadirkamanathan and Niranjan, 1993; Yingwei et al., 1997), and the other approach that we refer to is a sequential hierarchical clustering (SHC) method. The RAC has strong similarities to the mean-shift based clustering technique (Jurie and Triggs, 2005), but has some advantages which we demonstrate in chapter 2 and chapter 4 of this thesis. In classification, we demonstrate a particular novel approach that we call unbalanced decision tree (UDT) based on SVMs that mainly relieves the excessive testing time incurred in the directed acyclic graph (DAG) (Platt et al., 2000) technique.

1.3 Contribution

The main contribution of this work in cluster analysis is to demonstrate a novel approach that can sequentially process a large number of patch-based visual descriptors in a higher dimensional feature space to constructing codebooks for reliable visual object categorisation performance at drastically reduced computational needs. Along the way we also demonstrate a novel hierarchical clustering technique that sequentially processes the data in a one-pass setting with nearly the same efficiency as the traditional hierarchical clustering method. For classification, we demonstrate a multi-class classification architecture that has been sensibly combined using existing ideas to relieve the excessive time in classifying the test data. A short description of those proposed approaches follows.

1.3.1 Resource-Allocating Codebook (RAC)

We formulate the problem of constructing a discriminant codebook for visual object recognition by proposing a novel approach that we call Resource-Allocating Codebook (RAC) approach. The RAC is a simple and extremely fast way that constructs a codebook as a one-pass process which simultaneously achieves increased discrimination and a drastic reduction in the computational needs. This technique is comparable to or outperforms the codebook constructed by the traditional K-means clustering in the state-of-the-art visual object recognition systems. RAC seems particularly suitable for the codebook construction owing to its simplicity, speed and performance and its one-pass strategy that requires relatively little memory.

1.3.2 Sequential Hierarchical Clustering (SHC)

A common requirement amongst many existing clustering methods is that all pairwise distances between patterns must be computed in advance. This makes it computationally expensive and difficult to cope with large scale data used in several applications, such as in computer vision and bioinformatics. To address this issue, we demonstrate a novel Sequential Hierarchical Clustering (SHC) technique that initially builds a hierarchical tree from a small fraction of the entire data, while the remaining data are processed sequentially and the tree is adapted constructively. Our experimental results show that SHC does not degrade in performance while reducing the computational needs.

1.3.3 Unbalanced Decision Tree (UDT)

We demonstrate a new learning architecture that we call Unbalanced Decision Tree (UDT) for multi-class pattern classification, attempting to improve existing methods based on DAG and One-versus-All (OVA) SVMs. UDT implements the OVA-based concept (Vapnik, 1995) at each decision node. This technique relieves the problem of excessive testing time incurred by DAG while maintaining accuracy comparable to those standard techniques employed by SVMs. UDT is general, and could be applied to any classification task in machine learning in which there are natural groupings among the patterns. Apart from computer vision applications, we also evaluated the UDT on applications such as high throughput genomic data analysis generate classification tasks, where the number of classes is higher and the amount of data available per class is lower than in widely used benchmark datasets.

1.4 Thesis organisation

After this introductory chapter, this thesis is organised as follows. Chapter 2 reviews various codebook models that have been used in the literature to categorise visual objects. Chapter 3 presents the widely used visual descriptors in a patch-based visual object recognition framework. Chapter 4 provides details of exploratory work on clustering techniques and demonstrates two novel approaches, namely RAC and SHC methods. The central theme behind all of the methods described in this chapter is that of handling large scale data in an efficient way. In this section, the properties and variants of RAC are discussed in detail. In addition to this, we explore the use of RAC in incrementally constructing a discriminant codebook with classifier training. We extended the sequential idea of RAC to the traditional hierarchical clustering by demonstrating the SHC that initially builds a hierarchical tree from a small fraction of the entire data. In the remainder of this chapter, we explore another approach to construct a discriminant codebook based on the Fisher score, instead of applying any clustering techniques. In chapter 5, the multi-class classification process is explained mainly by addressing a particular novel approach UDT using support vector machines. Both chapters 4 and 5 provide a brief description of the datasets that were used in empirically supporting our claims. Finally, in chapter 6 we draw conclusions about our approaches and discuss future work.

1.5 Publications

The work in this thesis has contributed in part to the following publications:

- Chapter 2

  – Ramanan, A. and Niranjan, M. "A Review of Codebook Models for Patch-based Visual Object Recognition", In submission to the Journal of Signal Processing Systems, 2010.

- Chapter 4

  – Ramanan, A. and Niranjan, M. "A One-pass Resource-Allocating Codebook for Patch-based Visual Object Recognition", In IEEE International Workshop on Machine Learning for Signal Processing (MLSP'10), 2010. [accepted]

  – Ramanan, A. and Niranjan, M. "Resource-Allocating Codebook for Patch-based Face Recognition", In Proceedings of the IEEE International Conference on Industrial and Information Systems (ICIIS'09), pp. 268-271, 2009.

  – Ramanan, A. and Niranjan, M. "On the Construction of a Discriminant Codebook for Visual Object Recognition", In British Machine Vision Conference (BMVC'09) Workshop, 2009. [poster paper]

– Farran*, B., Ramanan*, A. and Niranjan, M. "Sequential Hierarchical Pattern Clustering", In Proceedings of the IAPR International Conference on Pattern Recognition in Bioinformatics (PRIB'09), LNBI 5780, pp. 79-88, 2009.

- Chapter 5

  – Ramanan*, A., Suppharangsan*, S., and Niranjan, M. "Unbalanced Decision Trees for Multi-class Classification", In Proceedings of the IEEE International Conference on Industrial and Information Systems (ICIIS'07), pp. 291-294, 2007.

$*$ the first two authors made equal contributions.

Chapter 2

A review of codebook models in visual object recognition

There is an extensive body of literature on the area of visual object recognition systems. However, a straightforward but effective approach lies in the use of the bag-of-features or codebook model. As the design of this codebook is the computational and performance bottleneck in such systems, we restrict our survey to this popular codebook model. In this chapter, we review several approaches that have been proposed over the last decade with their use of feature detectors, descriptors, codebook construction schemes, choice of classifiers in recognising objects, and datasets that were used in evaluating the proposed methods.

Several combinations of image patch detectors and descriptors, different features, matching strategies, various clustering methods and classification techniques have been proposed for visual object recognition. Assessing the overall performance of the individual components in such systems is difficult, since the computational requirements and the fine tuning of the different parts become crucial. The well-known framework in the literature uses the SIFT descriptors (Lowe, 2004) to describe the patches and cluster them using the standard K-means algorithm, in order to encode the images as a histogram of visual codewords as originally proposed by Sivic and Zisserman (2003), Willamowski et al. (2004), and Csurka et al. (2004). This visual codebook model approach, inspired by the bag-of-words approach used in text retrieval, has shown state-of-the-art categorisation performance (Csurka et al., 2004; Zhang et al., 2007; Winn et al., 2005; Li et al., 2008; Wu and Rehg, 2009a).

The approach to constructing patch-based visual codebooks can be achieved by the use of clustering techniques in an unsupervised manner, completely ignoring the location of the detected interest points, or by methods that incorporate the spatial information of the features. The first approach is straightforward and powerful that entirely relies on appearance information. This has an advantage in that the object categories do not

need to be segmented prior to recognition. The latter approach is a probabilistic model known as the constellation model (Fei-Fei and Perona, 2005; Larlus and Jurie, 2006; Quelhas et al., 2007) that additionally represents the spatial layout of the parts in the model. Many of these methods require minimal supervision, i.e. manual segmentation of the objects, within the training images. However if thousands of classes are to be learnt then these approaches become more difficult to process.

As mentioned above, a popular approach to constructing a visual codebook is the use of cluster analysis, usually undertaken by applying the traditional K-means method (Sivic and Zisserman, 2003; Csurka et al., 2004; Fei-Fei and Perona, 2005; Winn et al., 2005; Li et al., 2008; Sudderth et al., 2008). Several other clustering techniques have been employed to construct visual codebooks: Nister and Stewenius (2006) and Mikolajczyk et al. (2006) used hierarchical K-means clustering, whereas Leibe and Schiele (2003) used agglomerative clustering. Moosmann et al. (2007) used randomised clustering forests, whereas Farquhar et al. (2005), Dorko et al. (2005), Larlus and Jurie (2006) and Perronnin (2008) used Gaussian Mixture Models (GMMs). In the remainder of this chapter we review these approaches in detail.

Sivic and Zisserman (2003) proposed an approach to retrieve visual objects and scenes from a movie using a text retrieval approach. Local regions were extracted from each frame in the video in the following two different ways: One method is referred to as a shape-adapted (SA) region which surrounds an interest point by an elliptical shape. The second method is referred to as a maximally stable (MS) region which is constructed by intensity watershed image segmentation. The SA regions are detected on corner like regions and the MS regions correspond to blobs of high contrast with respect to the surroundings. Both SA and MS regions are then described by SIFT descriptors. The authors were aware of the difficulty in clustering a very large scale of descriptors extracted from their movies, so instead they selected 10000 frames which represent about 10% of all the frames in the movie, resulting in 200000 averaged track descriptors to construct a codebook. A visual codebook is constructed using K-means clustering algorithm, and Mahalanobis distance measure. The Mahalanobis distance function between two patch-based visual descriptors $\mathbf{x}$ and $\mathbf{y}$ of the same distribution with the covariance matrix Σ, is given by:

$$d(\mathbf{x}-\mathbf{y}) = \sqrt{(\mathbf{x}-\mathbf{y})^T \Sigma^{-1} (\mathbf{x}-\mathbf{y})} \tag{2.1}$$

A discussion on this distance measure (equation 2.1) when used with SIFT descriptors is provided in chapter 6. K-means was run several times with different sets of initial cluster centres to maximise retrieval results. The codebook constructed using SA features was about 6000 and the codebook using MS features was about 10000. The ratio of the size of codebooks for each type is chosen nearly to the ratio of detected descriptors of each type. The collections of codewords are used in the term frequency-inverse document

frequency (*tf-idf*) scoring of the relevance of an image to the query. The *tf-idf* scoring is used in information retrieval and text mining. The *tf* term measures the number of occurrences of a particular codeword in the example divided by the total number of patch-based features in the example. The *idf* term measures the distinctiveness of a particular codeword over different examples. The performance was evaluated on two feature films: 'Run Lola Run' and 'Groundhog Day'. The authors have constructed codebooks sufficient for two films in a very computationally expensive way, which makes it hard to apply by using the K-means method for a large number of films.

Leibe and Schiele (2003) used the Harris interest point detector (Harris and Stephens, 1988) to extract image patches. The pixel gray values of those patches are then clustered using the agglomerative clustering method (see chapter 4 under section 4.2.3) to generate a visual codebook. The size of the learnt codebook was further reduced by merging the most similar clusters in a pair-wise manner when the similarity between clusters exceeds a predefined threshold t. The similarity between two clusters C_1 and C_2 was measured by the normalised grey-value correlation (NGC).

$$similarity\,(C_1, C_2) = \frac{\sum_{x \in C_1, y \in C_2} NGC(x, y)}{|C_1| \times |C_2|} \tag{2.2}$$

where,

$$NGC\,(x, y) = \frac{\sum_i (x_i - \overline{x_i})(y_i - \overline{y_i})}{\sqrt{\sum_i (x_i - \overline{x_i})^2 \sum_i (y_i - \overline{y_i})^2}} \tag{2.3}$$

Instead of assigning image patches to their nearest codeword in the learnt codebook, every patch casts probabilistic votes to the codebook using the NGC measure whose similarity is above t. For classification, a generalised Hough transform-like (Lowe, 1999) voting scheme is applied. The proposed method was evaluated on a database of 137 images of scenes containing one car each in varying poses. The size of the codebook was around 2500.

Csurka et al. (2004) used the Harris affine region detector (Mikolajczyk and Schmid, 2002) to identify the interest points in the images which are then described by SIFT descriptors. A visual codebook was constructed by clustering the extracted features using K-means method. Images are then described by histograms over the learnt codebook. The authors run the K-means several times over a selected size of K and different sets of initial cluster centres. The reported results were the clusters that gave them the lowest empirical risk in classification. The size of the codebook used in reporting the results is 1000. The authors compared Naive Bayes and Support Vector Machine (SVM) classifiers in the learning task and found that the one-versus-all SVM with linear kernel gives a significantly (i.e. 13%) better performance. The proposed framework was mainly evaluated on their 'in-house' database that is currently known as 'Xerox7' image set containing 1776 images in seven object categories. The overall error rate of the classification is 15% using SVMs. The RAC approach described in section 4.3 applied on

the Xerox-7 image dataset, performs slightly better than the authors' method but was achieved in a tiny fraction of computation time.

Farquhar et al. (2005) proposed alternatives to the scheme introduced by Csurka et al. (2004). The Gaussian mixture model (GMM) was proposed as a replacement of the K-means based codebook construction, and summed responsibility replacing bin membership for histogram generation. The GMMs were all trained for category-specific codebooks and were then combined into a single codebook. Features were extracted using multi-scale Harris affine region detector that are then described by SIFT descriptors. The features were pre-processed to reduce its dimensionality. The authors used two different methods to reduce dimensions: the PCA and partial least squares (PLS), and found that PLS improves classification performance over the PCA method for the same number of reduced dimensions. The proposed method was also tested on the Xerox7 image dataset used by Csurka et al. (2004). The classification results were obtained by using one-versus-all SVM classifiers with linear kernel. Although about 2% of improvement was obtained over the original results of Csurka et al. (2004), the concatenation of category-specific codebooks into a single codebook approach is impractical for a large number of visual object categories, as the size of the concatenated codebook grows linearly with the number of classes. When the number of classes increases, not only does it increase the computational cost but it also makes the classification of histograms challenging due to its diverse range in object classes.

Fei-Fei and Perona (2005) proposed a Bayesian hierarchical model that represents the distribution of codewords in each category of natural scenes as a mixture of aspects. Each aspect is defined by a multinomial distribution over the quantized local descriptors. Their method is modified on the latent Dirichlet allocation (LDA) model (Blei et al., 2003) by introducing a category variable for classification, which explicitly requires each image example to be labelled during the learning process. A graphical representation of the modified LDA is depicted in Figure 2.1. The authors tested four different ways of extracting local regions: evenly sampled grid, random sampling, the Kadir and Brady Saliency detector (Kadir and Brady, 2001) and difference of Gaussian (DoG) detector (Lowe, 1999). The patches are then described by two different methods: normalised 11×11 pixel gray values or SIFT. Features extracted from all training images of all categories were clustered by the K-means algorithm. Following the construction of the codebook, clusters with too small number of members were pruned out. The dataset they used for evaluation contained 13 categories of natural scenes with 3859 images that were collected from a mixture of COREL images, Google image search engine and personal photographs. Based on their experimental results, they report that the SIFT representation is more robust than the pixel gray value representation. Furthermore, the evenly sampled grid-based SIFT approach out performs the random, saliency, and DoG based SIFT approaches by 4.5%, 12.1% and 12.7%, respectively.

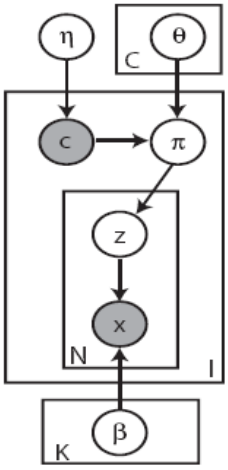


FIGURE 2.1: Image from (Fei-Fei and Perona, 2005). The graphical representation of the modified latent Dirichlet allocation model. An image I consists of N patches denoted by $\mathbf{x}$. The total number of object categories is C. η is a C-dimensional vector of a multinomial distribution, and π is the parameter of a multinomial distribution. K is the total number of themes. θ is a parameter conditioned on the category c. $\mathbf{x}$ and $\mathbf{c}$ are observed variables.

Jurie and Triggs (2005) proposed a mean-shift based clustering approach to construct codebooks in an undersampling framework. Our resource-allocating codebook (RAC) approach presented in this thesis has strong similarities to this technique. A detailed description of the mean-shift clustering method is given in section 4.2.2. The authors sub sample patches randomly from the feature set and allocate a new cluster centroid for a fixed-radius hypersphere by running a mean-shift estimator on the subset. The mean-shift procedure is achieved by successively computing the mean-shift vector of the sample keypoints and translating a Gaussian kernel on them. In the next stage, visual descriptors that fall within the cluster are filtered out. This process is continued by monitoring the informativeness of the clusters or until a desired number of clusters is achieved.

The features used in the experiments are the gray level patches, sampled densely from multi-scale pyramids with 10 layers. Three different feature selection methods proposed by Brank et al. (2002) were used in the experiments: (i) maximisation of mutual information, odds of ratio, and training an initial linear SVM on the entire training set and selection of the features that have the highest weight. Two different ways of producing fixed-length feature vectors from the learnt codebook were used in the experiments: Binary indicator vectors which were produced by thresholding the frequency counts of the codeword in the image, and the histograms. The proposed method was evaluated on three datasets: Side views of cars from Agarwal et al. (2004), Xerox7 image dataset (Csurka et al., 2004), and the ETH-80 dataset (Leibe and Schiele, 2003) containing four object categories (cars, horses, dogs and cows) each with 205 images. Naive Bayes and linear SVM classifiers were compared in all their experiments. The size of the codebook was 2500.

Based on the obtained experimental results, the authors conclude the following: (i) the initial training with linear SVM in feature selection was better however, full codebooks generally outperformed compact codebooks, (ii) mean-shift based codebooks outperformed K-means based codebooks, (iii) histogram representation performs better than binary indicators, and (iv) linear SVMs easily outperform Naive Bayes classifiers. The authors' mean-shift based clustering method is computationally intensive in determining the cluster centroid by mean-shift iterations at each of the sub samples. The convergence of such a recursive mean-shift procedure greatly depends on the nearest stationary point of the underlying density function and its utility in detecting the modes of the density. Also efficient computation of the mean-shift method requires the sub sampling of visual keypoints with a regular grid and the selection of the bandwidth. A technique that has many parameters can overfit data and generalise poorly (Platt, 1999). In contrast, the RAC approach pursued in this thesis has a single threshold that takes only one-pass through the entire data, making it computationally efficient.

Winn et al. (2005) optimised codebooks by hierarchically merging visual words in a pair-wise manner using the information bottleneck principle (Tishby et al., 1999) from an initially constructed large codebook. The final visual words are represented by the GMMs of pixel appearance. Training images were convolved with different filter-banks made of Gaussians and Gabor kernels. The resulting filter responses were clustered by the K-means method with a large value of K in the order of thousands. Mahalanobis distance between features is used during the clustering step. The learnt cluster centres and their associated covariances define a universal visual codebook. Following the construction of this large codebook, each region of the training images is processed to compute the histogram $\mathbf{h}$ over the initial codebook and the corresponding histogram $\mathbf{H}$ of target codewords. A mapping function $\mathbf{H} = \phi(\mathbf{h})$ is used to produce a much more compact visual codebook, where ϕ is the pair-wise merging operation that acts on the initial codewords. Classification results were obtained on photographs acquired by the authors, images from the web and a subset of 587 images in total that were selected from the PASCAL VOC challenge 2005 dataset containing four classes. Gaussian class models were compared with multi-modal nearest neighbours in classification. Their class models were learnt from a set of manually segmented photographs into object-defined regions. Even though the authors claim that the proposed technique is simple and extremely fast, the complex learning process i.e. the initial codebook construction based on K-means clustering and the merging of visual words make it harder to apply on large number of features. However, if two distinct visual words are initially grouped in the same cluster, they cannot be separated later. Also the vocabulary is tailored according to the categories under consideration, but it would require fully retraining the framework on the arrival of new object categories, whereas the RAC technique demonstrated in this thesis can cope with new object categories without retraining the whole system.

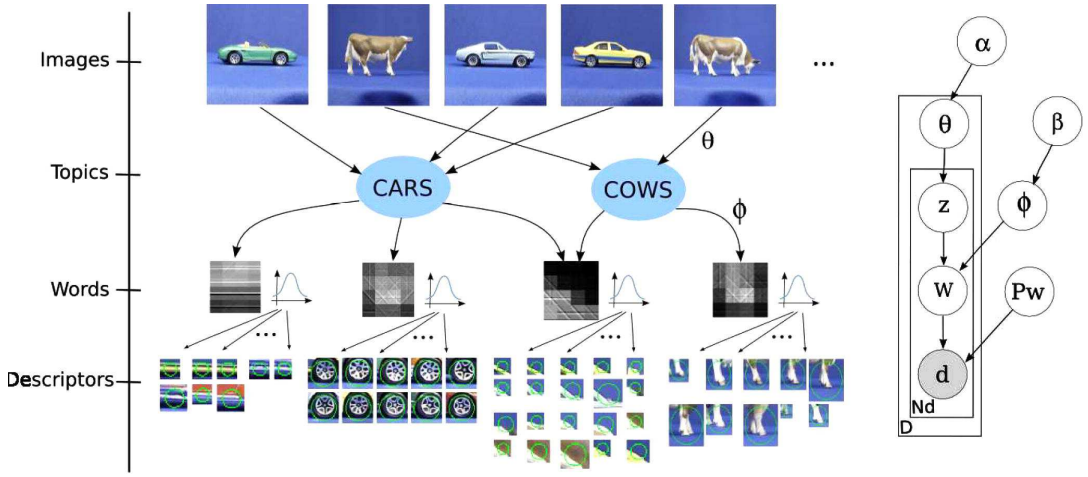


FIGURE 2.2: Image from Larlus and Jurie (2006). Overview of the latent mixture codebook model and the corresponding graphical model representation.

Larlus and Jurie (2006) proposed a generative model based on latent aspects that represent images at low-level feature descriptors. The construction of a visual codebook is achieved by an object model that embeds visual words as a component of the learning process. In their model, images are treated as distributions of topics, topics are considered as distributions of visual words, and visual words considered as Gaussian mixtures over SIFT descriptors. Figure 2.2 depicts the proposed model.

This latent variable model of Larlus and Jurie (2006) is a form of Gaussian-Multinomial latent Dirichlet allocation (GM-LDA). Topic distributions over words are sampled from a Dirichlet distribution. Compared to the model in (Fei-Fei and Perona, 2005), GM-LDA has an extra layer responsible for the generation of visual descriptors conditional to visual words that allows for learning the visual codebook. The model parameters are estimated by an iterative technique called Gibbs sampling. Experiments were carried out on two datasets: a subset of the ETH-80 dataset (Leibe and Schiele, 2003) containing four object categories and the Bird dataset (Lazebnik et al., 2005a) containing six categories each with 100 images. Local descriptors were extracted on a dense grid at different scales and each patch was represented by SIFT descriptor. The experiments, using the proposed model under different settings compare image categorisation based on the latent topics and visual features in a bag-of-features framework. Also the standard codebook model using K-means and the standard LDA model are compared with their model. The topic-based classification was compared with SVM classifiers and a Bayesian type classifier in which the authors note that both of the classifiers perform equally. The bag-of-features based classification employs SVM classifiers, from which the authors conclude that the GM-LDA is better than the K-means based method, and the bag-of-features approach compared to topic-based classification performs much better. As the proposed model has four parameters, its estimation is much more time consuming than a standard LDA or K-means clustering method.

Mikolajczyk et al. (2006) find local features by extracting edges with a multi-scale Canny edge detector (Canny, 1986) with Laplacian-based automatic scale selection. For every feature, a geometry term gets determined, coding the distance and relative angle of the object centre to the interest point, according to the dominant gradient orientation and the scale of the interest point. These regions are then described with SIFT features that are reduced to 40 dimension via PCA. The visual codebook is constructed by means of a hierarchical K-means clustering. Initially the features are clustered using K-means algorithm and then agglomerative clustering is performed to obtain compact feature clusters within each partition. Given a test image, the features were extracted and a tree structure is built using the hierarchical K-means clustering method in order to compare with the learnt model tree. Classification is done in a Bayesian manner computing the likelihood ratio. This test is done at local maxima of the likelihood function of the object being present. Some additional tests are applied to determine whether objects of different classes share similar clusters or whether overlapping objects exist. In this manner, the location, scale and orientation of multiple objects can be determined. Experiments were performed on a five class problem taken from the PASCAL VOC 2005 image dataset containing four classes and a RPG (rocket-propelled grenade) shooter that was collected from various sources.

Nister and Stewenius (2006) proposed a hierarchical K-means clustering that constructs a vocabulary tree in an offline training stage for image retrieval from a large database. Features were extracted using maximally stable extremal regions (MSERs) (Matas et al., 2002) which are then described by SIFT descriptors. SIFT features were then quantized with the vocabulary tree. The vocabulary tree is constructed by a hierarchical scoring scheme based on the term frequency-inverse document frequency (tf-idf) score. First, an initial K-means process is run on the training data, defining K centroids. The training data is then partitioned into K groups, where each group consists of the features closest to a particular centroid. The second step is then recursively processed by quantizing each node into K new parts, where K defines the number of children of each node. The tree is constructed level-by-level up to a maximum number of levels. Following the recursive process, in the online phase, each visual descriptor is propagated down the vocabulary tree by coding the closest node at each level. The proposed technique was tested on a ground truth database containing 6376 images in groups of four of the same object but under different conditions. From their experimental results, they found that larger vocabulary (between 1 and 16 million leaf nodes) improves retrieval performance. They claim that this methodology provides the ability to make fast searches on extremely large databases (i.e. one million images).

Quelhas et al. (2007) have extended the work of Fei-Fei and Perona (2005) for scene classification that integrates scale-invariant feature extraction and probabilistic latent semantic analysis (PLSA)-based clustering of images. Images are modelled as mixtures of aspects in an unsupervised way. The distribution over aspects serves as image repre-

sentation that is inferred from new images and then used for classification. The visual codebook was constructed by the K-means algorithm with a desired choice of K, typically K=1000. Following the construction of the codebook, the authors use the PLSA model to capture co-occurrence information between elements in the bag-of-features representation. The parameters of the PLSA model are estimated using the maximum likelihood principle. They compare different feature detectors: DoG, multi-scale Harris affine, multi-scale Harris, and a fixed 15×20 grid and three different descriptors: SIFT, complex filters, and a 11×11 pixel sample of the area defined by the detector were used in paired combinations. The main experiments were tested on two datasets, one used in (Fei-Fei and Perona, 2005) and the other on six natural scene classes containing a total of 700 images. The classification results were obtained by one-versus-all SVMs with Gaussian kernel. The authors' experimental results confirm that in practice DoG+SIFT constitutes a reasonable choice for image scene classification.

Wang (2007) proposed the construction of a discriminant codebook at a multi-resolution level using a hierarchical clustering technique and then use a boosting feature selection method to select the discriminant codewords. Features were extracted using the Harris affine interest point detector and SIFT descriptor. The extracted patch descriptors are clustered into a sufficiently large number of clusters (e.g. 2000). These clusters are then hierarchically clustered in a bottom-up way to generate new clusters in each level. Centroids of these clusters form a multi-resolution codebook that is usually very large as it includes more resolution levels. To reduce the size of the codebook, discriminant codewords are selected by a threshold-based boosting feature selection technique. To do this, frequency histograms of the training images are sorted according to a histogram feature. Using the threshold through the sorted list, the weak classifier giving the minimal training error is selected and the corresponding codeword in the codebook is indicated to be inactive. The choice of classifier was the Kernel Fisher Discriminant Analysis (KFDA) with the RBF kernel. Their method is evaluated against a selected four class problem (motorbikes, airplanes, faces_easy, and background_Google) from the Caltech-101 image dataset. However, this method involves greater computation and suffers from the difficulty in identifying the optimal value of the size of an initial codebook.

Zhang et al. (2007) compare sets of local features in two different methods. Their first method involved clustering a set of patch-based descriptors in each image to form a representation of (c_i, w_i) pairs, that they refer to as image *signature* where c_i is the cluster centre and w_i is the proportional size of the i^{th} cluster. Cluster centres were obtained using K-means algorithm with K=40. Earth Mover's Distance (EMD) (Rubner et al., 1998) was the choice for measuring similarities between image representations. The EMD between two image signatures $S_1 = \{(p_1, u_1), \cdots, (p_m, u_m)\}$ and $S_2 = \{(q_1, w_1), \cdots, (q_n, w_n)\}$ is defined as:

$$D\left(\mathbf{S}_1, \mathbf{S}_2\right) = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij}\; d\left(p_i, q_j\right)}{\sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij}} \tag{2.4}$$

where, f_{ij} is a flow value that is usually determined by solving a linear programming problem, and $d\left(p_i, q_j\right)$ is the ground distance (e.g. Euclidean distance) between cluster centres p_i and q_j.

The second method was clustering the patch-based descriptors from a training set to construct a global codebook and then represent each image as a frequency histogram. The global codebook was also constructed by K-means method. χ^2 distance measure was used in this case to compare two histograms $\mathbf{x}$ and $\mathbf{y}$, which is defined as:

$$D\left(\mathbf{x}, \mathbf{y}\right) = \frac{1}{2}\sum_{i=1}^{m}\left[\frac{\left(x_i - y_i\right)^2}{x_i + y_i}\right] \tag{2.5}$$

Interest points were detected using the Harris and Laplacian detector, and were compared with different invariance properties: Scale invariance only, scale with rotation invariance, and affine invariance. The SIFT and/or SPIN (Lazebnik et al., 2005b) descriptors were used to describe the interest points found by different detectors as mentioned above. Each detector/descriptor pair is considered as a separate channel at the classifier stage. One-versus-one SVM classifiers were compared with three different kernels: linear, χ^2, and the EMD. Their experimental evaluations were performed on four texture (UIUCTex, KTH-TIPS, Brodatz, and CUReT) datasets and five object category (Xerox7, Caltech-6, Caltech-101, Graz, and PASCAL VOC 2005) datasets. Based on their experiments they conclude that the combination of Harris and Laplacian detectors with SIFT and SPIN descriptors is the preferable choice in terms of classification performance together with the choice of the χ^2 kernel. The χ^2 kernel performs better than the linear one and at the same time it is comparable with the EMD kernel.

Kim and Kweon (2007) proposed a technique to reduce the size of a codebook and enhance its discriminative power by eliminating some visual codes from the codebook using an entropy-based minimum description length (MDL) criterion. This process involves the construction of intra-class and inter-class codebooks. The intra-class codebook is initially constructed for each object category using an agglomerative K-means clustering method. The MDL of each category-specific codebook is then computed. If the MDL is not minimum then the codebook that has the lowest entropy is removed. Following this step, the inter-class entropy of a codebook that has large entropy is removed from the intra-class codebook yielding the inter-class codebook. Kim and Kweon (2007) used their own feature that they refer to as the generalised robust invariant feature (G-RIF) (Kim and Kweon, 2006). The 189 dimensional G-RIF was reduced to five dimensions via PCA. In their first experiment, an intra-class codebook was used to compare SVMs with nearest neighbour classifiers using different distance measures: KL-divergence, χ^2,

histogram intersection and Euclidean distances. Based on the experimental results, they found that the NN with KL-divergence gives better performance than SVMs. This might be the case as a small set of 15 training samples from each category was used in training the SVMs. Also it is reported that the directed acyclic graph (DAG) SVMs for multi-class classification performed worse than one-versus-all SVMs. A selected 10 object category from the Caltech-101 image dataset was used in their first experiment. In the second experiment, an inter-class codebook was used to evaluate the classification performance of the entire Caltech-101 image dataset (15 training and 15 images for testing) using nearest neighbour with KL-divergence distance metric. However, there is a large amount of computation involved in constructing both the intra and inter-class codebooks and the resized codebook is not optimally compact.

Moosmann et al. (2007) introduced extremely randomised clustering (ERC) forests to construct a visual semantic codebook. Initially a tree is built using random forests (Breiman, 2001). This tree is used as a spatial partitioning method by assigning each leaf of each tree a visual word, which is how a semantic visual codebook is constructed, instead of using it as a classifier. Compared to random forests using C4.5 (Quinlan, 1993), extremely randomised trees are faster to construct. Different types of features were used in their experiments: an HSL (hue, saturation, and lightness) colour descriptor of 768 dimensions (16×16 pixels × 3), a Haar wavelet-based colour descriptor that transforms this into another 768 dimensions, and SIFT descriptor of 128 dimension. A detailed experimental piece of work was carried out with a Graz-02 image dataset[1] containing three object categories (bicycles, cars and persons) and negatives (i.e. none of the three object categories are present). The PASCAL VOC challenge 2005 image dataset and a horse database[2] were also used in evaluating their method. The sizes of the codebooks used with Graz-02 and PASCAL VOC 2005 are 5000 and 30000, respectively. A linear SVM classifier was employed in the classification tasks. However, this approach creates a very large codebook which has difficulty in coping with large datasets. In addition, it can lead to overfitting.

Sudderth et al. (2008) developed a family of hierarchical probabilistic models for object recognition in natural scenes. Visual objects are modelled as a set of parts with an expected appearance and position, in an object-centred coordinate frame. The authors started developing models for images with single objects, and models which share parts among related categories, and finally turned to multiple object scenes through the use of Dirichlet processes. They extracted interest regions from images using three different criteria: Harris affine invariant regions, Laplacian of Gaussian operator (Lowe, 2004) and the maximally stable extremal regions (MSERs) (Matas et al., 2002) algorithms that were then described by SIFT descriptors. Edge-based features were also extracted using the Canny detector (Canny, 1986). K-means clustering was used to construct a visual codebook of size 1000, where the K was set by cross-validation. Each of the three

[1] http://www.emt.tugraz.at/∼pinz/data/
[2] http://pascal.inrialpes.fr/data/horses/

different feature types is then mapped to a disjointed set of visual words. An expanded codebook then jointly encodes the appearance and coarse shape of each feature. The parameters of the models are learnt via a Gibbs sampler which uses a graphical model to analytically average over many parameters. They evaluated the model on a collection of 16 categories containing seven animal faces, five animal profiles and four wheeled vehicles as object categories and also evaluated the model on a simple street scene containing three object categories (buildings, cars, and roads). Classification is undertaken using the likelihood ratio. The approach only works for images with roughly aligned objects, as in the Caltech 101 object database.

Li et al. (2008) proposed the construction of a discriminant codebook in a similar fashion to that proposed by Winn et al. (2005) and Wang et al. (2008), i.e. constructing a compact codebook through selecting a subset of codes from an initially learnt large codebook. An initial codebook was constructed using K-means clustering algorithm. Each codeword in this codebook is then modelled by a spherical Gaussian function through which an intermediate representation for each training image is obtained. A Gaussian model for every object category is learnt based on this intermediate representation. Following this step, an optimal codebook is constructed by selecting discriminant codes according to the learnt Gaussian model. The discriminative capability is measured either by likelihood ratio or by Fisher score. Interest points in their experiments were detected by the DoG detector and were described by SIFT descriptors. Classification was performed using SVM classifiers with RBF kernel. The authors claim that the likelihood ratio performs better than the Fisher score as it fits their classification problem. This method was evaluated on the Caltech-4 object dataset containing four object categories within a total of 2876 images and a background class with 450 images. All the images for training or test were scaled to 300 pixels in width. They also carried out experiments with different codebook sizes using the algorithm proposed in Csurka et al. (2004) using the Caltech-4 dataset. The highest classification rate achieved was 91.5% with a codebook size 900. The best performance of Li et al. (2008)'s method was 90.5%, that was achieved with a more compact codebook of size 100, where the optimal codes were selected from an initial codebook of size 1400. Although effective, it still suffers from the disadvantages caused by K-means clustering in the construction of an initial large codebook.

Perronnin (2008) characterised images using a set of category-specific histograms generated one per object category, where each histogram describes whether the content can be best modelled by a universal vocabulary or by its corresponding category-specific codebook. A universal codebook describes the visual content of all the considered categories that are trained with data from all classes under consideration and a codebook is represented by GMMs using maximum likelihood estimation. On the other hand, category-specific codebooks are obtained by adapting the universal codebook using the class training data and a form of Bayesian adaptation based on the maximum a poste-

riori (MAP) criterion. The maximum number of Gaussians in the universal codebook was set to 2048. An image is then characterised by a set of histograms called bipartite as they can be split into two equal parts. Each part describes how well one codebook accounts for an image compared to the other codebook. Local patches were extracted from regular grids at five different scales. Each patch is then described by SIFT and colour features. PCA was applied to reduce the dimensionality of SIFT from 128 to 50, and the RGB colour channels from 96 to 50. Evaluations were performed on their own in-house database containing 19 classes of object categories and scenes, and the PASCAL VOC 2006 image dataset containing 10 classes. Classification was performed using one-versus-all linear SVMs and a logistic regression with a Laplacian prior. However, in this approach, if two visual object classes are visually close, there is no guarantee that a distinctive visual word will be obtained. On the other hand, the process that generates bipartite histograms is computationally expensive.

Yang et al. (2008) proposed a unified codebook generation that is integrated with classifier training. Unlike clustering approaches that associate each image's low-level features with a single codeword in their approach (see Figure 2.3) images are represented by means of visual bits associated with different categories, i.e. an image which can contain objects from multiple categories is represented using aggregates of visual bits for each category. If a feature is considered to better describe an image category, then its visual bit is '1', otherwise '0'. Each visual bit is a linear/RBF kernel classifier that maps the features to a binary bit for classification. These visual bits are augmented iteratively to refine visual words based on the learning performance of the classifier. The iterative process is carried out until a desired performance is achieved. Harris Laplace corner detectors (Mikolajczyk and Schmid, 2001) were used in detecting interest points and were described by SIFT descriptors. The authors compare their technique with the K-means based codebook of size 1000 followed by an SVM classifier that uses the χ^2 kernel, and with a codebook constructed using the extremely-random classification forest algorithm (Moosmann et al., 2007). Evaluations were performed on the PASCAL VOC Challenge 2006 image dataset that contains 10 classes of total 5304 images.

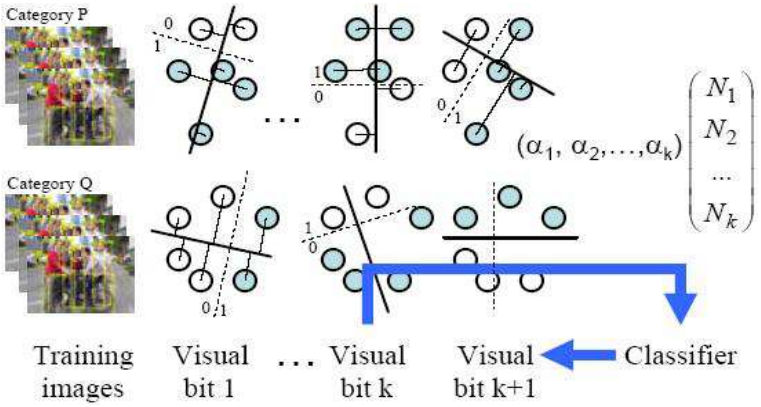


FIGURE 2.3: Image from Yang et al. (2008). Overview of the unified visual bit generation and classification process.

Zhang et al. (2009) proposed an iterative non-redundant codebook construction process by means of a weighted voting scheme of the AdaBoost procedure that is integrated with classifier learning. The authors applied this framework in visual object recognition and document classification domains with different experimental setups. However, the visual object recognition part is described for clarity. The following steps are iterated for a pre-defined number of iterations T:

1. a base codebook is learnt from a bag-of-features that are associated with a set of weights. The weights are initialised to be uniform over the training set.
2. training images are then mapped to fixed-length vectors using the *tf-idf* weight. A classifier is then learnt from the fixed-length feature vectors.
3. the predictions of the classifier in step 2 are used to update the weights using the AdaBoost procedure to the next iteration from step 1.

Different feature detectors: Hessian affine, the Kadir and Brady Salient regions, and the principal curvature-based region (PCBR) detector (Deng et al., 2007) described by SIFT descriptors are compared in their experiments. A base codebook is constructed using the K-means clustering algorithm with different weighted sampling techniques. A separate codebook for each detector is constructed with K=100 and then concatenated to form a global codebook. The number of boosting iterations T is set to 30. This straightaway increases the model's complexity by $T \times K$, making it difficult to cope with a large number of images and a large number of object categories. Evaluations are made on the Stonefly image dataset (Larios et al., 2007) containing 3826 images of nine different species. An ensemble of 50 unpruned C4.5 decision trees (Quinlan, 1993) was employed in each boosting iteration.

Wu and Rehg (2009a) recently showed that when the histogram intersection kernel (HIK) are used in clustering patch-based visual descriptors that are histograms, the codebooks constructed produce improved bag-of-features classifiers. The proposed method replaces K-means clustering that uses the L_2 distance metric with HIK for better performance when the choice of feature representation is histograms. When comparing K-means with K-median, the latter uses the L_1 distance metric. In the first step, features are extracted to construct a visual codebook of size 200. At the next step, an image or image sub-window is represented by a histogram of codewords in a specified image region. An image is represented by the concatenation of histograms from all 31 sub-windows that split an image into three levels, resulting in a histogram of dimension 6200. Spatial and edge informations are incorporated as an additional input, and histograms are concatenated from the original input and Sobel gradient image. The authors also propose a one-class SVM formulation using HIK that can be used to improve the effectiveness of the HIK-based codebook, by compact clusters in histogram feature space. The proposed methods are validated using three datasets: the dataset used in (Lazebnik et al., 2006)

containing 15 classes, a sports event dataset containing eight categories, and the Caltech-101 object recognition dataset. For the experiments performed with Caltech-101 image datasets, SIFT descriptors were used to describe the image patches and densely sampled features over grid. The census transform histogram (CENTRIST) descriptors proposed by the authors (Wu and Rehg, 2009b) was used with the other datasets. The original dimensionality of the CENTRIST descriptor is 256 which can be also reduced to 40 via PCA. One-versus-one SVM is used for classification with the histogram intersection kernel. The authors empirically show that the K-median codebook is a compromise between the HIK and K-means codebooks.

Martínez-Muñoz et al. (2009) very recently proposed a framework that is free from the use of a codebook for categorising objects in images. The dictionary-free categorisation is achieved by learning an initial random forest of trees, followed by the construction of a second-level ('stacking') training set, and learning through a stacked classifier. Bootstrap samples of images are drawn with replacements from the training set to create an initial random forest using a modified version of C4.5 (Quinlan, 1993). A histogram of the training examples belonging to each class is stored at each leaf of the decision tree. The purpose of the second-level training set is to consider the images that were not used to build the initial tree. For each image, its descriptors are dropped through each tree and their histograms are concatenated to obtain the feature vector for the stacking example. The authors extracted several features with the use of different combinations of detectors and descriptors. A random forest is associated with each and every combination of the detector and descriptor. Experiments were carried out with the Stonefly-9 (Larios et al., 2007) image dataset containing 3826 images of nine different species, and the PASCAL VOC Challenge 2006 image dataset containing 10 classes. For the PASCAL06 image set, interest points in each image were detected using Harris, Hessian and PCBR detector (Deng et al., 2007) and regularly sampled image patches. These interest points are then independently described by three different descriptors: SIFT, Colour SIFT (van de Sande et al., 2010), and the filter-bank descriptor employed by Winn et al. (2005). For the Stonefly-9 dataset, interest points were found using Hessian, Kadir and Brady salient region, and PCBR detectors; each of them was then described using SIFT descriptors. Edges were extracted using the Canny edge detector. The classifier is a boosted decision tree. Although they claim that the proposed method is simple and elegant, they were unable to grow any single tree on all the extracted descriptors as they drew a random sample of the descriptors. Despite this, they have to determine the minimum number of training examples in each leaf node, the minimum number of trees in each random forest, and the number of boosting iterations for the stacked classifier.

Summary

This chapter provides a review of the literature on the codebook model-based approach to visual object recognition. The approach, while ignoring any structural aspect in vision, nonetheless provides state-of-the-art performances on current datasets. This is impressive because we are simply modelling the statistical distributions of low-level image features. As in any review, the coverage here is not exhaustive. However, as our focus in this thesis is on the design of the codebook, we have attempted to provide an exhaustive coverage of the different codebook design strategies different authors have adopted. A summary of those several approaches that have been proposed over the last decade with their use of different feature detectors, descriptors, codebook construction schemes, choice of classifiers in recognising objects is depicted in Figure 2.4 and 2.5.

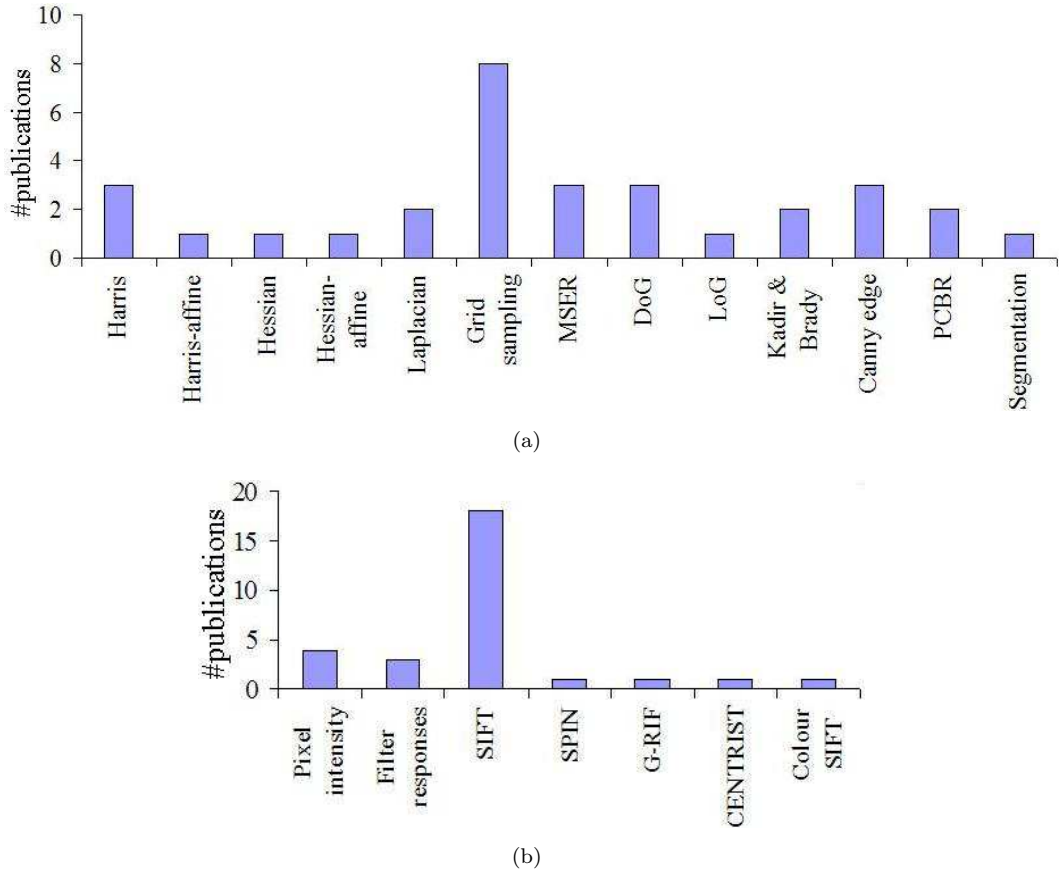


FIGURE 2.4: Number of publications reviewed in this chapter vs the usage of image patch (a) detectors and (b) descriptors for visual object recognition.

The vast majority of methods in the literature relating to the construction of codebooks are either K-means or Gaussian mixture models (GMMs), in which the obtained cluster centres are those that have high probability density. These codewords are not necessarily the most discriminative. GMM has better representative power than a single cluster. However, it requires more computational power.

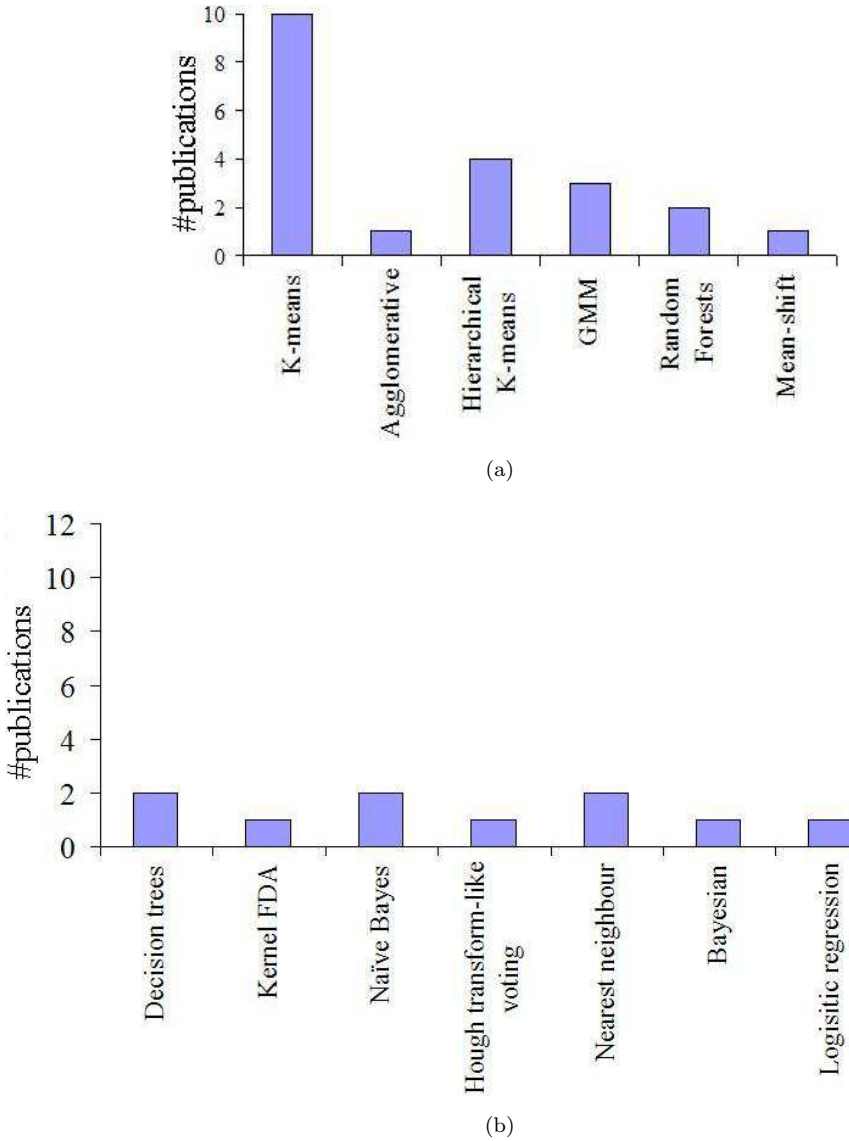


FIGURE 2.5: Number of publications reviewed in this chapter vs (a) different clustering techniques and (b) various classifiers in patch-based visual object recognition. It can be noticed that majority of the visual codebook construction involves K-means or its combination with hierarchical clustering. Another popular approach is the use of Gaussian Mixture Models (GMMs) in constructing a codebook. Both of these techniques constructs a codebook in such a way that the obtained cluster centres are those that have high probability density. In the classification step, the choice of SVMs are quite straightforward as they are naturally designed to perform classification in high dimensional spaces.

The size of the codebooks that have been used in the literature range from 10^2 to 10^4, resulting in very high-dimensional histograms. A larger size of codebook increases the computational needs in terms of memory usage, storage requirements, and the computational time to construct the codebook and to train a classifier. On the other hand, a smaller size of codebook lacks good representation of true distribution of features. Thus, the choice of the size of a codebook should be balanced between the recognition rate and computational needs.

Recent studies have started to explore the construction of visual codebook leading to an improved categorisation performance in terms of discriminative power, compactness, and inclusion of spatial information. Winn et al. (2005); Kim and Kweon (2007); Wang et al. (2008); Li et al. (2008); Yang et al. (2008) focused on both compactness and discriminative power of visual codebooks. Larlus and Jurie (2006); Lazebnik et al. (2006); Moosmann et al. (2007) focused on incorporating spatial information in the codebook model, and Grauman and Darrell (2005); Nister and Stewenius (2006); Agarwal and Triggs (2006); Wang (2007); Zhang et al. (2009) focused on constructing multi-resolution codebooks.

It is worth noting that in their work, Sivic and Zisserman (2003) used $200000\times\mathbb{R}^{128}$ features which represent about 10% of the original dataset that was clustered into 10000 and 6000 clusters for each type of detector used in constructing a codebook. Winn et al. (2005) used a subset of the PASCAL VOC Challenge 2005 dataset (587 images) to construct an initial codebook of size 1200. Furthermore, Moosmann et al. (2007) in their experiments with the PASCAL VOC Challenge 2005 dataset, used 50000 patches in total over the 648 images (73 patches per image) to construct a codebook of size 30000. These examples show the major bottleneck occurs in handling the massive scale of the datasets and the patch-based descriptors in constructing a visual codebook.

While several approaches were explored, there has been very little attempt at a large scale clustering of patch-based descriptors. The methods that we reviewed in this chapter are mostly applied to modest size problems. As the size of training sets increases, the size of codebook and complexity of construction will increase. That is why a clever algorithm is needed. Therefore, we are demonstrating a resource-allocating codebook approach in this thesis which is efficient due to fast clustering and which is capable of dealing with large high-dimensional features.

Chapter 3

Patch-based visual descriptors

In most pattern recognition applications it is necessary to transform the data into some new representation before training the classifiers. The simplest case is the linear transformation of the input data and also possibly of the output data. The transformation on the input data is termed as *pre-processing* and sometimes the transformation on the output data is termed as *post-processing*. The complex case may involve dimensionality reduction on the input data that can lead to improved performance. The combinations of inputs are normally referred to as *features*. The process of generating features from the image sets is called as *feature extraction*. The feature extraction process is in connection with the structure chosen for the input data representation, and strongly depends on the implemented applications (Nixon and Aguado, 2008).

Feature extraction is the first step in a processing chain followed by codebook construction, computing the frequency histograms and object classification as seen in Figure 1.2. The feature extraction process in visual object recognition systems generally seeks for invariance properties that do not vary according to different conditions such as scale, rotation, affine and illumination changes. Further, the dimension of a descriptor has a direct impact on the computational time and a lower number of dimensions are therefore desirable. Depending on the image content, the number of interest points found with a given detector may vary for different image categories. Therefore, efficiency is one of the major properties when selecting a feature detector, especially for online applications where large amounts of data need to be processed. Usually images are composed of different sets of colours, a mosaic of different texture regions, and different local features. Most previous studies have focused on using global visual features such as edge orientation, colour histogram and frequency distribution. Recent studies use local features that are more robust to occlusions and spatial variations. In visual object recognition, image features such as colour, texture, and shape are important to describe semantically image contents that can be used by humans to categorise objects in scenes. These image features can be correlated with the high-level image semantics (Chen et al., 2005).

However, the introduction of powerful patch-based Scale-Invariant Feature Transform (SIFT) descriptors proposed by Lowe (1999) had a significant impact on the popularity of local features. Interest points combined with local descriptors started to be used as a black box providing reliable and repeatable measurements from images for a wide range of applications such as object recognition, texture recognition, robot navigation and visual data mining. These local descriptors computed on the interest points can capture the essence of a scene without the need for semantic-level segmentation. This new way of looking at local features has opened up a whole new range of applications and has brought us a step closer to cognitive level image understanding. Even though many different methods for detecting and describing local image regions have been developed, in this chapter we explore the well known patch-based SIFT descriptor and its follow up technique, the Speeded-Up Robust Features (SURF) (Bay et al., 2008) descriptors that we used in our evaluation.

The Harris corner detector (Harris and Stephens, 1988) is based on the eigen values of the second order matrix. Harris corners are not scale-invariant. Mikolajczyk and Schmid (2001) proposed robust and scale-invariant feature detectors with high repeatability that they refer to as Harris-Laplace and Hessian-Laplace. They used the Harris measure or the determinant of the Hessian matrix to select the location and the Laplacian to select the scale. Lowe (1999) approximated the Laplacian of Gaussian (LoG) by a difference of Gaussians (DoG) filter, whereas Bay et al. (2008) achieved by relying on integral of images for image convolutions. A survey on invariant detectors, descriptors and implementation details can be found in (Mikolajczyk and Schmid, 2005; Tuytelaars and Mikolajczyk, 2008).

The SIFT and SURF features detected on the Lena's image are illustrated in Figure 3.1 (b) and (c), respectively. Here the features detected are shown by centres of the circles, where the radius reflects magnitude and the direction reflects the orientation of the feature. The majority of the features are detected in the face, rim of the hat and mirror, and in other textured regions of the image. The SIFT detected $473 \times \mathbb{R}^{128}$ interest points

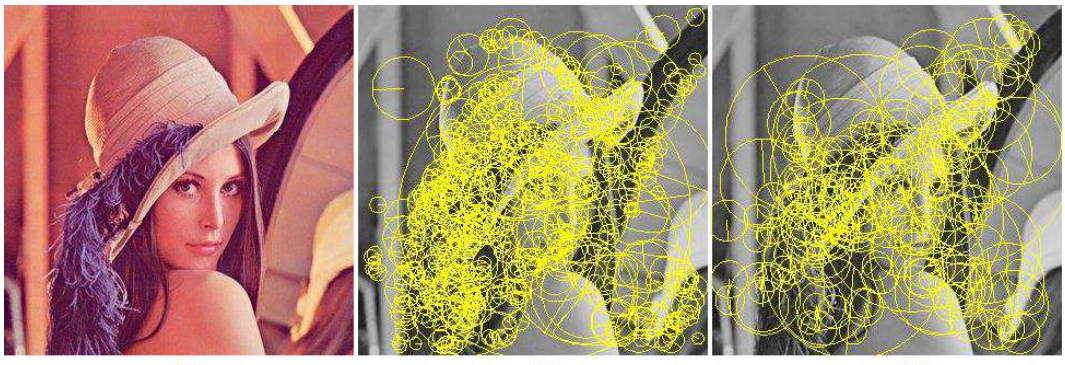


FIGURE 3.1: (a) Original Image (b) SIFT keypoints with magnitude and direction (c) SURF keypoints with magnitude and direction

while SURF detected $176 \times \mathbb{R}^{64}$ keypoints, resulting in 57 keypoints as exact overlap. The time taken to calculate the SIFT keypoints was 0.60 seconds whereas SURF keypoints was calculated in 0.06 seconds on a desktop computer with an Intel Core 2 running at 2.4GHz and 4GB of RAM.

3.1 Scale-invariant feature transform (SIFT)

The SIFT is a method to extract distinctive features from gray-value images, by filtering images at multiple scales and patches of interest that have sharp changes in local image intensities. It can be used in the context of matching and recognition of the same scene or object observed under different viewing conditions: scale and rotation. The SIFT operator was initially presented in a paper by Lowe (1999) and developed further in 2004 (Lowe, 2004). The SIFT algorithm consists of four major stages: Scale-space extrema detection, keypoint localisation, orientation assignment, and representation of a keypoint descriptor.

Ke and Sukthankar (2004) improved upon SIFT by replacing the smoothed weighted histograms with principal components analysis (PCA) at the final stage of the SIFT (Ke and Sukthankar, 2004). The authors refer to their method as PCA-SIFT. In this method, the dimensionality of the feature space was reduced from 128 to 20 which requires less storage and increased speed in matching images. Following the first three stages of the standard SIFT algorithm, the PCA-SIFT extracts an image patch centred over a keypoint with size 41×41 pixels at the given scale, rotated to its dominant orientation. Following this step, the horizontal and vertical gradients are computed, resulting in a vector of size $2 \times 39 \times 39 = 3042$ elements per patch. Then the covariance matrix of $k \times 3042$ is calculated to express the gradient images of local patches, where k is the number of keypoints detected. The first n eigen vectors are selected to form a projection matrix of size $n \times 3042$. Finally, given a patch, its local image gradient vector is projected using the eigen space to derive a compact feature vector which is significantly smaller than the standard SIFT feature vector. PCA-SIFT is less distinctive than SIFT (Mikolajczyk and Schmid, 2005).

An overview of the four major stages in the SIFT technique is presented in the subsections as discussed by Lowe (2004).

3.1.1 Scale-space extrema detection

The following cascade filtering approach can be used to identify the candidate locations of the keypoints in the image scale space:

FIGURE 3.2: Local scale-invariant points: (Left) The 640x480 pixel original image of a bicycle, (Right) The 2773 keypoints that are detected in the image are marked by '+' in yellow

1. An input image I is convolved with a Gaussian function G with variance σ to give an image A.

2. I is convolved again with the G using $k\sigma$ to give another image B, where k is a constant factor in scale space.

3. The difference of Gaussian (DoG) is obtained by: DoG $\leftarrow$ B-A

4. Repeat steps 1 to 3 such that, the initial image I is incrementally convolved with G to produce images separated by k (the stacked images are shown in the left in Figure 3.4). The image I becomes increasingly more blurred as the higher spatial frequencies (i.e., noise) are filtered out by increasing the σ.

The DoG images produce responses which approximate the LoG that avoids the computation of second order derivatives in x and y directions. Based on the diffusion equation in scale-space theory (Witkin, 1983), it can be shown that the Laplacian corresponds

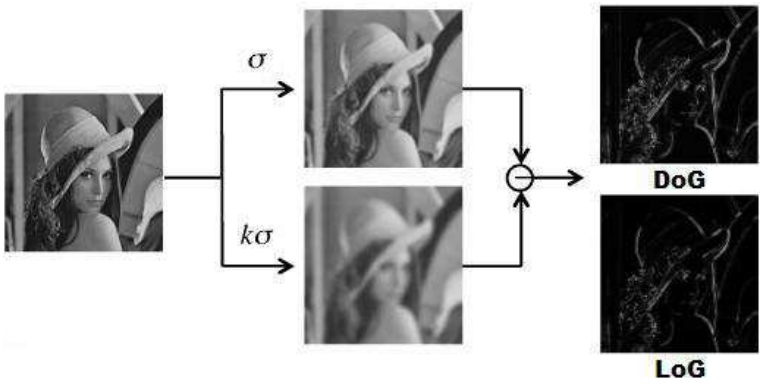


FIGURE 3.3: A difference of Gaussian (DoG) filters at different scales (shown in the middle) applied on the same image (shown in the left) approximates the Laplacian of Gaussian (LoG)

to the derivative of the image in the scale direction. The difference between images at different scales approximates the derivative with respect to scale (see Figure 3.3).

Once a complete octave has been processed, the next octave is obtained by down-sampling the image that has twice the initial value of σ by taking every second pixel in each row and column (see Figure 3.5). Each octave of scale space is divided into s intervals such that $k = 2^{1/s}$ produces s+3 images in the stack of blurred images for each octave. The value of k covers a complete octave for the final extrema detection.

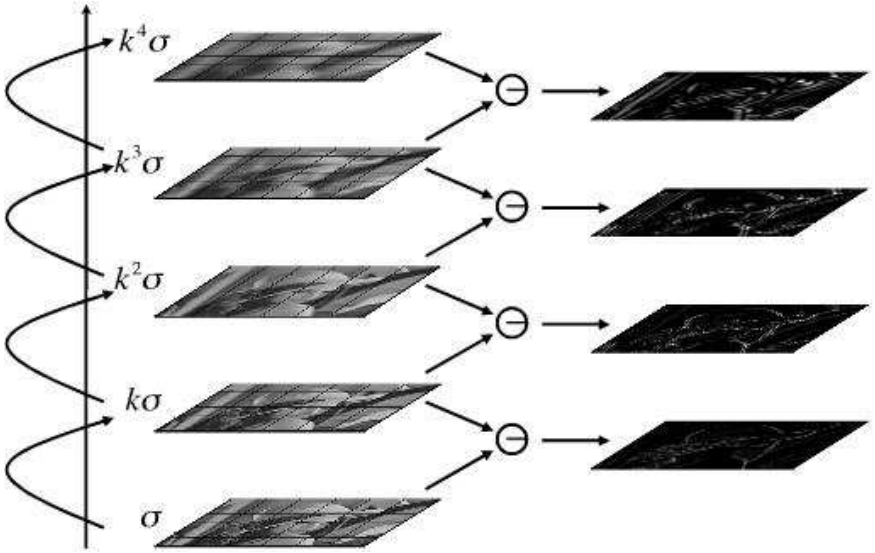


FIGURE 3.4: Scale space images and difference of Gaussian (DoG) images

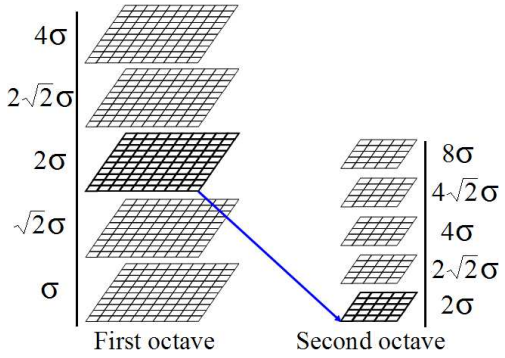


FIGURE 3.5: Two octaves of a Gaussian scale-space image pyramid with s=2 intervals. Each octave is obtained by down sampling the image that has twice the initial value of σ by taking every second pixel in each row and column.

The scale space of an image is defined as a function $L(x, y, \sigma)$ as follows:

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y)$$

where $*$ is the convolution operation in x and y.

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2}$$

$$D(x, y, \sigma) = (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y)$$
$$= L(x, y, k\sigma) - L(x, y, \sigma)$$

The local maxima and minima of $D(x, y, \sigma)$ are computed by comparing each sample point in the current image with its eight neighbours in the same scale space and nine neighbours in the scale above and below (see Figure 3.6). This point will be selected only if it is the maximum or minimum of all of them.

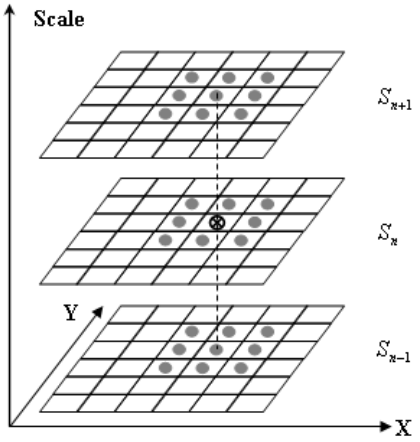


FIGURE 3.6: Maxima/minima is defined as any value in the DoG greater than all its neighbours in scale-space. That is, each pixel is compared to eight neighbours in current image (S_n), nine neighbours in scale above (S_{n+1}), nine neighbours in scale below (S_{n-1}) and a pixel (i.e. a keypoint) is taken if larger/smaller than all of them.

3.1.2 Keypoint localisation

This stage eliminates those keypoints that are unstable that is, keypoints are eliminated by finding those that have low contrast or are poorly localised on an edge. In order to determine the interpolated locations of the keypoints, the Taylor expansion up to the quadratic terms of the $D(x, y, \sigma)$ is shifted so that the origin is at the sample point.

$$D(\mathbf{x}) = D + \left(\frac{\partial D}{\partial \mathbf{x}}\right)^T \mathbf{x} + \frac{1}{2}\mathbf{x}^T \left(\frac{\partial^2 D}{\partial \mathbf{x}^2}\right) \mathbf{x} \tag{3.1}$$

where D and its derivatives are evaluated at the sample point and $\mathbf{x}$ is the offset from this point. The location of the extremum $\hat{\mathbf{x}}$, is found by taking the derivative of equation 3.1 with respect to $\mathbf{x}$ and setting it to zero:

$$\hat{\mathbf{x}} = -\left(\frac{\partial^2 D}{\partial \mathbf{x}^2}\right)^{-1} \left(\frac{\partial D}{\partial \mathbf{x}}\right) \tag{3.2}$$

The function value at this extremum, $D(\hat{\mathbf{x}})$, is useful to reject those points with low contrast. This value can be evaluated by substituting equation 3.2 into 3.1.

$$D(\hat{\mathbf{x}}) = D + \frac{1}{2}\left(\frac{\partial D}{\partial \mathbf{x}}\right)^T \hat{\mathbf{x}} \tag{3.3}$$

Extrema with a value of $|D(\hat{\mathbf{x}})| < 0.03$ were used to filter out low contrast keypoints. To eliminate extrema based on poor localisation it is noted that in these cases there is a large principle curvature across the edge but a small curvature in the perpendicular direction in the difference of Gaussian function. To reject the keypoints that are situated along edges, the ratio of the principal curvature at the sample point can be found by calculating the Hessian matrix $\mathbf{H}$ and rejecting those ratios that are too large (Lowe, 2004).

$$\mathbf{H} = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{bmatrix}$$

The trace of $\mathbf{H}$, $\mathrm{Tr}(\mathbf{H})$, and determinant, $\mathrm{Det}(\mathbf{H})$, can be computed as follows:

$$Tr(\mathbf{H}) = D_{xx} + D_{yy} = \lambda_1 + \lambda_2$$
$$Det(\mathbf{H}) = D_{xx}D_{yy} - (D_{xy})^2 = \lambda_1\lambda_2$$

Let r be the ratio between the largest eigen value λ_1 and the smallest eigen value λ_2, so that $\lambda_1 = r\lambda_2$. Then,

$$\frac{Tr(\mathbf{H})^2}{Det(\mathbf{H})} = \frac{(\lambda_1 + \lambda_2)^2}{\lambda_1\lambda_2} = \frac{(r+1)^2}{r}$$

Therefore, to check that the ratio of principal curvatures is below some threshold, r, it is only needed to check the following inequality:

$$\frac{Tr(\mathbf{H})^2}{Det(\mathbf{H})} < \frac{(r+1)^2}{r}$$

According to (Lowe, 2004), keypoints that have a ratio between the principal curvatures greater than 10 are eliminated.

3.1.3 Orientation assignment

After the step of keypoint localisation, each of the stable keypoints detected is assigned with orientations so that a keypoint is rotationally invariant. This can be achieved by computing the gradient magnitude $m(x, y)$ and orientation $\theta(x, y)$ at a sample point (x, y) in a region around the keypoint location. The computational steps are as follows:

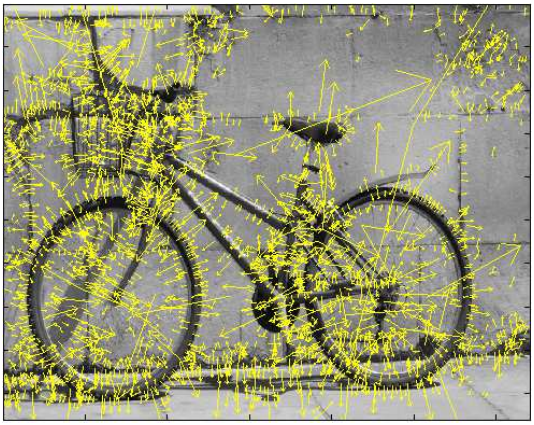


FIGURE 3.7: The 2773 keypoints are displayed as vectors indicating location, scale and orientation of Figure 3.2.

$$m(x,y) = \sqrt{(L(x+1,y) - L(x-1,y))^2 + (L(x,y+1) - L(x,y-1))^2}$$

$$\theta(x,y) = \tan^{-1}\left(\frac{L(x,y+1) - L(x,y-1)}{L(x+1,y) - L(x-1,y)}\right)$$

3.1.4 Representation of a keypoint descriptor

The next stage is to create descriptors to represent the selected keypoints in an invariant form. A keypoint descriptor is a 3D histogram of location (x, y), gradient $m(x, y)$, and orientation $\theta(x, y)$, where each location is covered by a 16×16 sample array which is then summarised into a 4×4 subregions and the gradient angle is quantized into eight orientations. Histograms consist of eight bins, one for each 45° step (see Figure 3.8). The magnitude of each of the sample points is weighted by a Gaussian window with σ equal to 1.5 times the keypoint's scale. The Gaussian window is used to evade sudden changes in the descriptor with small changes in the position of the window, and to give less importance to gradients that are far away from the centre of the descriptor. The orientation is in the range [-π, π] radians. The resulting descriptors are of dimension N×128, where N is the number of keypoints that are detected in image I.

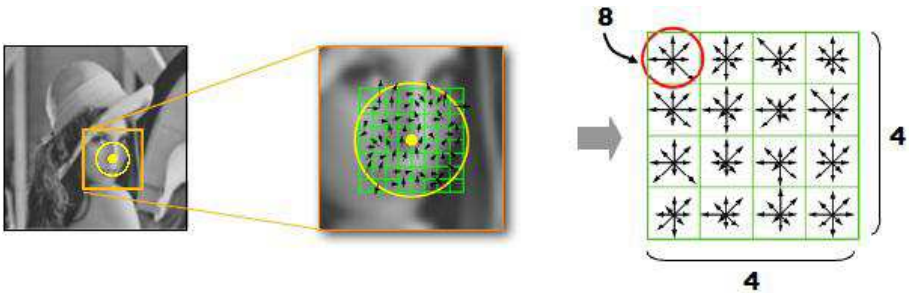


FIGURE 3.8: (Left) An interest keypoint detected in the Lena's image (Middle) Keypoint descriptors are weighted by a Gaussian window, indicated by a yellow circle. (Right) A 2x2 descriptor array computed from an 8x8 set of samples.

3.2 Speeded-up robust features (SURF)

SURF is a scale and rotation-invariant descriptor presented by Bay et al. (2008). SURF is partly inspired by SIFT that makes use of integral images (see Figure 3.9) to efficiently compute a rough approximation of the Hessian matrix. The Hessian matrix (defined in section 3.1.2) is roughly approximated using a set of box-type filters (see Figure 3.11) and no smoothing is applied when going from one scale to the next. Image convolutions with these box filters can be computed rapidly by using integral images independently of their size. The use of integral images drastically reduces the computation time.

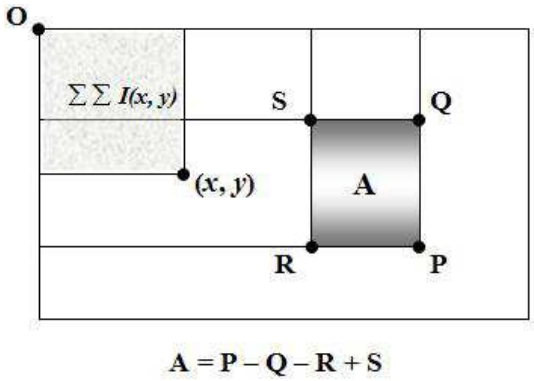


FIGURE 3.9: The integral image at a location $\mathbf{x} = (x,y)$ represents the sum of all pixels in the input image I of a rectangular region formed by the origin and the point $\mathbf{x}$. The area of **A** takes only 4 operations using the integral images.

The entry of an integral image (summed area tables) $I_{\Sigma}(x)$ is an intermediate representation for the image at location (x, y) and contains the sum of gray-scale pixel values in the image I of a rectangular region formed by the origin and (x, y).

3.2.1 Scale-space detection

Interest points need to be found at different scales, where scale spaces are usually implemented as an image pyramid. The SIFT descriptor iteratively reduces the image size, and uses a DoG and Hessian detector by subtracting these pyramid layers. Instead, in SURF, the scale space is rather analysed by up-scaling (see Figure 3.10) the integral image-based filter sizes in combination with a fast Hessian matrix-based approach. In SURF, as there is no down sampling of the image, hence there are no aliasing effects. The detection of interest points is selected by relying on the determinant of the Hessian matrix where the determinant is maximum.

Given a point (x, y) in an image I, the Hessian matrix $H(\mathbf{x}, \sigma)$ in $\mathbf{x}$ at scale σ defined as follows:

$$\mathbf{H} = \begin{bmatrix} L_{xx}(\mathbf{x}, \sigma) & L_{xy}(\mathbf{x}, \sigma) \\ L_{xy}(\mathbf{x}, \sigma) & L_{yy}(\mathbf{x}, \sigma) \end{bmatrix}$$

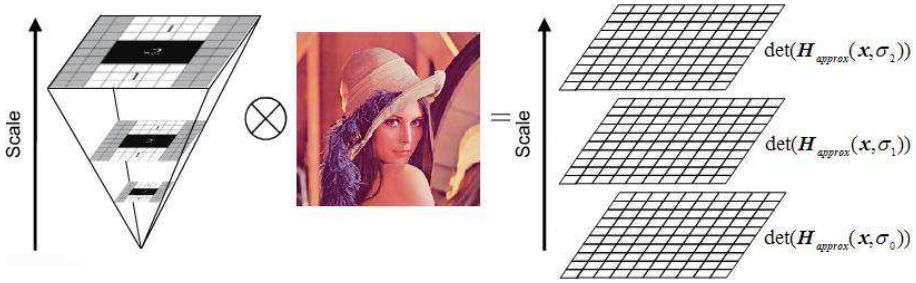


FIGURE 3.10: An octave represents a series of up-scaling filter response maps obtained by convolving the same image with a filter of increasing size.

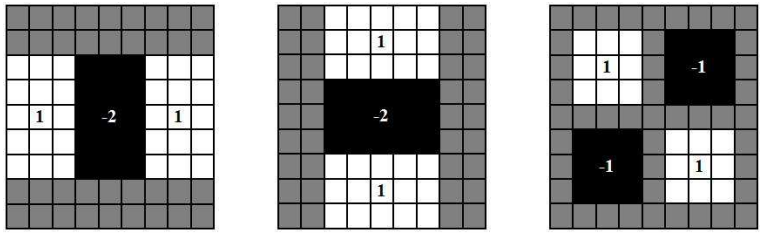


FIGURE 3.11: SURF's box filter approximation for the second-order Gaussian partial derivative in x-direction (D_{xx}), y-direction (D_{yy}) and in xy-direction (D_{xy}). The gray regions are equal to zero.

where $L_{xx}(\mathbf{x}, \sigma)$ is the convolution of the Gaussian second order derivative with the image I in point $\mathbf{x}$ and similarly for $L_{xy}(\mathbf{x}, \sigma)$ and $L_{yy}(\mathbf{x}, \sigma)$.

The approximation for the Hessian matrix is achieved with box filters. These box filters approximate to second-order Gaussian derivatives and can be evaluated using integral images. The 9×9 box filters in Figure 3.11 are used to replace a Gaussian with σ=1.2. A weight factor w indicated in equation 3.4 is used to balance the expression for the Hessian determinant.

$$det\left(H_{approx}\right) = D_{xx}D_{yy} - \left(wD_{xy}\right)^2 \tag{3.4}$$

$$w = \frac{\left|L_{xy}\left(1.2\right)\right|_F \left|D_{yy}\left(9\right)\right|_F}{\left|L_{yy}\left(1.2\right)\right|_F \left|D_{xy}\left(9\right)\right|_F} \approx 0.9 \tag{3.5}$$

where $|.|_F$ is the Frobenius norm (i.e. $|A|_F = \sqrt{Tr\left(AA^H\right)}$, where A^H is the conjugate transpose). The value computed for w using equation 3.5, depends on the scale factor. Furthermore, the filter responses are normalised with respect to their size.

3.2.2 Keypoint localisation

Interest points are localised in scale and image space by applying a non-maximum suppression in a 3×3×3 neighbourhood. Finally, the found maxima of the determinant of the approximated Hessian matrix are interpolated in scale and image space.

3.2.3 Orientation assignment

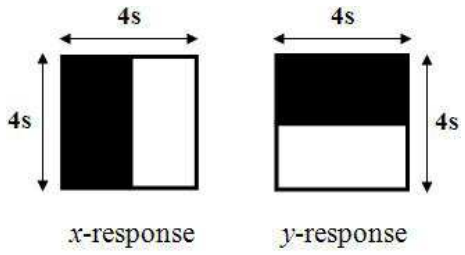


FIGURE 3.12: The Haar wavelet filter used in x-response and y-response with SURF descriptors.

The orientation is computed using 2D Haar wavelet responses in both x and y direction as shown in the Figure 3.12, calculated in a 4×4 sub region around each interest point. A sliding orientation window of size $\frac{\pi}{3}$ is used to detect the dominant orientation of the Gaussian weighted Haar wavelet responses within a circular neighbourhood of radius $6s$ around the interest point, where s is the scale at which the point was detected. Then the dominant orientation is determined by the summed responses of x (D_x) and y (D_y) that yields the longest vector.

3.2.4 Representation of a keypoint descriptor

The steps involved in representing a keypoint descriptor are summarised in the following:

1. The interest region is then split up into 4×4 square sub regions with 5×5 regularly spaced sample points inside.
2. Then the Haar wavelet responses D_x and D_y are calculated and the responses are weighted with a Gaussian kernel centred at the interest point.
3. The responses over each sub-region for D_x and D_y are computed separately. This results in a descriptor vector of length 32. In order to bring in information about the polarity of the intensity changes, the sum of absolute value of the responses is extracted. This in turn results in a descriptor vector of length 64.
4. The vector is then normalised into unit length in order to achieve invariance to contrast.

The standard SURF descriptor has a dimension of 64 and the extended version (e-SURF) has a dimension of 128.

3.3 Summary

The SIFT detects interest points by filtering images at multiple scales that have sharp changes in local image intensities. The features are located at maxima and minima of a difference of Gaussian (DoG) functions applied in scale space. Next, the descriptors are computed based on eight orientation histograms at a 4×4 sub region around the interest point, resulting in a 128 dimensional vector.

The SURF is partly inspired by SIFT that makes use of integral images. The scale space is analysed by up-scaling the integral image-based filter sizes in combination with a fast Hessian matrix-based approach. The detection of interest points is selected by relying on the determinant of the Hessian matrix where the determinant is maximum. Next, the descriptors are computed based on orientation using 2D Haar wavelet responses calculated in a 4×4 sub region around each interest point, resulting in a 32 dimensional vector. When information about the polarity of the intensity changes is considered, this in turn results in a 64 dimensional vector. The extended version of SURF has the same dimension as SIFT.

SIFT and SURF descriptors are invariant to common image transformations, such as scale changes, image rotation, and small changes in illumination. These descriptors are also invariant to translations as from the use of local features. SURF features can be extracted faster than SIFT using the gain of integral images and yield a lower dimensional feature descriptor resulting in faster matching and less storage space. Furthermore, SURF shows slightly better performance to SIFT in illumination changes, but performs less when the rotation is large (Juan and Gwun, 2009). The SIFT descriptors have been found highly distinctive in performance evaluation, but SURF has been reported three to five times faster than SIFT (Mikolajczyk and Schmid, 2005). In contrast, SIFT, when compared to PCA-SIFT and SURF, has better performance but it is slow and performs poorly at illumination changes (Juan and Gwun, 2009).

Several authors, e.g. Zhang et al. (2007); Tuytelaars and Mikolajczyk (2008) suggest that different detectors should be used complementarily, as they have different properties (some detect edge-like, some corner-like structures), so more information can be captured. The use of different detectors and/or descriptors results in large scale features in a higher dimensional space. A way to cope with the diversity and size of the feature sets is to cluster them, i.e. instead of the features themselves, to use only the cluster means or some other representations. Depending on the clustering algorithm, we might obtain different clustering solutions, some of which might be more suitable than others for object class recognition. We discuss these in the next chapter.

Chapter 4

Visual vocabulary design

A simple nearest neighbour design for patch-based visual object recognition is a possible way forward, but is computationally not feasible for large scale data. Hence, a way to cope with the enormous amount of the patch-based descriptors and their higher dimensionality is to cluster them by using an appropriate clustering method that captures the span of the feature space. Instead of the features themselves, the cluster centroids or representative points are used for the different cluster members. In this chapter firstly we describe the well known and widely used codebook model in visual object recognition. Secondly, some standard clustering techniques are reviewed which are fundamental to this thesis and which will appear multiple times in this chapter. In the following sections, we then demonstrate the two novel approaches, that we call RAC and SHC techniques. Finally, we present an approach which, instead of clustering, adaptively constructs a codebook by computing Fisher scores between the classes of interest.

4.1 Codebook model

In state-of-the-art patch-based visual object recognition systems, the visual codebook model has shown excellent categorisation performance in large evaluations (e.g. the PASCAL VOC Challenges). Desirable properties of a visual codebook are compactness, low computational complexity, and high accuracy of subsequent categorisation. Discriminative power of a visual codebook determines the quality of the codebook model, whereas the size of a codebook controls the complexity of the model. Thus, the construction of a codebook plays a central role that affects the model complexity. In general, there are two types of codebook that are widely used in the literature: global and category-specific (or object class-wise) codebook. A global codebook may not be sufficient in its discriminative power but it is category-independent, whereas a category-specific codebook may be too sensitive to noise.

However, another type of codebook is the semantic codebook approach that is widely used in image scene categorisation (van Gemert et al., 2006). The construction of a semantic codebook is achieved by manually annotating image patches that yield meaningful codewords making the codebook more compact and discriminative. The underlying phenomenon of selecting meaningful codewords is that the local image semantics will propagate to the global codebook image model. However, not all images can be decomposed into semantic codewords. For example, an indoor scene, say a *house*, is unlikely to contain *sea*, *sky*, *rock*, *sand*, and *mountain*. Moreover, manually annotating local patches in large evaluations, especially when there are multiple object categories present in most of the images, becomes a time consuming process. In this chapter we consider only the global and category-specific codebooks as we are more focused on large evaluations.

4.1.1 Bag-of-features

The bag-of-words (BOW) approach was originally used in text mining (Joachims, 1998) and is now widely used in image scene classification (Fei-Fei and Perona, 2005; Quelhas et al., 2007), retrieval of objects from a movie (Sivic and Zisserman, 2003), and object classification (Csurka et al., 2004; Winn et al., 2005; Zhang et al., 2009) tasks in computer vision. The bag-of-words in computer vision is normally referred to as 'bag-of-features' or 'bag-of-keypoints'. The pseudocode of bag-of-features approach is given in Algorithm 1.

Algorithm 1 Proccess of building a bag-of-feature (BOF) representation for images

```
for all image do
    interestPts ← detectPts(image)
    descriptors ← describePts(interestPts)
end for
codebook ← quantizePts(descriptors(training_images))
for all image do
    BOF ← computeHistogram(codebook, descriptors(image))
end for
```

Interest points or regions are detected in training images and a visual codebook is constructed by a vector quantization technique that groups similar features together. Each group is represented by the learnt cluster centres referred to as 'visual words' or 'codewords'. The size of the codebook are the number of clusters obtained from the clustering technique. Each interest keypoint of an image in the dataset is then quantized to its closest codeword in the codebook, such that it maps the entire patches of an image in to a fixed-length feature vector of frequency histograms, i.e. the visual codebook model treats an image as a distribution of local features.

FIGURE 4.1: A schematic example of an object and it's possible codewords. (a) image of an "object" (e.g. dog) category (b) possible codewords that makes up a visual codebook

Figure 4.1 shows an image of an object category and the corresponding image patches that are used as the codewords. Figure 4.2 illustrates some example images that are of four object categories ('dog', 'aeroplane', 'motorbike', and 'car') represented by the bag-of-features approach.

The aforementioned histogramming process can be mathematically expressed as follows. For each codeword c in the visual codebook $\mathbf{C}$, the traditional codebook model constructs the distribution of codewords over an image by

$$H(c) = \sum_{c \in IR} \begin{cases} 1 & ; \text{ if } c = \underset{c \in \mathbf{C}}{\text{argmin}}\, S(c, r) \\ 0 & ; \text{ otherwise} \end{cases} \tag{4.1}$$

where, IR denotes the set of regions or patches in an image I and $S(c, r)$ denotes the similarity (e.g. Euclidean distance) between a codeword c and a region r. The mathematical expression in 4.1 of assigning a single codeword to a single image feature is referred to as hard-assignment. Instead of hard-assignment, each region r, can be assigned to all codewords in a probabilistic manner, i.e. assign weights w_c to neighbouring codewords. Hard-assignment becomes soft-assignment when equation 4.1 is replaced by

$$H(c) = \sum_{c \in IR} S(c, r) \times w_c \tag{4.2}$$

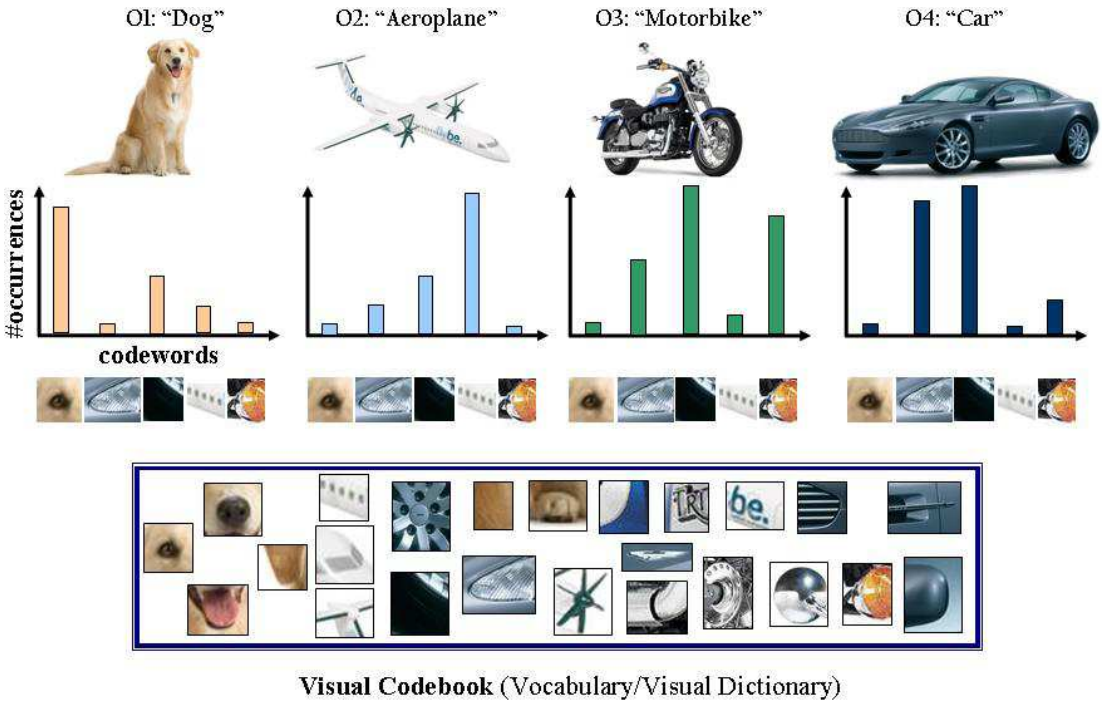


FIGURE 4.2: Image representation using bag-of-features approach. An image is represented as a bag-of-features where each local feature (local patch) is represented by the best fitting codeword in the visual codebook. The distribution of the codeword-counts (histograms) yields the image model.

The traditional codebook approach makes use of the hard-assignment method. A soft-assigned method combines the spatial verification, in which each interest point in an image has more assigned codewords and can potentially match more features in the other image.

It is important to point out the distinction between clustering the visual descriptors sampled densely by using the K-means clustering method to that of carving the feature space in a way that captures the rare and informative visual keypoints from a set of training images. The dense clustering methods will work well in homogeneous images, such as texture analysis, but the visual descriptors of images in real world object recognition tasks are distributed very nonuniformly in feature space (Jurie and Triggs, 2005). Furthermore, clustering millions of data vectors of 128 dimensions into thousands of cluster centres using the K-means technique is not straightforward to apply and need for novel approaches arises. Therefore, in this chapter we demonstrate sequential methods to construct a visual codebook in a very fast way by *drastically reducing* the computational needs while comparably or slightly outperforming more traditional approaches in visual object recognition.

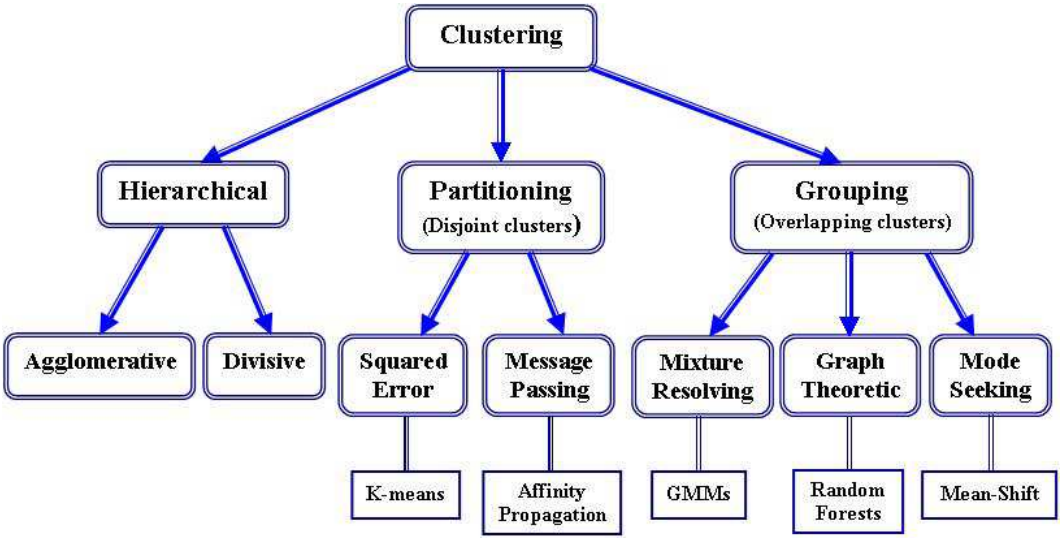


FIGURE 4.3: A taxonomy of clustering techniques.

4.2 Clustering techniques

Clustering as a tool in pattern recognition has a wide spectrum of applications: text classification (Zhao et al., 2005), mining in large data warehouse environments (Achtert et al., 2005), dynamic routing in optical networks (Hasan and Jue, 2008), and codebook construction for bag-of-features in visual scene analysis problems (Zhang et al., 2007) are examples of this. Generally, clustering has the property that all points closer to all others in their own cluster than to any points in any other cluster. Figure 4.3 depicts a taxonomy of various clustering techniques.

Once clusters are obtained by a method, the next step is to evaluate the resultant clusters for their quality. There are several ways of evaluating the usefulness of a clustering technique, such as cluster artificial datasets (e.g. generated by a mixture of Gaussians) and evaluation of classification error.

The evaluation of the usefulness of a clustering technique needs to consider the usefulness of the retained clusters for a given purpose or application rather than an application-independent mathematical problem. An example would be using clustering as a preprocessing step for visual object recognition. The final goal of this example can be quantified in terms of error rate in the classification or detection task. Here it does not help at

Task	**Goal**	**Evaluation**
Preprocessing	prediction	improved prediction performance
Compression	reduce size	compression rate
Exploratory data analysis	scientific discovery	statistical tests
Define categories	taxonomy	stability across data representation

TABLE 4.1: Applications and the evaluation procedure of clustering techniques

all to compute scores like within-cluster similarity or stability. A list of problems and their evaluations of the usefulness of a clustering technique are listed in Table 4.1(Guyon et al., 2009). We will now explore various clustering techniques that we used to compare and contrast our methods.

4.2.1 *K*-means

Given a matrix $\mathbf{X} \in \mathbb{R}^{N \times d}$ (representing N points - rows - described with respect to d features - columns), then K-means clustering aims to partition the N points into K disjoint sets or clusters by minimizing an objective function, which is the squared error function, that minimizes the within-group sum of squared errors:

$$dist_{ij} = \parallel X_i^{(j)} - C_j \parallel^2$$

$$X_{opt} = \sum_{j=1}^{K} \sum_{i=1}^{N} dist_{ij}$$

where $dist_{ij}$ is a chosen distance measure between a data point $X_i^{(j)}$ and the cluster centre C_j, is an indicator of the distance of the N data points from their respective cluster centres. K-means is a Gaussian mixture model with isotropic covariance matrix the algorithm is expectation-maximization (EM) algorithm for maximum likelihood estimation.

There are several known difficulties with the use of K-means clustering, including the choice of a suitable value for K, the computational cost of clustering when the dataset is large. It is also significantly sensitive to the initial randomly selected cluster centres. The K-means algorithm can be run multiple times to reduce this effect, but that makes it computationally more expensive and might take several weeks or even months to cluster millions of data! According to our experience, these are major issues affecting the performance of visual object recognition systems.

Clustering the visual keypoints using K-means, forces the relationship between different codes to be assigned to one of the fixed clusters K. Any interest point lying far away from any of the centres in the feature space, can significantly distort the position of the centre that they are assigned to. K-means is unable to handle noisy data and outliers. Although it can be proved that the iterative procedure will always terminate, the algorithm does not necessarily find the most optimal solution, corresponding to the global objective function that minimises the squared error within clusters (Jain et al., 1999; Leibe et al., 2008).

The time complexity of the traditional K-means method is O($NdKm$), where the symbols in parentheses represent number of data, dimensionality of features, the number of desired clusters and the number of iterations of the EM algorithm.

4.2.2 Mean-shift

The mean-shift clustering was initially proposed by Fukunaga and Hostler (1975), later adapted by Cheng (1995) for image analysis and thereafter extended by Comaniciu and Meer (2002) for low-level vision tasks such as image smoothing and image segmentation. More recently, Jurie and Triggs (2005) adapted the mean-shift approach in (Comaniciu and Meer, 2002) and the online clustering method in (Meyerson et al., 2004) to construct codebooks for visual object recognition.

Now, we briefly explain the underlying mathematical theory of the mean-shift based clustering approach as discussed by Comaniciu and Meer (2002). A feature space is regarded as a probability density function and the dense regions in this feature space correspond to local maxima (i.e. the modes) of the probability density function.

For notation, we closely follow $\mathbf{X}$ with N items of d-dimensional vectors to describe the modes detecting procedure in the underlying density $f(\mathbf{x})$. The modes are located among the zeros of the gradient $\bigtriangledown f(\mathbf{x}) = 0$. The mean-shift procedure locates these zeros without estimating the density (Comaniciu and Meer, 2002).

The multivariate kernel density estimate using a radially symmetric kernel (e.g. Gaussian kernel) $K(\mathbf{x})$ is given by,

$$\widehat{f}(x) = \frac{1}{Nh^d} \sum_{i=1}^{N} K\left(\frac{\mathbf{x} - \mathbf{x_i}}{h}\right) \tag{4.3}$$

where h defines the radius of the kernel, which is usually termed as the bandwidth parameter of the kernel K. The radially symmetric kernel is defined as follows:

$$K(x) = c_{k,d} k\left(\|\mathbf{x}\|^2\right) \tag{4.4}$$

where $c_{k,d}$ represents a normalization constant and k represents the kernel profile. The density estimator in 4.3 when using 4.4 can be written as:

$$\hat{f}_{h,K}(\mathbf{x}) = \frac{c_{k,d}}{Nh^d} \sum_{i=1}^{N} k\left(\left\|\frac{\mathbf{x} - \mathbf{x_i}}{h}\right\|^2\right) \tag{4.5}$$

Taking the gradient of the density estimator in 4.5 and further algebraic manipulation yields,

$$\nabla \widehat{f}_{h,K}(\mathbf{x}) = \widehat{f}_{h,G}(\mathbf{x}) \, m_{h,G}(x) \tag{4.6}$$

where $\widehat{f}_{h,G}(\mathbf{x})$ is proportional to the density estimate at $\mathbf{x}$, and $\mathbf{m}_{h,G}(\mathbf{x})$ is called the mean shift vector.

$$\widehat{f}_{h,G}(\mathbf{x}) = \frac{2c_{k,d}}{Nh^{d+2}} \left[\sum_{i=1}^{N} g\left(\left\| \frac{\mathbf{x}-\mathbf{x}_i}{h} \right\|^2 \right) \right] \tag{4.7}$$

$$\mathbf{m}_{h,G}(\mathbf{x}) = \left[\frac{\sum_{i=1}^{N} \mathbf{x}_i g\left(\left\| \frac{\mathbf{x}-\mathbf{x}_i}{h} \right\|^2 \right)}{\sum_{i=1}^{N} g\left(\left\| \frac{\mathbf{x}-\mathbf{x}_i}{h} \right\|^2 \right)} - \mathbf{x} \right] \tag{4.8}$$

The function $g(x)$ is the derivative of the kernel profile k (i.e. $g(x) = -k'(x)$) and the kernel $G(\mathbf{x})$ is defined as $G(\mathbf{x}) = c_{g,d} g\left(||\mathbf{x}||^2\right)$.

Once the locations of the stationary points of $\widehat{f}_{h,K}$ are determined by the mean-shift procedure, thereafter only the local maxima points are retained. The quality of the density estimator is measured by the mean squared error between the density and its estimate. Further, the data points associated with the same stationary point are considered members of the same cluster. The major computational cost of the mean-shift procedure corresponds to identifying the neighbours of a point in the feature space, defined by the kernel and its bandwidth.

As discussed earlier in chapter 2, our algorithm that we refer to a resource allocating codebook (RAC), which has strong similarities in concept to the mean-shift based method (Jurie and Triggs, 2005), is computationally far simpler in that it is one-pass and achieves very similar or slightly better than the mean-shift based method.

4.2.3 Hierarchical

Given a matrix $\mathbf{X} \in \mathbb{R}^{N \times d}$, the hierarchical clustering takes $N \times N$ measures of similarity between data points as input and assigns each data item to its own cluster, i.e. initially there will be N clusters each containing only one item. In the following phase it iteratively performs the following two steps until all items are clustered into a single cluster of size N:

1. finds the closest pair of clusters and merges them into a single cluster to obtain one cluster less than the previous case.
2. computes the similarities between the new cluster obtained in step-1 and each of the old clusters.

The result of a hierarchical clustering is normally represented in a tree structure called a *dendrogram*. At the highest level of the tree, all the items are contained in one cluster,

at the next highest level there are two and so on until at the lowest level there are N clusters.

Hierarchical clusterings are generally either agglomerative ('bottom-up' clustering) or divisive ('top-down' clustering). The agglomerative procedure has the *single-linkage* and *complete-linkage* methods. When merging two clusters the single-linkage procedure uses the minimum distance between elements of each cluster, whereas the complete-linkage procedure uses the maximum distance between elements of each cluster. There is a hybrid procedure of single-linkage and complete-linkage, called as *average-linkage* procedure that uses the mean distance between elements of each cluster when merging two clusters. The Unweighted Pair Group Method with Arithmetic mean (UPGMA) (Legendre and Lengendre, 2003) uses the average-linkage procedure. The hierarchical-divisive methods proceed in the opposite way of the hierarchical-agglomerative method.

The hierarchical clustering method does not require the number of clusters K as an input similar to the K-means clustering method, but needs a termination condition. The complexity of the hierarchical-agglomerative clustering methods is usually quadratic in N and clusters, while the hierarchical-divisive clustering method is linear in N and clusters. Due to the time complexity of hierarchical-agglomerative clustering methods, they do not scale well and can never undo what was done previously.

In section 4.4, we demonstrate a formulation for hierarchical clustering that sequentially processes the data in a one-pass setting, that we call sequential hierarchical clustering (SHC) method. For a better illustration purpose of the SHC technique, we make use of some applications that are of bioinformatics. SHC, as a clustering technique, has also been tested on a subset of the UCI dataset in order to compare and contrast with a recently well known affinity propagation (Frey and Dueck, 2007) and vertex substitution heuristic (VSH) (Brusco and Köhn, 2008) clustering techniques. The VSH is a well established heuristic for the K-median model.

4.2.4 Affinity propagation

Given a matrix $\mathbf{X} \in \mathbb{R}^{N \times d}$ (representing N points described with respect to d features), then the affinity propagation technique proposed by Frey and Dueck (2007) takes $N \times N$ measures of similarity between data points as input and iteratively passes real-valued messages between data points until a set of clusters gradually emerges. The cluster centres referred to as 'exemplars' are a subset of the real points of the dataset. Given a penalty parameter p and the measurements S of each pair of the data, it maps:

$$f : x_i \rightarrow f(x_i)$$

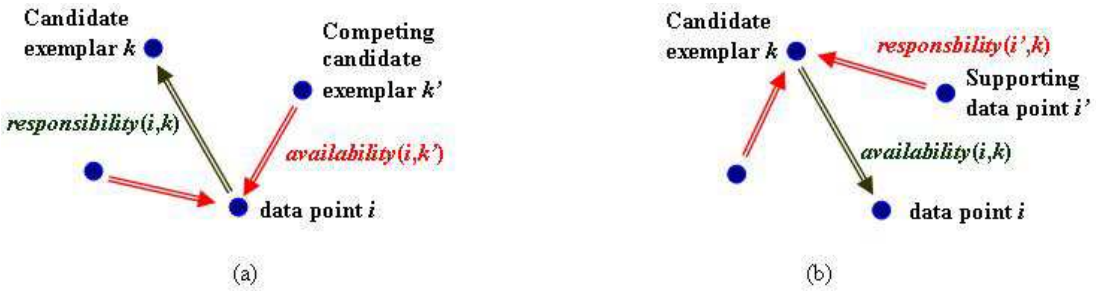


FIGURE 4.4: Message passing process in Affinity propagation. (a) Sending 'responsibilities' (b) Sending 'availabilities'.

such that,

$$\min \sum_{i=1}^{N} S\left(x_i, f\left(x_i\right)\right)$$

where

$$S\left(x_i, x_j\right) = \begin{cases} -d^2\left(x_i, x_j\right) & ; \text{ if } i \neq j \\ -p^* & ; \text{ otherwise} \end{cases}$$

The penalty $p^* \geq 0$ will decide the number of clusters. For example,

$$p^* = \begin{cases} \infty & ; \text{ only one exemplar, i.e., 1 cluster} \\ 0 & ; \text{ every point is an exemplar, i.e., } N \text{ clusters} \end{cases}$$

In each iterative process, a point sends messages ('availability') to every other point to find how confident the particular point can be an exemplar of the others and the other points send a message ('responsibility') how much they want this point to be their exemplar. Figure 4.4 shows an instance of message passing process between data points in affinity propagation.

Frey and Dueck (2007) claim lower computational cost in comparison to the K-centres algorithm, but they do not include the cost of pairwise similarity computations. This is the most expensive stage in large scale problems. While comparing K-centres clustering with affinity propagation method, both clustering techniques are iterative and cope much harder with large datasets. The claimed advantage over K-centres is that the K-centres needs to set the number of clusters in advance while this is not required in the affinity propagation method, but it merely replaces the problem of choosing K with the problem of setting the preference vector. Thus, the problem of choosing p is not yet resolved by affinity propagation. Different preferences p yields different numbers of clusters. The complexity of the affinity propagation algorithm is $\mathrm{O}(N^2 \log N)$.

4.3 Resource-allocating codebook (RAC)

The RAC is an adaptation of the resource allocating network (RAN) family of algorithms (Platt, 1991; Kadirkamanathan and Niranjan, 1993; Yingwei et al., 1997). The RAN, originally proposed by Platt (1991), was developed as a means to overcome the problem of NP-completeness in learning fixed size networks that can be used at any time in the learning process and the learning patterns do not have to be repeated. It either allocates a new unit based on the novelty of a newly seen pattern, or adapts the network parameters by using the standard least means square gradient descent algorithm to fit that observation. Subsequently, Kadirkamanathan and Niranjan (1993) interpreted in a function estimation setting trained by extended Kalman filtering (EKF-RAN). Moreover, Yingwei et al. (1997) proposed a minimal resource-allocation network (MRAN) to overcome the drawback in the RAN, by removing inactive hidden units as learning progresses that yield more compact network. Farran and Saunders (2009) proposed a similar one-pass algorithm, that they call 'Voted Spheres', which couples a hypersphere fitting over the feature space with a voting strategy that yields a novel classification technique for inference from large datasets. Our goal is to construct a visual codebook by discarding visually similar keypoints at the nearest neighbours in a fixed-radius hyperspheres. The RAC carves the input space in a wider span than that which would be found by any density preserving method.

4.3.1 Methodology

The RAC starts by arbitrarily assigning the first data item as an entry in the codebook. When a subsequent data item is processed, its minimum distance to all entries in the current codebook is computed using an appropriate distance metric. If this distance is smaller than a predefined threshold r (radius of the hypersphere), the current codebook is retained and no action is taken with respect to the processed data item. If the threshold is exceeded by the smallest distance to codewords, a new entry in the codebook is created by including the current data item as the additional entry. This process is continued until all data items are seen only once. The pseudocode of the RAC is given in Algorithm 2.

To make the RAC technique more informative to the order of presence of the data, we use an incremental update to shift the cluster centres by means of computing the weighted average of all its points. We denote all visual descriptors as $X \in \mathbb{R}^d$ and the centres of the hyperspheres as $C \in \mathbb{R}^d$, where d is the dimension of the visual descriptors (d=128 in SIFT). We keep records of the weights (number of descriptors) of each hypersphere. Whenever a new descriptor falls into a hypersphere H_i then its centre C_i is redefined by the weighted average of all its previous points and the new point. Thereafter the number of descriptors in H_i is incremented by one. The pseudocode of this updating approach is given in Algorithm 3.

Algorithm 2 Resource-Allocating Codebook

Input: Visual descriptors ($\mathbf{X}$) and radius (r) of the hyperspheres.
Output: Visual codebook ($\mathbf{C}$)

Step 1: $C_1 \leftarrow X_1$ // initialise the codebook
$j \leftarrow 1$ // initial size of the codebook
$i \leftarrow 2$ // subsequent visual descriptor
Step 2: Repeat steps 3 to 4 until $i \leq \text{size}(\mathbf{X})$
Step 3: if $\min \| \mathbf{X_i} - \mathbf{C}\|^2 > r^2$
then create a new hypersphere of r such that,
$j \leftarrow j + 1$
$C_j \leftarrow X_i$
else go to step 4
endif
Step 4: $i \leftarrow i + 1$
Step 5: return $\mathbf{C}$

Algorithm 3 Resource-Allocating Codebook with incremental update of codewords

Input: Visual descriptors ($\mathbf{X}$) and radius (r) of the hyperspheres.
Output: Visual codebook ($\mathbf{C}$)

Step 1: $C_1 \leftarrow X_1$ // initialise the codebook
$n_1 \leftarrow 1$ // initialise the number of descripors in $\mathbf{C}$
$j \leftarrow 1$ // initial size of the codebook
$i \leftarrow 2$ // subsequent visual descriptor
Step 2: Repeat steps 3 to 4 until $i \leq \text{size}(\mathbf{X})$
Step 3: if $\min \| \mathbf{X_i} - \mathbf{C}\|^2 > r^2$
then create a new hypersphere of r such that,
$j \leftarrow j + 1$
$C_j \leftarrow X_i$
$n_j \leftarrow 1$
else $\forall j$, find the k^{th} hypersphere, such that,
$\min \| \mathbf{X_i} - \mathbf{C_j}\|^2 \leq r^2$ and update its location
$\mathbf{C_k} \leftarrow \frac{n_k \times C_k + X_i}{n_k + 1}$
$n_k \leftarrow n_k + 1$
endif
Step 4: $i \leftarrow i + 1$
Step 5: return $\mathbf{C}$

Limited experimentation with updating the codewords did not yield significant changes in accuracy performance (see section 4.3.9.5 for details). A detailed description of the properties of RAC including the computational savings and its wider span in the feature space are provided in sections 4.3.9.1 and 4.3.9.2, respectively. The contribution of codebook entries together with the predefined threshold lead to a partitioning of the space into a set of overlapping hyperspheres when the distance metric used is the Euclidean norm. Local correlations between features could also be modelled in this framework by estimating covariance matrices associated with each vocabulary entry and using a Mahalanobis distance metric, similar to the sequential input space partitioning (SISP)

algorithm in Shadafan and Niranjan (1994), though for simplicity we restrict ourselves to Euclidean distance in this thesis.

In the following sections we describe the details of our experimental setup and the results that support our claims. Experiments were carried out to compare the relative performance of the RAC with the K-means clustering method tested on face and visual object recognition tasks when using SIFT/SURF features. We also compare the proposed technique with a closely related mean-shift based method as in (Jurie and Triggs, 2005). The obtained results support our claims with respect to the RAC.

We performed three different experiments tested on benchmark image datasets:

1. In the first experiment, we present results of RAC compared with K-means when applied on two benchmark face databases, namely the AT&T (Samaria and Harter, 1994) and Yale (Georghiades et al., 2001) faces, for classification. As a baseline we also evaluate our approach with a nearest neighbour (NN) classification.

2. In the second experiment, the performance of RAC is compared with K-means and mean-shift based methods when tested on a set of binary classification tasks of the PASCAL VOC Challenge 2007 image dataset (Everingham et al., 2007).

3. In the third experiment, we carried out binary classifications on the entire PASCAL VOC Challenge 2009 image dataset (Everingham et al., 2009) by comparing the RAC and K-means methods.

Further, we carried out multi-class classification using the RAC method tested on the Xerox7 and PASCAL VOC07 image datasets that are reported in the next chapter (see Table 5.9 and 5.11).

4.3.2 Datasets

The AT&T (previously known as ORL) face database contains 40 persons with 10 images per subject. Images were taken at different times, varying the lighting, facial expressions and facial details. The Yale face database contains 15 persons with 11 images per subject. Images were taken with different facial expressions or configurations: centre-light, with glasses, happy, left-light, without-glasses, normal, right-light, sad, sleepy, surprised and wink. Figure 4.5 shows an example subject of those face databases with different facial expressions and configurations. The PASCAL VOC challenge 2007 and 2009 datasets contain 20 object classes. Some example images of each of the classes of PASCAL07 dataset are shown in Figure 5.6, and Figure 4.7 shows the distribution of classes in this dataset as bars. The PASCAL VOC Challenge, Caltech[1] and Xerox7 image sets are

[1] http://www.vision.caltech.edu/Image_Datasets/

FIGURE 4.5: Example faces with different facial expressions: (top row) AT&T face with different poses, (bottom row) Yale faces with varying lighting conditions.

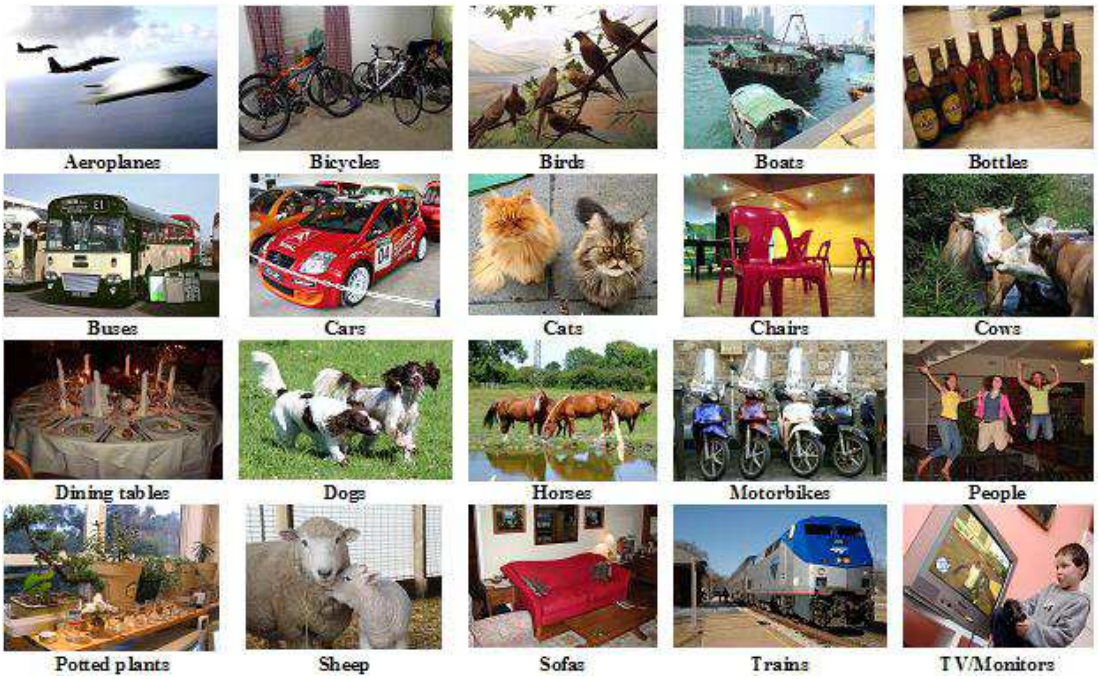


FIGURE 4.6: One image from each of the object categories in PASCAL VOC Challenge 2007 image dataset. These example images show the presence of multiple instances of the same object category in an image under various conditions that make the recognition difficult. In the example image of aeroplanes, different scales of an object category can be noticed, whereas in the image of bicycles different poses can be observed. Occlusion of objects can be observed in the images of buses and cars. Rotational effects can be seen in the image of chairs. An example of a cluttered image can be seen in the case of potted plants.

becoming the standard for measuring visual recognition performance in recent vision papers as we have seen in chapter 2. Ponce et al. (2006) and Pinto et al. (2008) argue that publicly available image collections such as Caltech-101 and PASCAL VOC image sets are lacking in several aspects that can actually mislead the progress in the long-term interest of being able to achieve near human levels of recognition. Detailed discussions about the benchmark image datasets are beyond the scope of this study and tangential to our main point.

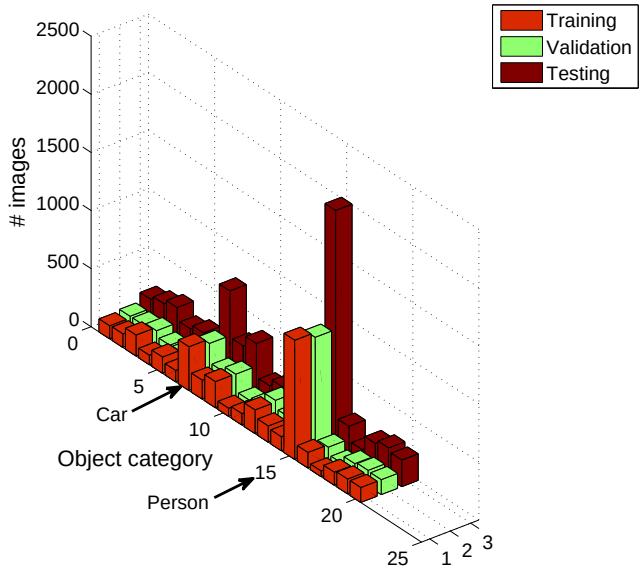


FIGURE 4.7: The number of images in each of the twenty classes of the PASCAL VOC Challenge 2007 dataset shown as bars. It can be seen that the *person* and *car* classes highly dominate other classes.

4.3.3 Experimental setup

In our first experiment, we selected the testing image of each subject in a leave-one-out fashion and the remaining images for training. The number of training images per subject at each run was 9 and 10 for the AT&T and Yale faces, respectively. Since there are 40 subjects in the AT&T faces we used 40×9=360 images for training and 40 images for testing. In the case of the Yale faces we used 15×10=150 images for training and 15 images for testing. This process is carried out until every image is used as a test image per subject. The reported results are the average of these runs performed in a leave-one-out fashion.

In the second experiment, we selected a set of binary classes from the PASCAL07 dataset based either on their appearance-based similarity (e.g. bicycle and motorbike, cow and sheep, bus and train) or on the number of images that are nearly equal from both of the object categories. We performed the classification task for each of the selected binary classes, predicting the presence or absence of an example of that class in the test image.

In the third experiment, we performed the classification for each of the twenty object categories on a balanced dataset which were randomly selected 50 images per interest class and three images per other classes (3×19=57). Further, we performed a classification task on the entire PASCAL09 dataset for each of the twenty classes tested on the validation set defined by the dataset providers. We present the classification results with mean/standard deviation values and paired t-test is used to analyse the significance in performance differences.

4.3.4 Feature extraction

We used SIFT features in our first two experiments, whereas e-SURF (dimension of 128) features were used in experiments performed on the PASCAL09 dataset due to the lesser number of interest points that are feasible for the K-means method. The effective number of SIFT features required in achieving reasonable recognition performance and the robustness of RAC with SIFT features to additive noise level are also tested (see sections 4.3.9.6 and 4.3.9.7).

In experiment 1, SIFT descriptors were automatically extracted from all images of the face databases without pre-processing the raw images. The K-means or RAC method was applied on the training images to construct visual codebooks when SVM was the choice for classification, otherwise instead of constructing a codebook, each keypoint of an image described by SIFT was treated as a feature that was used with the nearest neighbour classifier.

In experiment 2, SIFT features were extracted from the union of training and validation ('trainval') images in the PASCAL07 dataset. Then we applied K-means, mean-shift, and RAC approaches independently to those features to construct codebooks.

In experiment 3, SURF features were extracted from the training images of the PASCAL09 dataset. There were nearly 5.6 million interest points detected from the entire training images without the use of bounding box information. Due to this large scale of interest points and the prohibitive time in clustering using K-means, we randomly selected 10% of the SURF features from each object category to construct a global codebook.

Codebook construction involved features extracted within the provided bounding box information by ignoring objects marked as 'difficult' in all the experiments performed with the PASCAL07 and PASCAL09 datasets. We also ignored bounding boxes with minimum size that are less than 128 pixels. The number of SIFT descriptors for each object category that of the PASCAL VOC Challenge 2007 dataset is provided in section 5.9.

4.3.5 Classification

The simplest classification approach would be a nearest neighbour voting strategy that computes all pairwise Euclidean distances between keypoint representation of a test subject to all labelled subjects in the dataset. This strategy is impractical in all but the smallest of problems. An alternative classification strategy is to map the keypoints derived from an image into a histogram of nearest codeword of a codebook, and then apply support vector machines (SVMs). SVMs are quite naturally designed to perform classification in high dimensional spaces. Classification in this chapter was performed

using one-versus-all linear SVMs. Many researchers (Csurka et al., 2004; Farquhar et al., 2005; Kim and Kweon, 2007; Quelhas et al., 2007; Perronnin, 2008) have used the one-versus-all SVM with linear kernel as it gives a significantly better performance. We also found this to be true in our experience and used it most widely. Although when focusing at multi-class classification tasks we demonstrate a novel idea using SVMs that we describe in section 5.1.4.

For each problem, we estimate the regularisation parameter C (see Equation A.6) of the binary SVMs by means of cross-validation on the training set by using a search space of $\left[2^{-14}, 2^{-13}, 2^{-12}, \cdots, 2^{1}\right]$. The input to the SVMs is the fixed-length feature vector of histograms while, for the nearest neighbour approach, each interest point of a test subject described by SIFT was voted against all the labelled subjects in the training set.

4.3.6 Evaluation criterion

We used recognition rates as performance measure for the face recognition tasks, which is the fraction of the correctly recognised faces to the total number of test faces. In the visual object recognition tasks, we used average precision as performance measure that is widely being used in the recent PASCAL VOC challenges. Average precision is a single-valued measure that is proportional to the area under a precision-recall curve. In order to map the real valued outputs of the SVM into probabilities to compute the average precisions, we trained the parameters of a sigmoid function as described in (Platt, 1999).

$$\text{Recall} = \frac{TP}{TP+FN} \qquad \text{True Positive Rate} = \frac{TP}{TP+FN}$$

$$\text{Precision} = \frac{TP}{TP+FP} \qquad \text{False Positive Rate} = \frac{FP}{FP+TN}$$

Precision-Recall curves are used as an alternative to receiver operating characteristics (ROC) curves. ROC curves shows how the number of correctly classified positive examples varies with the number of incorrectly classified negative examples. In a ROC curve, a point in the upper-left-hand corner represents a good classification result. The point where the true positive rate is the highest and the false positive rate is the lowest is the optimised point (see Figure 4.8).

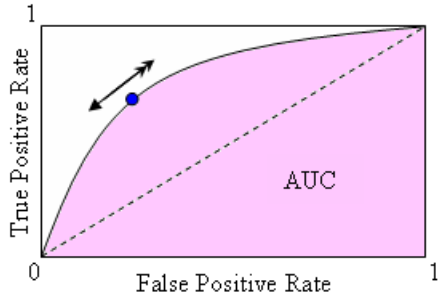
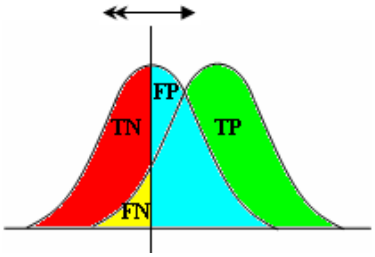


FIGURE 4.8: ROC curve and AUC



	Actual Positive	Actual Negative
Predicted Positive	TP	FP
Predicted Negative	FN	TN

FIGURE 4.9: Confusion matrix

The precision-recall analysis gives more intuitive and sensitive evaluation than the ROC analysis for analysing performance of classifiers using highly unbalanced classes (Davis and Goadrich, 2006). In this curve, the precision is plotted against the recall. recall is the same as true positive rate. The precision-recall evaluation criterion applied in the context of the matching and recognition of image scenes interprets the number of correct and false matches between two images.

4.3.7 Testing results

Results of experiment 1 are shown in Table 4.2. For the AT&T faces, the hyper-parameter of the RAC was r=0.7 and for the Yale faces, r was 0.5 when the test image had the lighting variations, otherwise 1.0. In both face recognition tasks, K-means is set with K=1000.

In Table 4.3 and Table 4.4, we report the recognition results of three independent runs as the means of average precision (MAP) of our experiments 2 and 3, respectively. Each run is carried out by randomly shuffling the order of presence of the images to the clustering process. While we have included the standard deviation for completeness, we note that these are estimates of uncertainty for a very small number of trials. Except when specified, we used r=0.8 with RAC technique that is roughly comparable to the size of K-means based codebook with K=1000. In Table 4.5, we report the results as average precision tested on the validation set ('val') of the PASCAL09 dataset.

TABLE 4.2: Recognition results on the AT&T and Yale faces.

Database	NN	KM+SVM	RAC+SVM
AT&T faces	100%	98.25%	100%
Yale faces (with lighting effect)	86.06%	89.09%	95.15%
Yale faces (without lighting effect)	100%	97.50%	100%

TABLE 4.3: Comparison of three codebook generation methods tested on a selected binary classification tasks from the PASCAL VOC Challenge 2007 dataset. K-means with K=1000. Both mean-shift and RAC codebook sizes are slightly greater than 1000. The parameters of the mean-shift were N=1000 and h=r=0.8. The r of RAC was 0.8.

Objects	KM + SVM	MS + SVM	RAC + SVM
Aeroplane vs Car	**0.7858 ± 0.0095**	0.7805 ± 0.0049	0.7815 ± 0.0071
Bicycle vs Motorbike	0.7721 ± 0.0132	0.7715 ± 0.0024	**0.7846 ± 0.0053**
Bird vs Horse	0.7558 ± 0.0175	**0.7579 ± 0.0175**	0.7578 ± 0.0189
Boat vs TVmonitor	0.7536 ± 0.0181	0.7509 ± 0.0012	**0.7584 ± 0.0012**
Bottle vs Pottedplant	0.6817 ± 0.0034	0.6777 ± 0.0070	**0.6832 ± 0.0024**
Bus vs Train	0.7139 ± 0.0033	**0.7191 ± 0.0069**	0.7185 ± 0.0036
Chair vs Dog	**0.8386 ± 0.0148**	0.8218 ± 0.0048	0.8330 ± 0.0092
Cow vs Sheep	0.6755 ± 0.0152	0.6953 ± 0.0133	**0.7056 ± 0.0105**

4.3.8 Discussion

Experiment 1: SIFT features perform quite well and are robust with different facial expressions and poses, but fail to work under lighting variations. SIFT features based on histograms of local orientation give some tolerance to illumination changes but not significant (Luo et al., 2007). Thus the performance drop in the Yale faces using the SIFT features was due to the varying lighting conditions (see Figure 4.5 (bottom row) for the lighting effects). To test the lighting effect with respect to SIFT features, we removed the images that have lighting variations (centre-light, left-light, and right-light) and tested the remaining images in a leave-one-out fashion. Table 4.2 (bottom row) shows the improved classification results.

Both nearest neighbour and RAC approaches carried out in a leave-one-out fashion outperform the reported results in (Yang et al., 2004) on the AT&T faces (98.3%) and Yale faces (84.24%). The best quoted results in the literature on the Yale faces (100%) are by Branson and Agarwal (2003) in which the images were cropped outside the face contour, aligned, Gabor filtered, z-scored and thereafter the dimensionality of the data is reduced by their proposed approach, structured principal component analysis (SPCA). They trained a perceptron by backpropagation on the training faces and tested on a novel face.

Experiment 2: In Table 4.3, the t-test is not powerful enough as we have only 8 paired samples and they are not necessarily normally distributed. In this case, the Wilcoxon signed rank test is more appropriate. RAC has better classification accuracy than the K-means and mean-shift based methods in 4 out of 8 cases as shown in the Table. The Wilcoxon test carried out with the RAC method comparing with the K-means and mean-shift methods, shows that the performance is comparable at the level of significance $p = 0.25$ and $p = 0.0391$, respectively.

TABLE 4.4: Recognition results as mean average precisions on a balanced subset of the PASCAL VOC Challenge 2007 classification task. K-means with K=1000 and RAC with r=0.8.

Object	KM+SVM	RAC+SVM
Aeroplane	**0.8643 ± 0.0215**	0.8482 ± 0.0098
Bicycle	0.7185 ± 0.0166	**0.7907 ± 0.0171**
Bird	0.6528 ± 0.0095	**0.6558 ± 0.0064**
Boat	**0.8015 ± 0.0185**	0.7819 ± 0.0073
Bottle	**0.7439 ± 0.0355**	0.7172 ± 0.0355
Bus	0.6532 ± 0.0188	**0.6594 ± 0.0182**
Car	0.6933 ± 0.0355	**0.7113 ± 0.0020**
Cat	0.6615 ± 0.0219	**0.7331 ± 0.0224**
Chair	**0.6416 ± 0.0127**	0.6274 ± 0.0102
Cow	0.7084 ± 0.0245	**0.8125 ± 0.0211**
Diningtable	0.6790 ± 0.0130	**0.6998 ± 0.0155**
Dog	0.7432 ± 0.0176	**0.7890 ± 0.0024**
Horse	0.6830 ± 0.0190	**0.7441 ± 0.0096**
Motorbike	0.7345 ± 0.0073	**0.7515 ± 0.0172**
Person	0.6512 ± 0.0120	**0.6894 ± 0.0083**
Pottedplant	0.5915 ± 0.0013	**0.6075 ± 0.0124**
Sheep	0.7810 ± 0.0225	**0.8259 ± 0.0200**
Sofa	**0.7373 ± 0.0156**	0.7241 ± 0.0112
Train	0.7280 ± 0.0120	**0.7582 ± 0.0282**
TVmonitor	0.7377 ± 0.0150	**0.7886 ± 0.0144**
MAP	0.7103 ± 0.0175	**0.7358 ± 0.0145**

In our experience, we were not able to extend the K-means method in clustering nearly 1.7 million $\times\mathbb{R}^{128}$ SIFT descriptors that were extracted from the two dominant object classes ('person' and 'car') of the PASCAL07 dataset to construct the visual codebook owing to prohibitive execution times. Therefore, we used our RAC technique to construct a codebook. RAC constructed the codebook with an average size of 2400 in an average time of 111700 seconds ($\approx$ 31 hours). The mean average precision of the Person vs Car classification was 0.8253 ± 0.0040.

Experiment 3: In Table 4.4, RAC is best for 15 out of 20 classes, whereas K-means is best for 5 out of 20 classes. We evaluated the experimental results based on the statistical pairwise t-test (1-tailed) and may conclude that RAC outperforms K-means at the level of significance $p = 0.0023$ on the randomly selected balanced dataset. In Table 4.5, the results of RAC are comparable to K-means or perform slightly better.

4.3.9 Properties of RAC

We have seen various clustering techniques and a novel approach that we refer to RAC. Now we explore the properties of the RAC technique.

TABLE 4.5: Recognition results as average precision on the PASCAL VOC Challenge 2009 classification task tested on the validation set. K-means with K=3000 and the RAC with r=0.75

Object	KM+SVM	RAC+SVM
Aeroplane	0.4728	**0.5167**
Bicycle	0.2907	**0.2954**
Bird	0.2891	**0.2998**
Boat	**0.1967**	0.1756
Bottle	0.0668	**0.0718**
Bus	**0.2022**	0.1347
Car	0.1568	**0.2081**
Cat	**0.2623**	0.1962
Chair	0.1254	**0.1404**
Cow	0.0520	**0.1312**
Diningtable	**0.0528**	0.0516
Dog	**0.2212**	0.1749
Horse	0.0899	**0.1543**
Motorbike	**0.1736**	0.1643
Person	0.4266	**0.4286**
Pottedplant	**0.0698**	0.0471
Sheep	0.1161	**0.1688**
Sofa	**0.1014**	0.0794
Train	0.2227	**0.2517**
TVmonitor	**0.2624**	0.2237
MAP	0.1926	**0.1957**

4.3.9.1 The coverage of the feature space

"*RAC looks for visual codebook that has a wider span of the input space than that found by K-means method*".

To illustrate this property, we consider four images each from two object categories, duck and horse (see Figure 4.10), that of the Amsterdam Library of Object Images (ALOI) colour image collection (Geusebroek et al., 2005). The objects are subjective to pose, angle and illumination changes taken against a clear background. These 8 selected images result in $1559\times\mathbb{R}^{128}$ SIFT descriptors that were clustered into 160 clusters using the K-means and RAC techniques.

Figure 4.11 shows a two dimensional projection of those 160 cluster centres projected on a plane defined by the first two principal components of those clustered data. While projecting the SIFT based cluster centres from 128 to 2 dimensions masks much of the distribution, it can still be visualized that RAC gives codebook prototypes spanning in a wider range of space than K-means.

Figure 4.12 indicates the logarithmic count of the data points that fall in each bin of the

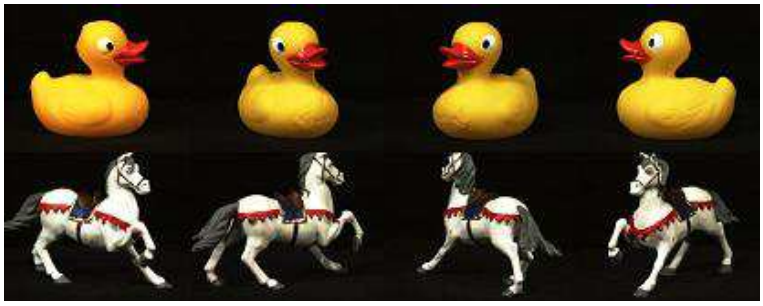


FIGURE 4.10: Example images (duck and horse) of the Amsterdam Library of Object Images (ALOI).

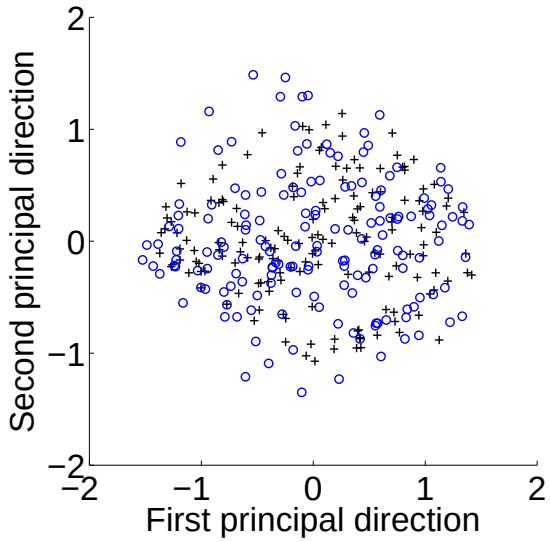
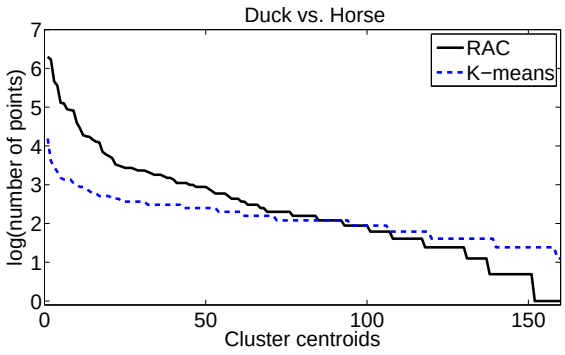


FIGURE 4.11: Projections of visual codebooks along the first two principal components of the data. K-means clustering ('+') yields cluster centres that preserve data density while RAC ('o') generates codebook that is more spaced out.

clusters ordered descendingly. We used the duck and horse images shown in Figure 4.10. It can be seen that K-means has clusters with more equal data points that have a narrower span in the feature space, while RAC has more unequal points that span more widely and capture rare points in the feature space. Furthermore, Figure 4.13 (a) and (b) illustrate the partitioning of a feature space using the K-means and RAC techniques, respectively. Note that the cluster centres found by K-means populate the densest part of the feature space, whereas RAC finds centres that each represent a distinct part of the feature space.



(a)

FIGURE 4.12: Plot of the logarithmic count of the data that falls in each histogram bin of the clusters generated by K-means (dotted line) and RAC (solid line) methods applied on the duck vs horse task. It can be seen that K-means has clusters with more equal data points that have a narrower span in the feature space, while RAC has more unequal points that span more widely and capture rare points in the feature space.

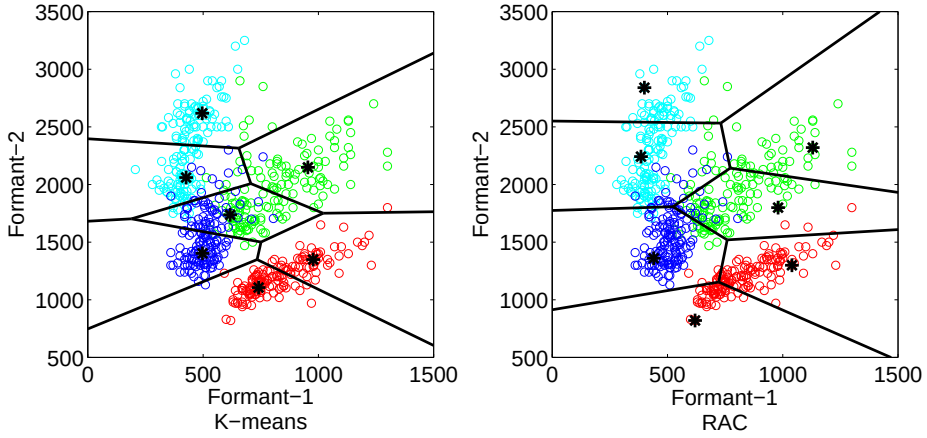
4.3.9.2 Computational savings

"*RAC has far lower computational cost than K-means clustering*".

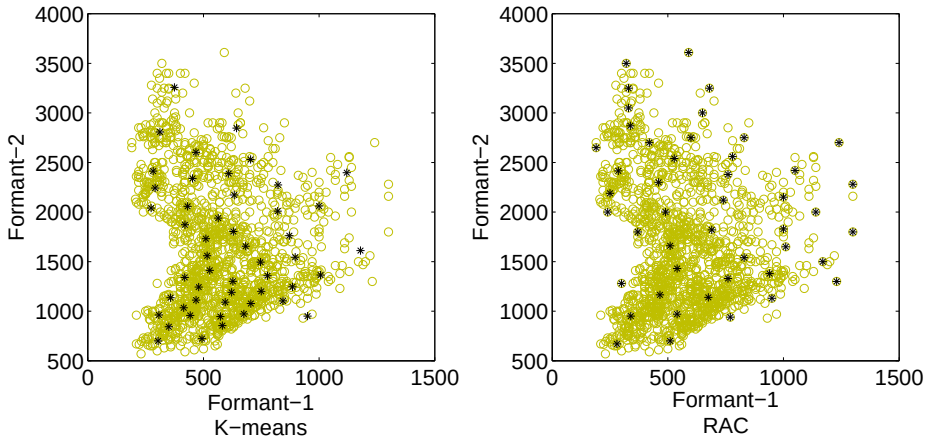
We provide the following empirical evaluations to support this claim.

- In the previous duck vs horse example, the $1559 \times \mathbb{R}^{128}$ SIFT descriptors were clustered into 160 clusters using K-means in 19.79 seconds while RAC only needed 0.58 seconds to complete the one-pass execution on a desktop computer with an Intel Core 2 running at 2.4GHz and 4GB of RAM.
- The $105000 \times \mathbb{R}^{128}$ SIFT descriptors used in our multi-class classification of the Xerox7 image dataset (see section 5.4) were clustered into 1000 clusters using K-means in an average time performing each fold of the 10-fold cross-validation in 536803 seconds ($\approx$ 149 hours), while RAC only needed an average time of 1147.7 seconds ($\approx$ 19 minutes) on a cluster computer with a dual core Xeon running at 2.6GHz and 48GB of RAM.
- In the selected two class classification tasks (listed in Table 4.3) of the PASCAL07 dataset, the features extracted from the 'trainval' set of Boat vs TVmonitor produce $134175 \times \mathbb{R}^{128}$ SIFT descriptors, which were clustered into 1000 clusters by K-means in 1029833.31 seconds ($\approx$ 286 hours) while RAC only needed 4900.52 seconds ($\approx$ 82 minutes) on the same cluster computer mentioned above.

These examples show the *drastic reduction* in the computational needs when using the RAC technique compared with the usage of the K-means clustering method. The time complexity of RAC depends on the size of the candidate cluster set $\mathbf{C}$, i.e. we compare each newly seen pattern to all existing clusters.



(a)



(b)

FIGURE 4.13: An example of feature space partitioning using K-means clustering (left) and RAC approach (right) applied on the Peterson and Barney (1952)'s dataset that are vowel sounds characterised by the first two formant frequencies. (a) plot of 7 clusters and their corresponding centres found on 4 selected classes (shown in different colours) out of 10 classes of the vowel data (b) plot of 50 cluster centres found on the entire vowel dataset. It can be seen that the cluster centres found by K-means are around densely populated areas, whereas the centres of RAC are around lesser occurrence data points. It is worth noting that K-means results in approximately the same number of data points associated with each cluster while RAC has clusters with a highly skewed distribution. Thus in RAC outlier (or less occurrence) data get included as part of the codebook.

4.3.9.3 The hyper-parameter

The novelty threshold used in RAC is regarded as a hyper-parameter, which denotes the maximum threshold between features that may be considered similar. As such, the hyper-parameter determines whether two patches describe the same codeword. The choice of the radius r has the same set of difficulties associated with the choice of K in K-means and the size of sub samples N, radius r, and mean-shift radius h in mean-shift based techniques. Our approach to setting r is to take a small sample of the data, compute all pairwise distances between these samples and set the threshold, so that an approximate target codebook size is achieved. Another way to estimate r is cross-validating over the training set.

4.3.9.4 Sensitivity of RAC with the order of presence of data

In order to evaluate the sensitivity of K-means to its initial random selection of centroids and the order of presentation of the data to RAC technique (without updating the centroids), we considered the Bicycle vs Motorbike classification sub task from the PASCAL09 dataset. The results given in Table 4.6 are the mean average precision of the 10-fold cross-validation. The reported results were tested on the validation set of the PASCAL09 dataset.

TABLE 4.6: Sensitivity of RAC with the order of presence of data: Recognition results for Bicycle vs Motorbike classification using the PASCAL VOC challenge 2009 dataset. K-means with K=1000 and RAC with r=0.8

Fold	KM+SVM	RAC+SVM
1	0.5843	0.5736
2	0.6175	0.5917
3	0.5871	0.6097
4	0.5642	0.5979
5	0.5968	0.5882
6	0.5964	0.5972
7	0.6103	0.5882
8	0.6022	0.5905
9	0.5866	0.6097
10	0.6026	0.6050
mean	0.5948	**0.5952**
std	0.0151	0.0112

All the experiments that we reported in this chapter in comparison with K-means show that RAC is slightly sensitive to the order of presentation of data, similar to the random initial selection of cluster centres in the K-means method, but has less variation in comparison.

4.3.9.5 RAC vs updated RAC

To compare the effectiveness of updating codewords of RAC (see Algorithm 3) to the non-updated version (see Algorithm 2) with respect to the order of presence of the data, we performed experiments with the PASCAL09 dataset using SURF-128 features. Table 4.7 shows the obtained results tested on the validation set of the PASCAL09 dataset. Limited experimental results show that the updated version performs slightly better than the non-updated one. While comparing the little overhead in updating the codewords in Algorithm 3, it is still better to use the approach in Algorithm 2. Therefore, we used the non-updated version of RAC in all our experiments.

TABLE 4.7: RAC vs updated RAC: Recognition results as average precision on the PASCAL VOC Challenge 2009 classification task tested on the validation set. RAC with r=0.8

Object	RAC	updated RAC
Aeroplane	**0.4991**	0.4868
Bicycle	0.2693	**0.2921**
Bird	0.2919	**0.3054**
Boat	0.2317	**0.2362**
Bottle	**0.1091**	0.0652
Bus	0.2109	**0.2125**
Car	0.1385	**0.1696**
Cat	**0.2292**	0.2120
Chair	**0.1153**	0.1122
Cow	**0.0784**	0.0704
Diningtable	0.0482	**0.0644**
Dog	**0.2063**	0.1750
Horse	0.0809	**0.1014**
Motorbike	0.1613	**0.1720**
Person	**0.4261**	0.4168
Pottedplant	0.0533	**0.0543**
Sheep	0.0495	**0.1099**
Sofa	0.0892	**0.0957**
Train	0.2004	**0.2996**
TVmonitor	**0.2429**	0.2067
MAP	0.1866	**0.1926**

4.3.9.6 Robustness to noise level

To check out the robustness of RAC with SIFT to additive noise level we performed experiments on the face recognition and visual object classes classification tasks with additive Gaussian noise by varying the standard deviation from 10 to 100 with increments of 10. Figure 4.14 shows an example subject in the AT&T faces with respect to the increase of additive Gaussian noise.

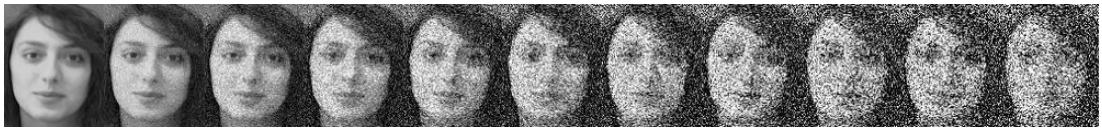
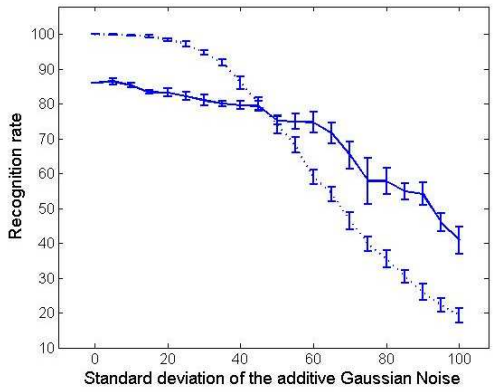
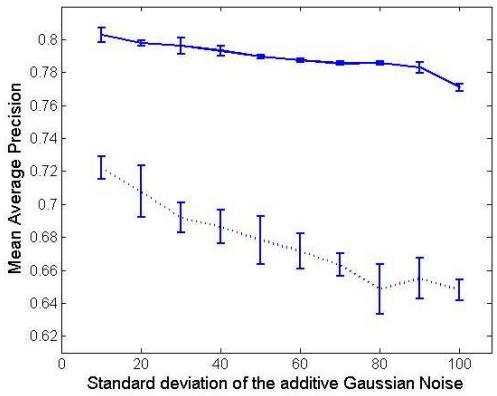


FIGURE 4.14: An example subject (left) in the AT&T face dataset. Left to right the standard deviation of the additive Gaussian noise varies from 10 to 100 with increments of 10.



(a)



(b)

FIGURE 4.15: Recognition performance vs additive Gaussian noise. (a) Face recognition task (dotted line indicates the AT&T faces and solid lines indicates the Yale faces) over 10 independent runs. (b) Visual object classes classification task (dotted line indicates Cow vs Sheep and solid lines indicates Bicycle vs Motorbike) over three independent runs.

When we compare the performance versus noise level on the face recognition tasks (see Figure 4.15(a)), SIFT features perform quite better until the standard deviation $\sigma = 20$. Thereafter, the performance drops constantly with the increase of noise level. After $\sigma = 50$, the variations of performance in the Yale faces are very high in comparison with AT&T faces. This is because of the lighting variations and higher noise levels that make the recognition harder on the Yale faces. In the visual object classes classification tasks (see Figure 4.15(b)), the performance is quite better until $\sigma = 20$. Thereafter,

the performance drops constantly with the increase of noise level. The variations of performance in the Cow versus Sheep are very high in comparison with Bicycle versus Motorbike, due to the relative hardness in classifying those objects (see classification results in Table 4.3).

4.3.9.7 Reduced SIFT descriptors for reliable performance

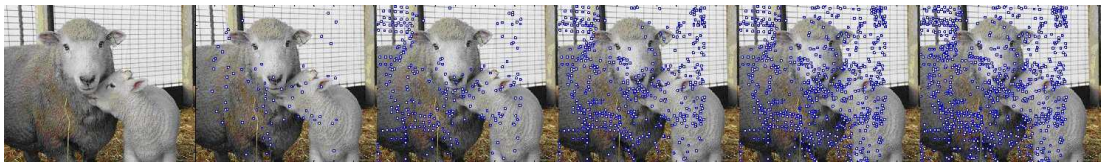


FIGURE 4.16: (left) An example image of sheep in the PASCAL VOC 2007 dataset. p% of keypoints that are detected in the image are shown as boxes. Left to right p varies from 5, 25, 50, 75 and 100.

We carried out experiments with the selected two class classification tasks Cow vs Sheep and Bicycle vs Motorbike from the PASCAL07 dataset, using a reduced subset of the extracted SIFT keypoints from each image in order to observe the classification performances. The keypoints were sorted in descending order by their scales. We then used p% of keypoints from each image/object by varying p from 5 to 100 with increments of 5 (see Figure 4.16).

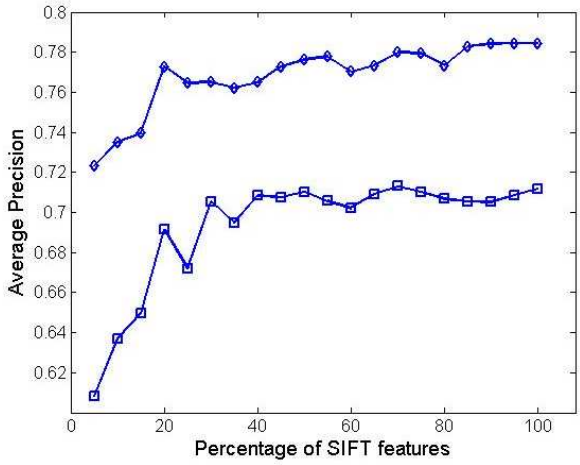


FIGURE 4.17: Average Precision vs p% of SIFT features (diamond plot indicates the Bicycle vs Motorbike and square plot indicates Cow vs Sheep classification tasks).

The performance of the classification increases rapidly with the increased number of keypoints (see Figure 4.17). From our experiments we may conclude that 70% of the SIFT keypoints suffices for reliable classification which saves the computational cost in constructing the visual codebook, histogramming and classification processes. However, note that the entire SIFT keypoints outperformed any of the reduced ones, so selection of keypoints when sorted in descending order by their scales should only be used if the computational cost of the recognition system needs to be reduced.

4.3.10 Outlier rejection during histogramming process

In text classification analogy, the most frequently occuring words in English, such as the articles (*a*, *an*, and *the*) are not the discriminant words between document classes. Similarly the bag-of-features approach discussed in this thesis can suffer from interest points that are detected in background clutter or outliers. The goal of employing an outlier rejection technique is to improve the classification accuracy by rejecting outliers and patterns for which the classifier has a low confidence.

The basic assumption in our approach is that visual keypoints detected from noise terms or background will have a greater distance to the smallest distance to centroids in the codebook than some predetermined threshold. For instance, 'house' is not an object of interest in the PASCAL07 dataset, but there are many cases where an object that is considered in this image dataset appears with a 'house' (e.g. many images have the presence of a person or an animal with a house). This non-interest object will not be considered during the construction of a visual vocabulary as the features used are found within the bounding boxes. However, when detected keypoints from the entire image are mapped to the learnt vocabulary entries much information is lost, resulting in less classification accuracy.

We make use of a threshold θ to reject outliers being assigned to the bag-of-features. When estimating θ we used the validation set of the PASCAL07 dataset with those SIFT features that fall within the bounding box for a specific object category. This makes the outlier rejection feasible at the later stage. θ is estimated as follows:

$$\theta = \frac{\sum_{i=1}^{N} \min \sqrt{\| \mathbf{D}(i) - \mathbf{C} \|^2}}{N}$$

where, $\mathbf{D}$ the set of N visual keypoints extracted within the bounding box of the training and/or validation set, and $\mathbf{C}$ the codebook constructed by the RAC approach.

Each visual keypoint vk_j of the image set is then computed with the codebook $\mathbf{C}$, to obtain its nearest centroid with distance d_j. $\forall_j \epsilon\ \mathbf{I}$, if $d_j \leq \theta$ then the corresponding histogram is updated with vk_j, otherwise no action is carried out.

Figure 4.18 shows one image from the motorbike and bicycle object categories with the entire SIFT keypoints and the reduced outliers. Our results using the outlier rejecting technique gave us 0.7909 ± 0.0014 on the Bicycle vs Motorbike and 0.7129 ± 0.0041 on the Cow vs Sheep classification tasks, which shows nearly 0.5% increase in performance with less uncertainty over three independent runs.

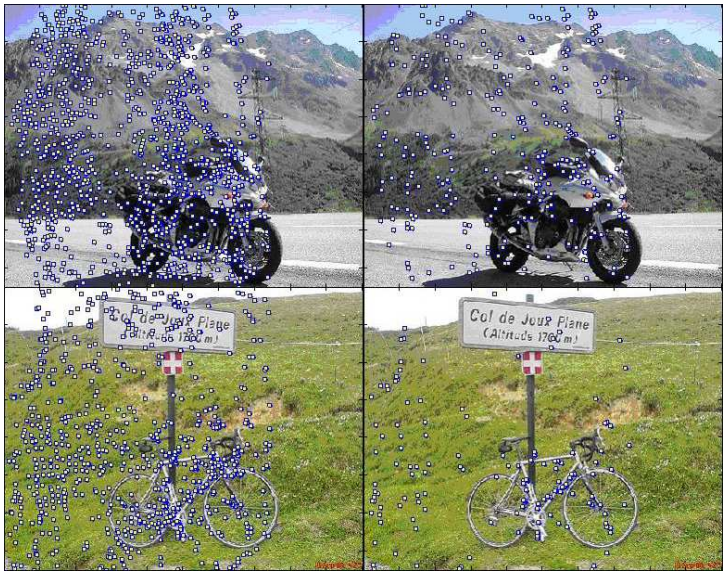
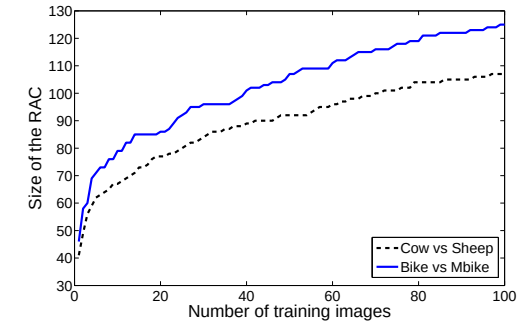


FIGURE 4.18: Images on the left of each category (motorbike and bicycle), show the entire SIFT keypoints found, and images on the right, show the reduced outliers using the outlier rejection technique. On average 70% of the entire detected keypoints are rejected as outliers.

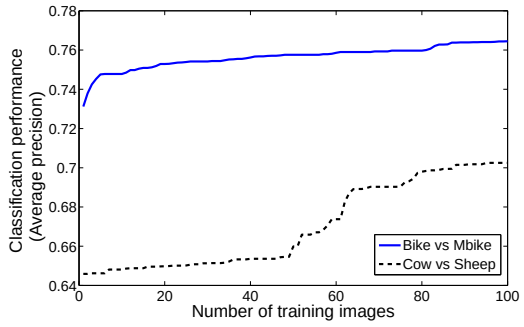
4.3.11 Constructing codebook over incrementally presented images with classifier training

Here, we focus on a similar concept that was proposed by Yang et al. (2008), in which a codebook is constructed with the classifier training. Our main interest is to construct a codebook that acquires information about objects in an incremental way. The strategy that we used to design discriminant codebook is to update it over sequentially arriving training images and the output classifier accounts for class specific discriminant features. At the arrival of each training image belonging to an interest or non-interest object category, only the novel information in the codebook will be absorbed as additional entries. The construction of a codebook in this context is achieved by the RAC technique.

We carried out experiments with the Bicycle vs Motorbike and Cow vs Sheep binary classification tasks of the PASCAL07 dataset. The scope of this work was to see whether the size of RAC would saturate with the increase of training images that are processed subsequently, similar to the human visual processing system. To test this effect we used a larger value for r of RAC and observed that it loses its discriminative power while its size stays roughly constant after seeing many images of the same object category (see Figure 4.19).



(a)



(b)

FIGURE 4.19: Classification performance and size of RAC versus the size of the learning set with training classifier. 100 images were processed subsequently to construct a codebook and the output classifier accounts for class specific discriminant features. In both experiments the r was set to 1.0 and SIFT descriptors were used.

4.4 Sequential hierarchical clustering (SHC)

We further investigate whether a similar formulation for hierarchical clustering by sequentially processing the data in a one-pass setting can be designed. Computational saving in such an approach will come from not having to evaluate all pairwise similarities between data items. This clearly is not possible for all the data as a measure of the scale of the distribution is required. Thus, the approach we take involves the construction of an initial tree of hierarchical clustering by processing a random subset of the data in batch mode. This establishes a scale structure of the input space. To a cluster structure formulated in this way, any further data may be sequentially included, adaptively changing the structure of the cluster tree at a heavily reduced cost of computing all the pairwise similarities between patterns.

4.4.1 Methodology

We construct an initial hierarchical tree by computing all pairwise similarities between a small subset of the data, and then passing these to the single-round-MC-UPGMA algorithm (Loewenstein et al., 2008). Following the construction of the initial tree using the single-round-MC-UPGMA, the remaining data are sequentially processed using Algorithm 4. Whenever a new pattern $\mathbf{x}_i$ arrives for clustering, its similarity distance d to the *root* of the current hierarchical tree is computed. If d is greater than a predefined threshold θ, a new root is created having the current pattern $\mathbf{x}_i$ and the previous root as its children, and as a consequence the depth of the tree increases by one. The value of the new root is assigned with the arithmetic mean of all the leaf nodes. However, if d is less than θ, the nearest child of the current node is retrieved. If the distance of $\mathbf{x}_i$ to this child node is also smaller than θ then we continue to repeat finding the closest child until either the distance to the current node is greater than θ or we reach a leaf node. In either of the two cases, $\mathbf{x}_i$ is created as a sibling to the node under consideration, and $\mathbf{x}_i$'s value is propagated up the tree to update its ancestors. Our algorithm for sequentially updating a hierarchical tree is shown as pseudo code in Algorithm 4.

Algorithm 4 Update the initial hierarchical tree in an online fashion

Input: Root of the initial tree (CurNode), the new pattern (NewNode), and the novelty threshold (θ)
Output: Updated hierarchical tree

```
simdist_CN ← similarity_distance(CurNode, NewNode)
if (simdist_CN ≤ θ) then
  (⋆) Children ← getChildrenOf(CurNode)
  if (Children == NULL) then
    Make NewNode as a sibling of CurNode and update ancestors
  else
    {CurNode has children}
    nearestNode ← min (similarity_distance(Children, NewNode))
    if (nearestNode ≤ thresh) then
      CurNode ← nearestNode
      Go to (⋆)
    else
      Make NewNode as sibling of CurNode and update ancestors
    end if
  end if
else
  Make NewNode as sibling of CurNode by creating a new root
end if
```

By adjusting θ, one can obtain different numbers of clusters at different levels of granularity. This threshold is seen as a hyperparameter in the algorithm and can be tuned in different ways. The threshold θ used in this thesis was determined from the data sample used to construct the initial tree. θ was set as the sum of the pairwise Euclidean

distances between the patterns in this data sample. The leaf nodes of the hierarchical tree are the input patterns, and each intermediate node (up to and including the root) contains the arithmetic mean of every leaf node it represents. Because of this, we need not traverse the entire tree during the update process.

We use the capitals dataset[2] for illustration and evaluation purposes of the SHC algorithm, as the structure of the clusters is known. The dataset consists of monthly average temperatures of 17 capital cities. We numbered the capital cities in the following way: Tallinn (1), Beijing (2), Berlin (3), Buenos Aires (4), Cairo (5), Canberra (6), Cape Town (7), Helsinki (8), London (9), Moscow (10), Ottawa (11), Paris (12), Riga (13), Rome (14), Singapore (15), Stockholm (16), Washington (17). In Figure 4.20, we show the tree constructed using single-round-MC-UPGMA on the entire capitals dataset, which is identical to the tree depicted on the data's website. Figure 4.21 shows how our approach inserts the last two points into the tree, given that the first 15 points were used for the construction of the initial tree. Cutting at the second level gives us the exact same four clusters obtained by cutting the tree in Figure 4.20 at the same level.

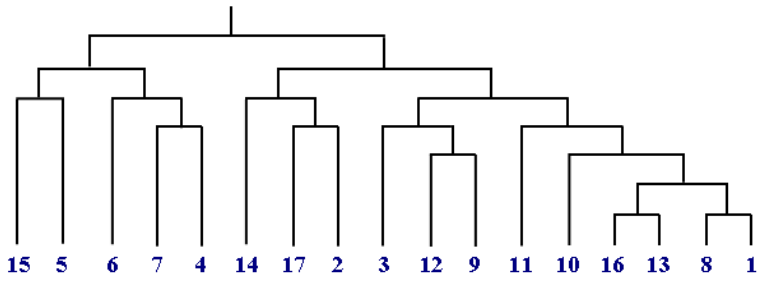


FIGURE 4.20: Hierarchical tree constructed using the single-round-MC-UPGMA on the entire capitals dataset.

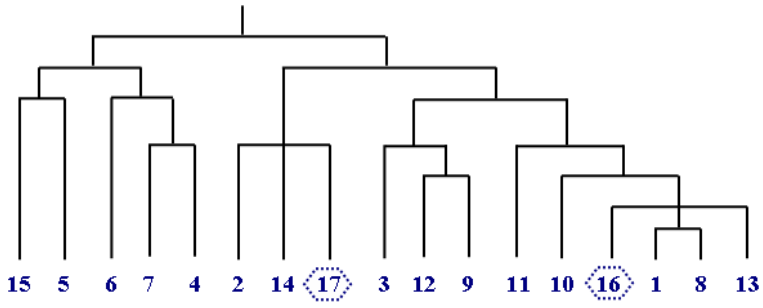


FIGURE 4.21: Hierarchical tree constructed by our approach in an online fashion with the aid of an initial tree constructed by the single-round-MC-UPGMA technique. The initial tree was constructed with the first 15 capitals in the dataset. Capitals Stockholm and Washington (depicted in dotted hexagons) were sequentially inserted using Algorithm 1.

To illustrate the importance of having a good initial tree, we use El-Sonbaty and Ismail (1998)'s method on the entire capitals dataset (shown in Figure 4.22). El-Sonbaty and Ismail (1998) proposed an on-line hierarchical clustering algorithm based on the single-linkage method that finds at each step the nearest k patterns with the arrival of a new

[2] http://www.quretec.com/HappieClust/

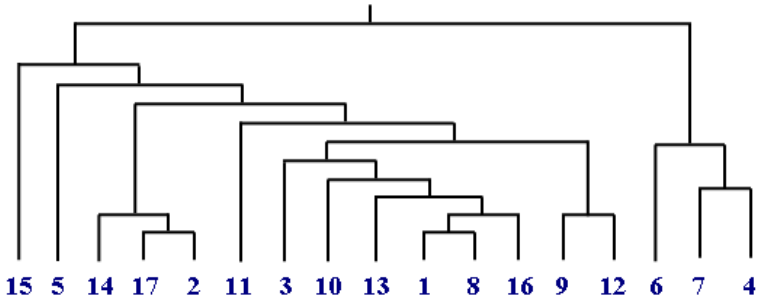


FIGURE 4.22: Hierarchical tree constructed by the approach proposed in (El-Sonbaty and Ismail, 1998) on the entire capitals dataset.

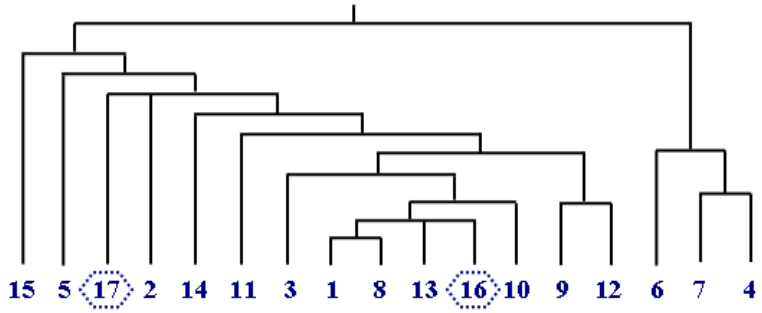


FIGURE 4.23: Hierarchical tree constructed by our approach with the aid of an initial tree constructed by the method proposed in (El-Sonbaty and Ismail, 1998) on the capitals dataset.

pattern and updates the nearest seen k patterns so far. Finally the patterns and the nearest k patterns are sorted to construct the hierarchical dendrogram. While they claim their method is sequential, at the arrival of each data item they compute similarity to *all* the data seen previously. Thus there is little computational saving in their method, and it is equivalent to re-training a new model at the arrival of new data.

Figure 4.23 shows how Algorithm 4 inserts the last two points, given that the first 15 points were used for the initial tree using El-Sonbaty and Ismail (1998)'s method. Even though Algorithm 4 inserted the last two points correctly with respect to Figure 4.22, the same four clusters obtained by the trees in Figure 4.20 and Figure 4.21 are not attainable due to the incorrect initial tree.

4.4.2 Testing results on benchmark datasets

The experimental results that we report in this section are the background for the algorithm development. We evaluate SHC on two bioinformatic datasets. The first is Eisen et al. (1998)'s gene expression clusters consisting of ten clusters formed by their clustering algorithm. We make the weak assumption that the clusters published by these authors are perfect associations and quantify how close our approach gets to this solution. The second dataset is from a protein fold classification problem, constructed by Ding and Dubchak (2001) on a subset of the structural classification of proteins (SCOP)

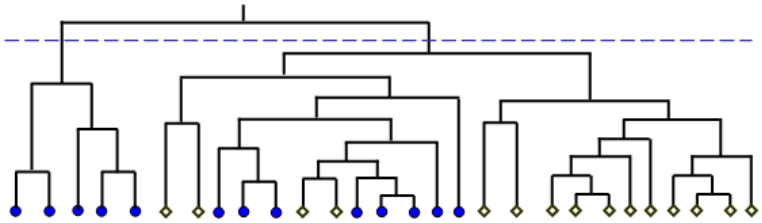


FIGURE 4.24: Tree constructed by single-round-MC-UPGMA on the two selected folds Beta: ConA-like lectins (depicted as diamonds) and Alpha: four-helical up-and-down bundle (depicted as filled circles) of the SCOP subset. We used the whole 28 data points for the construction of this tree.

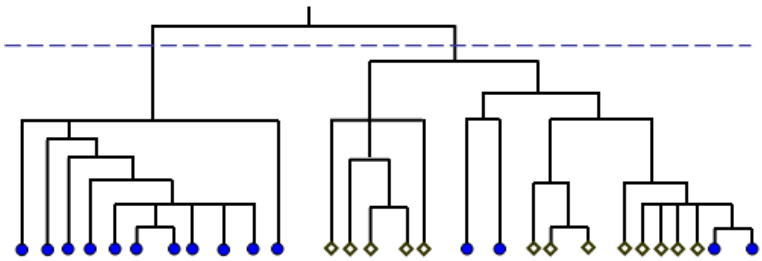


FIGURE 4.25: Tree constructed by the proposed approach with the aid of an initial tree constructed by single-round-MC-UPGMA on the two selected folds Beta: ConA-like lectins (depicted as diamonds) and Alpha: four-helical up-and-down bundle (depicted as filled circles) of the SCOP subset. We used 20 out of 28 data for the construction of the initial tree and used the rest in an online manner.

database (Lo Conte et al., 2000). The data statistics of the SCOP subset are given in section 5.2.2. This is essentially a classification problem, but we apply clustering to it (without using the class labels) and evaluate how well the resulting cluster membership matches clusters returned by the single-round-MC-UPGMA applied on the entire subset. We combined both training and testing sets provided in (Ding and Dubchak, 2001). Figure 4.24 shows the hierarchical tree constructed by the single-round-MC-UPGMA technique and Figure 4.25 shows how Algorithm 4 inserts eight points, given that 20 points were used for the initial tree using the single-round-MC-UPGMA method.

To quantify the quality of clusters, we use the F1 measure (see Equation 4.9), widely used in information retrieval literature, and the results given in Table 4.8 are the average of the 10 runs where we randomised the initial subset and the order of presentation of the remaining data.

$$F1 = \frac{2 \times precision \times recall}{precision + recall} \tag{4.9}$$

Furthermore, we compare the SHC technique with the affinity propagation (AP) (Frey and Dueck, 2007) and vertex substitution heuristic (VSH) (Brusco and Köhn, 2008) techniques using a subset of the UCI datasets (dataset 1: synthetic control chart time series and dataset 2: page block classification). VSH is an efficient heuristic implementation of the p-median model.

Dataset	No. of data	No. of data used for the initial tree	F1-measure
Eisen (B & D)	20	4	1.0
Eisen (C & I)	104	26	1.0
Eisen (C, B & I)	113	25	1.0
Eisen (C, B, D & I)	124	30	0.9247 ± 0.0652
SCOP (1) & (2)	28	20	1.0
SCOP (3) & (4)	151	100	0.9921 ± 0.0086

TABLE 4.8: Experimental results of the sequential hierarchical clustering performed on Eisen's data and a subset of SCOP database. In the SCOP subset (1) Beta: ConA-like lectins, (2) Alpha: Four-helical up-and-down bundle, (3) Beta: Immunoglobulin-like beta-sandwich, and (4) A/B: beta/alpha (TIM)-barrel.

In comparing SHC with AP and VSH, the preference parameter of AP (see section 4.2.4) was set to the minimum of the input similarities, and the p of VSH was set to the desired number of clusters as shown in Table 4.9. For the SHC, we used 400 and 2773 points for the construction of an initial tree using the single-round-MC-UPGMA when applied on the dataset 1 and 2, respectively. The SHC outperforms the AP and VSH at a *d*rastically reduced computational time. The reported time (in seconds) includes the time of computing the similarities between data points, the construction of an initial tree in the case of SHC, and the clustering time. The reported error in Table 4.9, is the sum of the Euclidean distances of intra-clusters, divided by the total number of clusters.

Dataset	Method	#Instances	#Clusters	Error	Time (in sec)
1	AP	600×60	7	1.73×10^5	9.37
	VSH			1.72×10^5	7.60
	SHC			$\mathbf{1.39\times10^5}$	**1.75**
2	AP	5473×10	2	3.15×10^{10}	12280.97
	VSH		5	2.78×10^9	1528.91
	SHC		4	$\mathbf{1.65\times10^6}$	**198.79**

TABLE 4.9: Performance comparison of SHC with AP and VSH techniques: Dataset 1. Synthetic Control Chart Time Series, and 2. Page Block Classification.

4.4.3 Discussion

In this work we demonstrated an algorithm for on-line hierarchical clustering that we refer to is a sequential hierarchical clustering (SHC) technique. The approach depends on the construction of an initial clustering stage using a random subset of the data. This establishes a scale structure of the input space. Subsequent data can be processed sequentially and the tree adapted constructively. We have shown that on small problems such an approach does not degrade the quality of the clusters obtained while saving computational cost. Further, we also compared the SHC technique with other stan-

dard methods: affinity propagation and vertex substitution heuristic. Our experimental results clearly show the better performance of SHC in comparison with the affinity propagation and VSH techniques with a drastic reduction in the clustering time.

This technique could be significantly improved with an appropriate choice of the novelty threshold θ. θ can be better estimated by taking into account the inter-cluster and/or intra-cluster information of the initial tree. This can be subsequently updated after the insertion of a newly arrived pattern. Another better way of estimating θ might be to use local thresholds associated with each parent or level of the tree, instead of a global threshold. The greatest benefit of the SHC technique lies in its application on very-large datasets.

4.5 Fisher score based codebook

Here we explore an alternate approach to visual vocabulary selection by considering the discriminability of a candidate codeword with respect to the two classes without use of a cluster analysis. When visual patch descriptors are extracted from an image corpus, some of them carry important information to make immediate representation of a codebook discriminatively powerful, while others contribute very little. Selecting these useful and important keypoints will lead to the construction of a compact and discriminant codebook. The class separability can be immediately evaluated once two visual keypoints are merged to an existing codebook by measuring the likelihood ratio or Fisher score. We use the Fisher score (Duda et al., 2000) as the class separability measure. Higher Fisher score yields more discriminative power. Therefore we selected visual keypoints with higher Fisher score in constructing a more compact and discriminative codebook. This approach is straightforward and could be applied in an online fashion to construct a discriminant codebook.

In this approach, we select a random subset of visual keypoints from a training set. Each codebook of the interest and non-interest objects was then initialised with a keypoint at random, respectively. Following the initialisation step, the remaining visual keypoints are sequentially processed one from each class in a pair-wise manner to the existing codebooks by measuring the Fisher score (F) that is compared with a pre-computed Fisher score ($\hat{F}$). $\hat{F}$ is computed over the initial random sub-sample of the visual keypoints. The existing codebook will be updated with these pairs of keypoints only if $F \geq \hat{F}$. This process is continued until a desired codebook size is achieved or all the visual keypoints are processed. The pseudocode of this approach is given in Algorithm 5.

To show the effectiveness of this approach, we did experiments with the Cow vs Sheep and Bicycle vs Motorbike tasks from the PASCAL07 dataset. Results are of three independent runs that indicate the mean and standard deviation of the classification.

Algorithm 5 Codebook construction based on Fisher score

Input: Visual descriptors $\mathbf{D}_1$ of target class C_1 and $\mathbf{D}_2$ of non-target classes C_2 extracted from the training set

Output: Category-specific (target and non-target) codebooks CB_1 and CB_2

$\mathbf{X} \leftarrow$ randomly chosen subset of $\mathbf{D}_1$ of size n_x, and
$\mathbf{Y} \leftarrow$ randomly chosen subset of $\mathbf{D}_2$ of size n_y
compute: $\mu_x \leftarrow \frac{1}{n_x}\sum_{i=1}^{n_x} X_i$

$$\sigma_x^2 \leftarrow \frac{\sum_{i=1}^{n_x}(X_i-\mu_x)^2}{n_x}$$

similary compute μ_y and σ_y^2
compute: Fisher score $\hat{F} \leftarrow \frac{(\mu_x-\mu_y)^2}{\sigma_x^2+\sigma_y^2}$

initialise: $\mathbf{CB}_1 \leftarrow \mathbf{D}_1(1,:)$ and $\mathbf{CB}_2 \leftarrow \mathbf{D}_2(1,:)$
$N \leftarrow \min(size(\mathbf{D}_1, \mathbf{D}_2))$
K a desired size of the codebooks, initially set to N
$i \leftarrow 2$ and $j \leftarrow 1$
while $(i \leq N)$ and $(j \leq K)$ **do**
  $keypoint1 \leftarrow \mathbf{D}_1(i,:)$ and $keypoint2 \leftarrow \mathbf{D}_2(i,:)$
  compute: Fisher score $F \leftarrow \{(\mathbf{CB}_1, keypoint1), (\mathbf{CB}_2, keypoint2)\}$
  if $(F > \hat{F})$ **then**
    merge: $\mathbf{CB}_1 \leftarrow \mathbf{CB}_1 \cup keypoint1$
    $\mathbf{CB}_2 \leftarrow \mathbf{CB}_2 \cup keypoint2$
    $j \leftarrow j + 1$
  end if
  $i \leftarrow i + 1$
end while

The trend of the classification performance with the increase of the codebook size is shown in Figure 4.26.

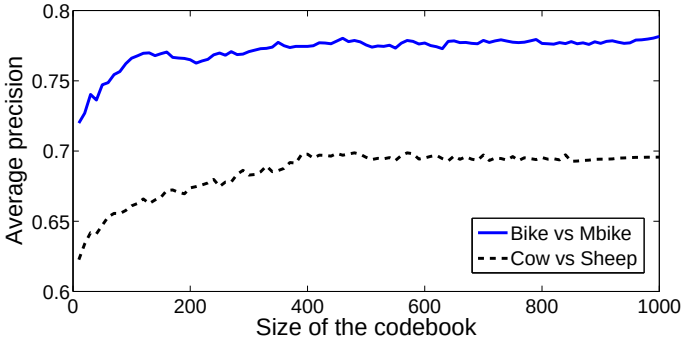


FIGURE 4.26: Classification performance of the two selected classes of the PASCAL07 dataset on the Fisher score based codebook.

In comparing the Fisher score based codebook with other techniques listed in Table 4.3, we considered the size of the codebook to be 1000. In this case, the Fisher score based codebook gave us 0.7816 ± 0.0018 on the Bicycle vs Motorbike and 0.6951 ± 0.0025 on the Cow vs Sheep classification tasks. In both tasks, the performance is lower than the performance with the RAC approach, but slightly better than the K-means based

technique. This technique could be further improved by updating the $\hat{F}$ whenever keypoints are merged to an existing codebook.

4.6 Summary

In this chapter we have demonstrated a one-pass sequential strategy that can be used to construct visual codebooks slightly outperforming traditional approaches in visual object categorisations. Our algorithm achieves this performance at drastically reduced computing times, because apart from an initial scan through a small subset to determine length scales, each data item is processed only once. This addresses the most important bottleneck of scalability with large size modern datasets with respect to the codebook model based object recognition systems. In addition to computational and recognition performance, our approach is also fundamentally different from traditional approaches where it is not the density of detected patches one needs to retain in the codebook but the coverage across the feature space. We have demonstrated RAC with a computationally much simplified algorithm compared to what others have achieved.

We have given a complete description of the RAC method along with its properties and shown a variant of RAC whose learning is incremental with training images, and the output classifier accounting for class specific discriminant features. Also, a pruning strategy has been employed to detect outliers during the computation of frequency histograms. Our results show the need of an outlier detection technique to improve the classification accuracy.

Along the way, we have also demonstrated a sequential hierarchical clustering technique. The approach depends on the construction of an initial clustering stage using a random subset of the data. This establishes a scale structure of the input space. Subsequent data can be processed sequentially and the tree adapted constructively. We have illustrated SHC on small bioinformatics problems and shown that such an approach does not degrade the quality of the clusters obtained while saving computational cost.

Yet another approach has been explored without the use of cluster analysis upon which the discrimination is based adaptive in size by measuring the Fisher scores between the class of interest. Our experimental results show that the latter approach can be used effectively with a carefully chosen subset of features that yields the optimal threshold in advance. In contrast, the Fisher based technique is less expensive in computation as it is free from cluster analysis.

Chapter 5

Multi-class classification for visual object recognition

Multi-class classification is an important but still ongoing research issue in machine learning. Several state-of-the-art approaches in recognising multiple visual object categories reduce the problem into a binary classification task. Such reductions allow one to leverage sophisticated classifiers for learning. These models are typically trained independently for each object category of interest against the non-interest object categories, to tackle multi-class classification. In this thesis, we demonstrate a novel learning architecture for multi-class classification tasks using support vector machines (see Appendix A for an overview of the background theory of SVMs) that has been sensibly combined using existing ideas. This technique is faster in testing compared to directed acyclic graph (DAG) SVMs (Platt et al., 2000), while maintaining comparable performance to the standard multi-class classification techniques.

Classification is one of the standards of the machine learning task. Many techniques such as Fisher's Linear Discriminant Analysis (LDA), Naive Bayes, Perceptron, Neural Networks, Hidden Markov Models (HMMs), and Support Vector Machines (SVMs) are employed in various classification tasks. In general SVMs outperform other classifiers in their generalisation performance (Burgues, 1998).

In multi-class classification each training point belongs to exactly one of the different classes. The goal is to construct a function which, given a new data point, will correctly predict the class to which the new data point belongs. Multi-class classification algorithms fall into two broad categories: the first type directly deals with multiple values in the target field, the second type breaks down the multi-class problem into a collection of binary class sub-problems and then combines them to make a full multi-class prediction. More generally, the second type contains a set of binary SVMs.

5.1 SVM-based multi-class classifier

There are three standard techniques frequently employed by SVMs to tackle multi-class problems, namely one-versus-one (OVO), one-versus-all (OVA), and directed acyclic graph (DAG).

To illustrate those standard techniques diagrammatically, we make use of four classes (i.e. k =4) selected from the Peterson and Barney (1952)'s dataset, consisting of 10 classes that are vowel sounds characterised by the first two formant frequencies. A circle in the diagram (Figure 5.2 - 5.5) denotes a binary SVM classifier.

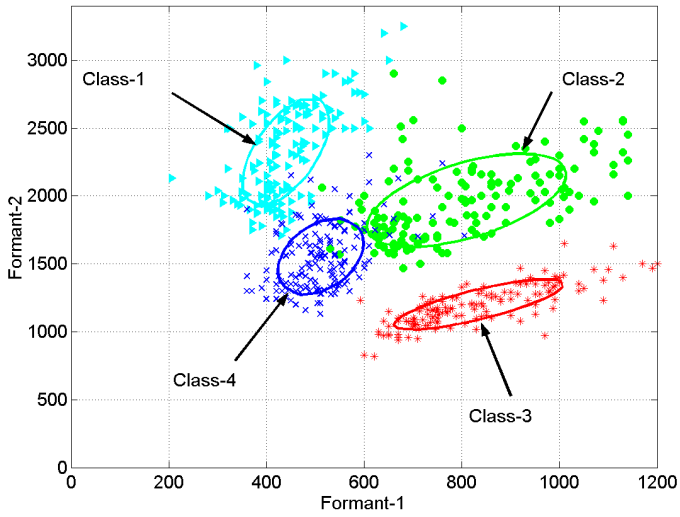


FIGURE 5.1: Distribution of the first two formants of four classes selected from the Peterson and Barney (1952)'s vowel dataset.

5.1.1 One-versus-one

OVO method is implemented using a "max-wins" voting strategy (Debnath et al., 2004). This method constructs one binary classifier for every pair of distinct classes, in total it constructs *k(k-1)/2* binary classifiers, where k is the number of classes (see Figure 5.2). The binary classifier C_{ij} is trained with examples from the i^{th} class and j^{th} class only, where examples from class i take positive labels while examples from class j generally take negative labels. For an example x, if classifier C_{ij} predicts x is in class i, then the vote for class i is increased by one, otherwise the vote for class j is increased by one. The max-wins strategy then assigns x to the class receiving the highest voting score.

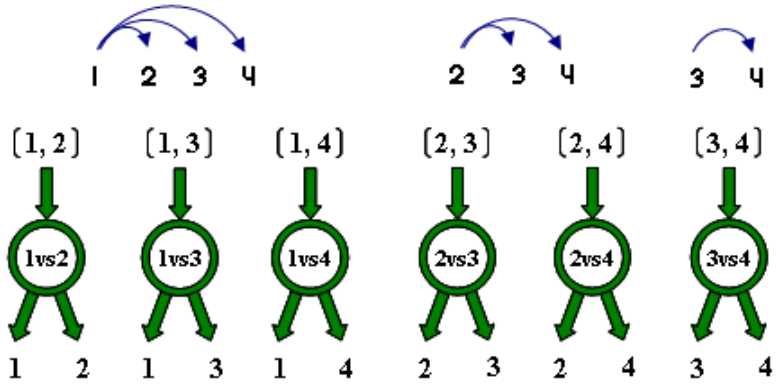


FIGURE 5.2: OVO-based SVM classifiers.

5.1.2 One-versus-all

OVA method is implemented using a "winner-takes-all" strategy (Rifkin and Klautau, 2004). It constructs k binary classifier models (see Figure 5.3). The i^{th} binary classifier is trained with all the examples in the i^{th} class with positive labels, and the examples from all other classes with negative labels. For an example x, the winner-takes-all strategy assigns it to the class with the highest classification boundary function value.

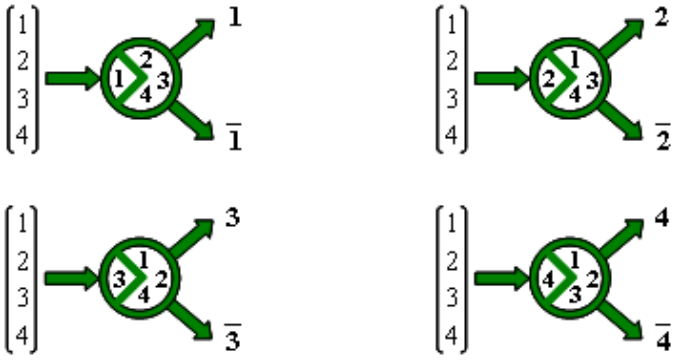


FIGURE 5.3: OVA-based SVM classifiers.

5.1.3 Directed acyclic graph

DAG-based SVMs are implemented using a "leave-one-out" strategy (Platt et al., 2000). The training phase of the DAG is the same as the OVO method, solving *k(k-1)/2* binary classifiers. In the testing phase it uses a rooted binary directed acyclic graph which has *k(k-1)/2* internal nodes and k leaves. Each node is a classifier C_{ij} from OVO. An example x is evaluated at the root node and then it moves either to the left or the right depending on the output value. Figure 5.4 shows the DAG architecture and the classification problems at each node for finding the best class out of four classes. The equivalent list state for each node is shown next to that node. Limited experimentation with re-ordering the list did not yield significant changes in accuracy performance (Platt et al., 2000).

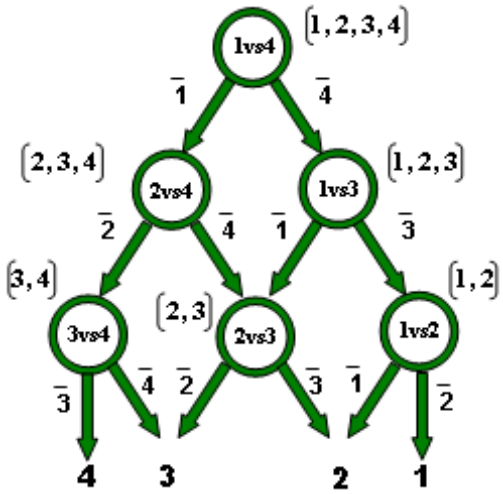


FIGURE 5.4: DAG-based SVM classifiers.

However, DAG suffers from having a long evaluation time as it needs to proceed until a leaf node is reached to make a decision on any input pattern. This problem becomes worse when k becomes large. In our work, we demonstrate a new learning architecture that we call unbalanced decision tree (UDT) to relieve the problem of excessive testing time in DAG, while maintaining accuracy comparable to those standard techniques.

5.1.4 Unbalanced decision tree

UDT attempts to improve existing methods based on DAG and OVA-based techniques for multi-class pattern classification tasks. Several standard techniques, namely OVO, OVA, and DAG, are compared against UDT by some benchmark datasets. Our testing results indicate that UDT is faster in testing compared to DAG, while maintaining accuracy comparable to those standard algorithms tested. UDT is general, and could be applied to any classification task in machine learning in which there are natural groupings among the patterns.

UDT (see Figure 5.5) implements the OVA-based concept at each decision node. Each decision node of UDT is an optimal classification model. The optimal model for each decision node is the OVA-based classifier that yields the highest performance measure. Starting at the root node, one selected class is evaluated against the rest by the optimal model. Then the UDT proceeds to the next level by eliminating the selected class from the previous level of the decision tree. The UDT terminates when it returns an output pattern at a level of the decision node, while DAG needs to proceed until a leaf node is reached. In contrast,we can say that UDT uses a "knock-out" strategy with at most *(k-1)* classifiers to make a decision on any input pattern and is an example of a 'vine' structured testing strategy (Blanchard and Geman, 2005). The pseudocode of UDT is given in Algorithm 6.

Algorithm 6 Unbalanced decision tree for multi-class classification

Input: Training set (*train*), validation set (*val*), testing set (*test*), number of classes ($N > 2$) and set of parameters Θ of SVMs

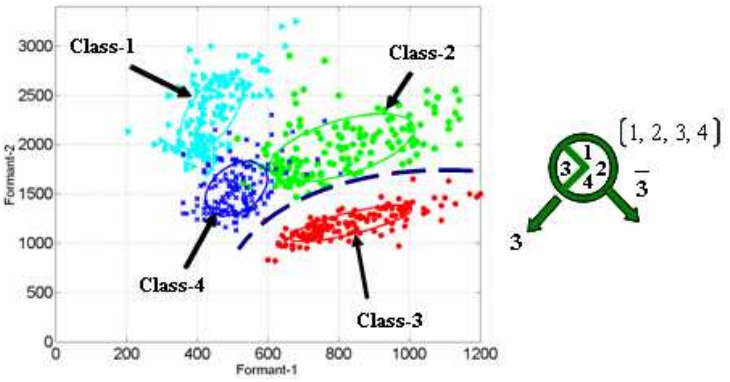
Output: UDT consisting of N-1 one-vs-all SVM classifiers and class *predictions* of *test*

// initialisation
$k \leftarrow \{1, 2, \cdots, N\}$
$depth \leftarrow 1$ // initial depth of the UDT

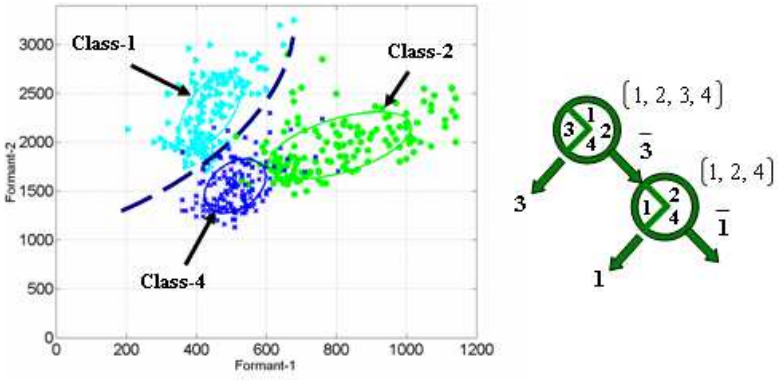
// Training phase
while $N \geq 2$ **do**
  $score \leftarrow \{\}$
  for all class $i \in k$ **do**
    for all parameter $j \in \Theta$ **do**
      // in case of linear kernel, the parameter C is tried over a range of values, whereas in RBF kernel, a set of combinations of parameters (C,γ) are tried over a range of values
      $R_j \leftarrow$ classification rate of the i-vs-all classifier using *train* and *val* with Θ_j
    end for
    find θ_{opt} that corresponds to the maximum of R_j in Θ
    $score(i) \leftarrow$ classification rate obtained from the i-vs-all classifier with θ_{opt} on *val*
  end for
  // finding the optimal decision node at *depth* of UDT
  UDT(*depth*) $\leftarrow$ n-vs-all classifier, such that $\max_n(score)$
  $depth \leftarrow depth + 1$
  // eliminate nth class from *train* and *val*; update the following
  $k \leftarrow k - \{n\}$
  $N \leftarrow N - 1$
end while

// Testing phase
$predictions \leftarrow \{\}$
$t \leftarrow 1$ // the first test data
Repeat steps up to ($\star$) until $\boldsymbol{x}_t \in test$
$depth \leftarrow 1$
while $depth < (N - 1)$ **do**
  if UDT(*depth*) predicts $\boldsymbol{x}_t$ is in class i **then**
    // i is the class of interest at *depth*
    $predictions \leftarrow predictions \cup \{i\}$
    Go to ($\star$)
  else
    $depth \leftarrow depth + 1$
  end if
end while
// $\boldsymbol{x}_t$ has reached the leaf node of the UDT
if UDT(*depth*) predicts $\boldsymbol{x}_t$ is in class i **then**
  // i is the class of interest at leaf node
  $predictions \leftarrow predictions \cup \{i\}$
else
  // j is the other class at leaf node
  $predictions \leftarrow predictions \cup \{j\}$
end if
($\star$) $t \leftarrow t + 1$
return *predictions*

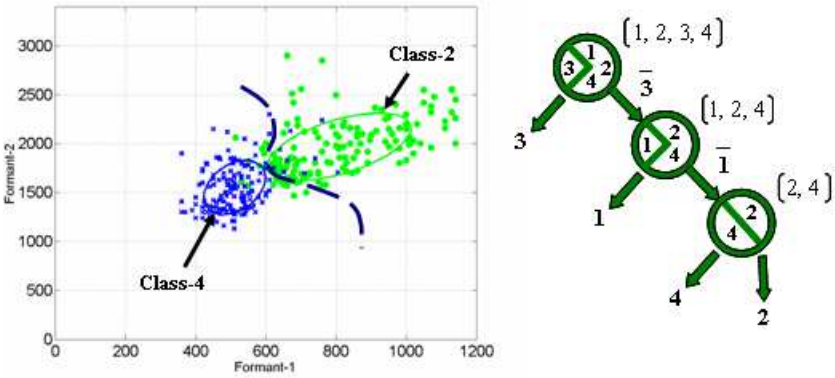
We illustrate the construction of UDT step-by-step from Figure 5.5(a) to 5.5(c) during the training phase by using the example shown in Figure 5.1, with four classes. In this example with the UDT procedure, 3-versus-all classifier is the optimal model at the root node as class 3 is easily separable from the rest (see Figure 5.5(a)). In the next level the training data with class 3 labels are eliminated. At this level, 1-versus-all classifier is the optimal model (see Figure 5.5(b)). At the leaf node the remaining training data with class 1 labels are eliminated and 4-versus-all classifier becomes the optimal model (see Figure 5.5(c)).



(a)



(b)



(c)

FIGURE 5.5: Step-by-step construction of UDT during training phase.

5.2 Datasets

The aim of this experimental part of multi-class classification is to assess the relative performance of UDT with the standard techniques OVO, OVA, and DAG employed by SVMs. We tested the UDT on three different benchmark databases to perform the comparisons that are used in different contexts. The first collection is the University of California, Irvine (UCI) repository of machine learning databases (Newman et al., 1998), the second is a subset of the Structural Classification of Proteins (SCOP) database (Ding and Dubchak, 2001), and the third is the Xerox-7 image dataset. Furthermore, we compare the multi-class classification performed on the PASCAL VOC Challenge 2007 dataset with the RAC and K-means methods.

5.2.1 UCI dataset

We used the *iris*, *wine*, *glass*, *vowel*, *vehicle*, *segment*, and *letter* datasets that we briefly describe as follows.

- The *iris* plant dataset is a well known dataset in the pattern recognition literature. The dataset contains three classes (Iris-virginica, Iris-versicolor, and Iris-setosa), four features and 150 instances (50 in each of the three classes), where each class refers to a type of the iris plant.
- The *wine* dataset consists of the results of a chemical analysis of wines derived from three different wineries in the same region. The dataset contains thirteen features describing various properties of wine and 178 instances (59 in class 1, 71 in class 2, and 48 in class 3).
- The *glass* dataset is used to identify the origin of a sample of glass through chemical analysis. It consists of nine features and 214 instances. The distribution of instances by class is as follows: 70 float processed building windows, 17 float processed vehicle windows, 76 non-float processed building windows, 13 containers, 9 tableware, and 29 headlamps.
- The *vowel* dataset is used in speaker independent recognition of the eleven steady state vowels of British English. The dataset contains ten features and 990 instances.
- The *vehicle* dataset is used to classify a given silhouette as one of four types of vehicles. It consists of 18 features and 846 instances. The distribution of instances by class is as follows: 212 Opel, 217 Saab, 218 double-decker bus, and 199 Chevrolet van.

- The *segment* dataset is a subset drawn randomly from another database of seven outdoor (brickface, sky, foliage, cement, window, path and grass) images that were hand segmented to create a classification for every pixel resulting in 2310 instances (330 in each of the seven classes) with 19 features.
- The *letter* dataset is used to identify each of a large number of black-and-white rectangular pixel displays as one of the 26 capital letters (from A to Z) in the English alphabet, 16 features and 20000 instances.

5.2.2 SCOP dataset

Computational biology in determining structure similarity is one of the major applications of machine learning techniques such as nearest neighbours and SVMs. We did experiments with a protein fold classification problem originally studied by Ding and Dubchak (2001) and more recently by Girolami and Zhong (2007). The task is to devise a predictor of 27 distinct SCOP classes from a set of low homology protein sequences. Six different data representations are available characterizing: amino acids composition (C), predicted secondary structure (S), hydrophobicity (H), polarity (P), Van der Waals volume (V) and polarizability (Z). Each C, S, H, P, V, and Z comprised around 20 features. The statistics of the subset of SCOP dataset are listed in Table 5.1.

5.2.3 Xerox7 dataset

For the computer vision based multi-class image classification task we consider the Xerox-7 image dataset. The Xerox-7 dataset contains 1776 images in seven classes. This image dataset was originally used by Csurka et al. (2004). Figure 5.6 shows some example images of the seven object categories and Table 5.2 shows the image dataset statistics.

FIGURE 5.6: One image from each of the object categories in Xerox7 dataset (bikes, buildings, phones, faces, books, trees, and cars). The images contain presence of multiple instances of the same object category, variable poses, and significant amounts of background clutter which makes the classification much harder.

TABLE 5.1: Dataset statistics of the non-redundant subset of 27 SCOP folds

Fold		**#Training**	**#Testing**
	Globin-like	13	6
	Cytochrome c	7	9
α:	DNA-binding 3-helical bundle	12	20
	4-helical up-and-down bundle	7	8
	4-helical cytokines	9	9
	Alpha: EF-hand	7	9
	Immunoglobulin-like β-sandwich	30	44
	Cupredoxins	9	12
	Viral coat and capsid proteins	16	13
	ConA-like lectins/glucanases	7	6
β:	SH3-like barrel	8	8
	OB-fold	13	19
	Trefoil	8	4
	Trypsin-like serine proteases	9	4
	Lipocalins	9	7
	TIM-barrel	29	48
	FAD-binding motif	11	12
	Flavodoxin-like	11	13
	NAD(P)-binding Rossmann-fold	13	27
α/β:	P-loop containing nucleotide	10	12
	Thioredoxin-like	9	8
	Ribonuclease H-like motif	10	14
	Hydrolases	11	7
	Periplasmic binding protein-like	11	4
	β-grasp	7	8
$\alpha+\beta$:	Ferredoxin-like	13	27
	Small inhibitors, toxins, lectins	14	27
Total		**313**	**385**

TABLE 5.2: Xerox-7 image dataset statistics

Classes	Bikes	Books	Buildings	Cars	Faces	Phones	Trees
#data	125	142	150	201	792	216	150

5.3 Experiments on benchmark datasets

The experiments in this section were mainly carried out for the development of the UDT method. In experiment 1, the experimental setup was such that, for the larger dataset *letter* we used 25% for testing, and a reduced training set was used by training only on 70% of the training set and validating on the other 30% of the training set. For smaller datasets a 10-fold cross validation was carried out on the datasets. All the training data were scaled to be in [-1, 1], then test data were adjusted using the same

linear transformation. For each dataset, we considered linear and radial basis function (RBF) kernels in the experiment. Kernel parameters for the standard techniques were taken from Hsu and Lin (2002). For UDT, an initial experiment was performed to determine the optimal parameter(s) for each kernel type with a range of values of the cost parameters (C) for the linear kernel model, and a set of combinations of parameters (C,γ) for the RBF kernel model.

In experiment 2, we carried out a 10-fold cross validation on the datasets listed in Table 5.1. Data were scaled to be in [-1, 1]. For each dataset, we considered linear and RBF kernels in the experiment. Initial experiments were performed with the standard techniques and UDT classifier SVMs to determine the optimal parameter(s) for each kernel type. A range of values of $C = [2^{-2}, 2^{-1}, \ldots, 2^{11}, 2^{12}]$ and $\gamma = [2^{-10}, 2^{-9}, \ldots, 2^{3}, 2^{4}]$ were tried in the initial experiment.

5.3.1 Testing results

For experiment 1, we present the optimal parameter C for linear kernel and the corresponding classification rates in Table 5.3, the optimal parameters (C,γ) for RBF kernel and the corresponding classification rates in Table 5.4. Also we present the testing time for solving the optimal model, averaged over the 10-fold cross validation runs in Table 5.5 and Table 5.6, respectively. Similarly for experiment 2, we present the optimal parameter(s) and the corresponding classification rates in Table 5.7 and Table 5.8.

TABLE 5.3: A comparison of a subset of the UCI dataset using linear kernel

Dataset	**OVO**		**OVA**		**DAG**		**UDT**
	C	rate	C	rate	C	rate	rate
iris	2^{4}	96.00	2^{12}	95.33	2^{8}	96.67	96.00
wine	2^{-2}	98.33	2^{2}	98.33	2^{-2}	98.33	97.22
glass	2^{8}	64.11	2^{5}	61.56	2^{4}	65.54	65.22
vowel	2^{5}	85.43	2^{11}	48.83	2^{6}	81.08	73.68
vehicle	2^{5}	80.16	2^{12}	74.37	2^{5}	80.87	77.69
segment	2^{12}	95.152	2^{12}	90.952	2^{11}	95.801	95.671
letter	2^{2}	84.760	2^{0}	59.060	2^{4}	83.660	74.800

5.3.2 Discussion

Experiment 1: Considering the training phase of the UDT, it requires longer training time in finding the optimal model for each decision node of the tree. However, this undesirable training time can be greatly reduced by finding the order of classifiers based on their performances during the selection of the root node, and fix this order to form the hierarchy of the decision tree. In contrast, we may conclude by the results obtained,

TABLE 5.4: A comparison of a subset of the UCI dataset using RBF kernel

Dataset	**OVO**		**OVA**		**DAG**		**UDT**
	(C,γ)	rate	(C,γ)	rate	(C,γ)	rate	rate
iris	$(2^{12}, 2^{-9})$	96.00	$(2^{9}, 2^{-3})$	96.00	$(2^{12}, 2^{-8})$	96.67	93.33
wine	$(2^{7}, 2^{-10})$	98.33	$(2^{7}, 2^{-6})$	98.33	$(2^{6}, 2^{-9})$	98.33	92.78
glass	$(2^{11}, 2^{-2})$	71.73	$(2^{11}, 2^{-2})$	71.33	$(2^{12}, 2^{-3})$	72.69	67.52
vowel	$(2^{4}, 2^{0})$	99.62	$(2^{4}, 2^{1})$	99.24	$(2^{2}, 2^{2})$	99.43	97.55
vehicle	$(2^{9}, 2^{-3})$	85.46	$(2^{11}, 2^{-4})$	86.75	$(2^{11}, 2^{-5})$	86.17	84.14
segment	$(2^{6}, 2^{0})$	96.970	$(2^{7}, 2^{0})$	97.359	$(2^{11}, 2^{-3})$	96.970	97.143
letter	$(2^{4}, 2^{2})$	97.570	$(2^{2}, 2^{2})$	97.640	$(2^{4}, 2^{2})$	97.590	96.360

TABLE 5.5: UCI dataset: A comparison of testing time (in seconds) using linear kernel

Dataset	**OVO**	**OVA**	**DAG**	**UDT**
iris	0.06	0.06	0.46	0.40
wine	0.05	0.05	0.47	0.39
glass	0.32	0.13	1.50	1.26
vowel	1.20	0.33	7.44	4.29
vehicle	0.20	0.22	4.98	3.35
segment	0.94	0.47	21.30	15.39
letter	143.77	25.21	3682.02	1969.65

TABLE 5.6: UCI dataset: A comparison of testing time (in seconds) using RBF kernel

Dataset	**OVO**	**OVA**	**DAG**	**UDT**
iris	0.05	0.05	0.43	0.36
wine	0.05	0.11	0.71	0.49
glass	0.48	0.13	1.47	1.36
vowel	1.26	0.32	7.35	5.45
vehicle	0.20	0.19	3.97	3.46
segment	1.29	0.57	19.34	14.60
letter	176.14	558.14	4392.56	2618.00

TABLE 5.7: A comparison of protein dataset using linear kernel

	OVO		**OVA**		**DAG**		**UDT**
Dataset	C	rate	C	rate	C	rate	rate
(C)	2^{3}	45.7	2^{-1}	34.8	2^{-5}	22.9	37.1
(S)	2^{2}	43.5	2^{-10}	22.5	2^{-6}	23.0	36.4
(H)	2^{-10}	9.3	2^{-10}	15.6	2^{-4}	16.0	27.8
(CSH)	2^{-2}	54.4	2^{-9}	49.9	2^{-6}	28.3	50.7

TABLE 5.8: A comparison of protein dataset using RBF kernel

	OVO		**OVA**		**DAG**		**UDT**
Dataset	(C,γ)	rate	(C,γ)	rate	(C,γ)	rate	rate
(C)	$(2^2, 2^{-2})$	51.7	$(2^{-4}, 2^{-2})$	50.1	$(2^1, 2^{-2})$	45.7	48.8
(S)	$(2^4, 2^{-2})$	47.6	$(2^{-10}, 2^{-2})$	29.3	$(2^1, 2^{-2})$	40.1	44.7
(H)	$(2^2, 2^{-1})$	39.7	$(2^{-10}, 2^{-2})$	18.9	$(2^1, 2^{-2})$	31.2	38.2
(CSH)	$(2^2, 2^{-1})$	54.4	$(2^{-10}, 2^{-2})$	49.9	$(2^1, 2^{-2})$	28.3	50.7

that UDT is faster in testing compared to DAG, while maintaining accuracy comparable to those standard techniques. UDT involves fewer classifiers than OVO, OVA and DAG-based SVMs. Also, we observe that in most cases, the overall accuracy produced increases when using the RBF kernel.

Experiment 2: Shows that the composed `C`, `S` and `H` parameters of the SCOP subset database improved predictive accuracy. The performance accuracies achieved by our experiments for each dataset and their composition show that UDT is better than DAG in both cases of linear and RBF kernel.

Effect of unbalanced data: Unfavourable classification results of OVA and UDT are due to the unbalanced data when separating a target class from the rest. One of the main reasons for the decreased performance is that the decision boundary between one 'true' class (typically +1) and its complementary combined 'others' class (typically -1) cannot be drawn precisely due to the majority of data points in the 'others' class. We tested this effect by considering the classes 2 and 4 from the Peterson and Barney (1952)'s vowel dataset (see Figure 5.1) and reducing the number of data points in class 2. Corresponding class boundaries are presented in Figure 5.7. The performance of the classifier at decision nodes could be improved by addressing the imbalance between classes in a random under-sampling or over-sampling manner.

5.4 Experiments on multi-class object recognition

In this section we perform experiments on multi-class visual object recognition. The multi-class classification performed with the Xerox7 dataset was carried out with 10-fold cross-validation by splitting each of the seven classes of the dataset into 10-folds. In each cross-validation SIFT features were extracted from the 9-folds of the dataset. The RAC approach that we demonstrated in the previous chapter under section 4.3, was applied on a randomly selected 15000 interest points from each class that yield $105000 \times \mathbb{R}^{128}$ visual keypoints to construct a global codebook. We report the confusion matrix and the overall error rate to evaluate our multi-class classification. Finally, we extracted the features from the union of the training and validation ('trainval') images

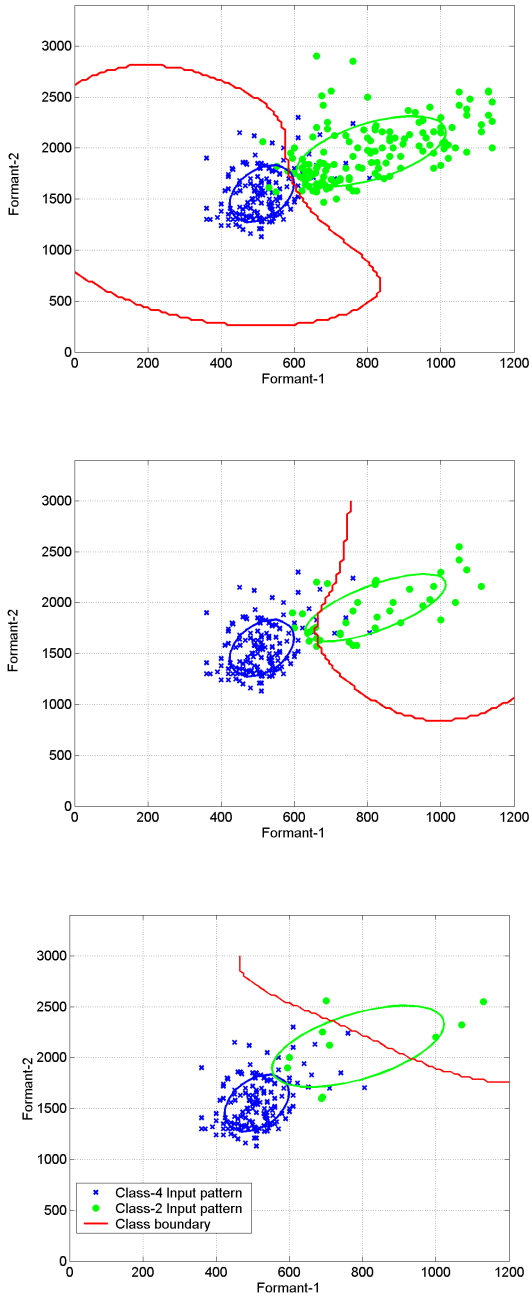


FIGURE 5.7: Class 2 vs 4 (top) balanced dataset (150:150) (middle) slightly balanced dataset (50:150) (bottom) imbalanced dataset (10:150)

of the PASCAL VOC Challenge 2007 dataset to construct a codebook using the RAC or K-means method. The features were extracted within the provided bounding box information by ignoring objects marked as 'difficult' for the construction of a codebook.

5.4.1 Testing results

In Table 5.9, we present the average classification rate of the multi-class classification evaluated with 10-fold cross validation along with the testing times (in seconds). The best results in our experiments with Xerox7 dataset were obtained by using the one-versus-all SVM classifier. Therefore, we report the confusion matrix of the 10-fold cross validation performed on the Xerox7 dataset as shown in Table 5.10 for the case of OVA based linear SVMs.

TABLE 5.9: A comparison of Xerox7 dataset using linear kernel

Xerox-7	**OVO**	**OVA**	**DAG**	**UDT**
classification rate	86.19	86.36	85.43	85.45
testing time (in sec)	22.08	21.95	84.99	82.64

TABLE 5.10: Recognition performance on the Xerox7 dataset using OVA-based linear SVMs. These results are very similar to (or slightly better than) Willamowski et al. (2004) but achieved in a tiny fraction of computation time (see section 4.3.9.2).

True classes →	Bikes	Books	Buildings	Cars	Faces	Phones	Trees
Bikes	**93.33**	0.83	0	2.5	0	1.67	1.67
Books	0.71	**72.86**	7.14	5.0	6.43	7.14	0.71
Buildings	2.0	6.0	**64.67**	2.67	13.33	4.0	7.33
Cars	1.0	3.0	3.0	**65.5**	23.0	4.50	0
Faces	0.13	0.38	0.25	0.25	**98.48**	0.25	0.25
Phones	0.95	2.38	1.90	5.24	1.43	**86.67**	1.43
Trees	2.0	0	2.67	0	16.67	0	**78.67**

In Table 5.12, we present the confusion matrix as the classification rate of the multi-class classification evaluated on the PASCAL VOC Challenge 2007 dataset using RAC technique. In Table 5.13, the reported confusion matrix is of K-means method. OVA-based linear SVMs were used in both experiments.

TABLE 5.11: A comparison of the PASCAL07 dataset using linear kernel

PASCAL07	**OVO**	**OVA**	**DAG**	**UDT**
classification rate	29.09	26.12	29.08	28.57
testing time (in sec)	1523.65	348.69	46498.68	23626.29

TABLE 5.12: Confusion matrix for the PASCAL VOC 2007 dataset based on RAC method. RAC with r=0.8. The classification was performed with OVA-based linear SVMs.

True classes →	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	Total
1	**83**	1	6	5	0	1	45	16	0	2	1	0	2	1	15	3	5	0	9	9	204
2	7	**41**	8	2	0	1	38	1	1	3	6	8	5	9	66	6	12	1	21	3	239
3	19	9	**13**	5	0	2	28	27	1	4	2	18	4	16	93	11	22	1	6	1	282
4	19	11	13	**5**	1	3	49	5	1	0	1	12	4	5	8	1	7	0	25	2	172
5	13	3	1	3	**0**	7	11	3	1	0	1	3	4	4	139	2	1	0	11	5	212
6	12	5	3	5	1	**22**	32	0	5	0	2	3	3	7	28	3	0	1	41	1	174
7	43	13	16	9	4	27	**273**	12	6	6	15	10	16	42	91	24	11	7	85	11	721
8	6	4	10	3	0	2	16	**83**	0	5	1	11	3	4	140	3	8	6	11	6	322
9	23	7	7	9	1	19	34	6	**10**	0	8	2	7	6	201	6	4	15	39	13	417
10	8	1	3	2	0	0	7	9	2	**13**	1	8	6	1	23	9	28	0	6	0	127
11	8	1	1	3	0	7	13	1	1	0	**3**	1	2	1	124	3	1	2	16	2	190
12	17	7	7	7	0	2	19	43	4	8	6	**37**	8	11	185	8	33	2	12	2	418
13	13	6	10	1	0	3	7	9	1	9	9	12	**26**	21	83	18	33	0	10	3	274
14	6	8	7	1	0	4	54	3	3	5	7	11	6	**22**	53	6	5	1	20	0	222
15	80	39	37	21	4	32	193	43	13	30	29	52	54	81	**1043**	40	60	10	129	17	2007
16	10	4	7	3	0	4	19	0	10	4	2	1	5	12	94	**16**	0	3	26	4	224
17	3	2	3	1	1	0	5	10	0	2	0	9	6	4	13	4	**31**	1	2	0	97
18	19	5	3	5	2	5	24	4	3	0	6	3	4	0	105	3	1	**6**	14	11	223
19	14	14	4	10	1	10	26	9	4	6	3	10	5	2	32	7	13	4	**84**	1	259
20	20	3	2	4	1	8	31	2	7	0	4	2	4	3	86	3	0	11	17	**21**	229

TABLE 5.13: Confusion matrix for the PASCAL VOC 2007 dataset based on K-means method. K-means with K=1000. The classification was performed with OVA-based linear SVMs.

True classes →	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	Total
1	**92**	5	2	3	0	2	35	14	1	0	3	3	6	0	8	1	2	3	15	9	204
2	8	**57**	6	4	0	2	33	2	1	1	4	9	14	18	34	8	6	0	31	1	239
3	20	10	**32**	5	2	3	16	26	1	2	7	22	12	18	57	18	14	6	9	2	282
4	18	7	20	**16**	1	3	32	4	1	0	4	0	6	9	8	5	6	5	24	3	172
5	9	6	6	3	**2**	8	10	3	0	0	3	5	3	6	125	3	0	3	9	8	212
6	17	12	1	6	0	**17**	22	1	0	1	5	1	4	8	17	2	0	3	54	3	174
7	44	15	9	15	0	13	**282**	8	1	4	13	17	27	55	69	15	1	3	119	11	721
8	15	3	5	2	0	3	7	**91**	4	6	6	20	6	6	103	5	8	13	9	10	322
9	15	8	1	8	3	26	34	3	**11**	2	8	3	11	14	164	10	2	13	35	46	417
10	10	2	5	2	0	0	5	4	0	**12**	0	14	6	8	17	9	23	1	8	1	127
11	6	2	2	4	1	10	18	0	3	0	**6**	0	4	2	108	1	0	3	10	10	190
12	10	3	16	8	1	2	22	40	2	8	5	**57**	19	9	151	17	27	5	13	3	418
13	15	5	6	7	2	2	4	6	2	8	10	22	**47**	23	53	21	19	3	19	0	274
14	8	9	1	2	2	3	48	3	0	1	12	8	7	**38**	43	9	5	3	20	0	222
15	80	44	39	27	5	33	188	43	8	16	39	67	91	121	**909**	41	34	19	170	33	2007
16	10	9	5	8	0	7	19	0	5	0	4	5	8	13	65	**19**	0	10	21	16	224
17	3	1	5	2	0	0	4	11	0	2	1	13	4	2	9	4	**30**	0	5	1	97
18	13	1	4	2	1	5	15	1	5	0	4	2	3	4	98	4	1	**14**	17	29	223
19	14	16	7	6	0	8	25	3	4	5	1	7	7	11	18	9	8	1	**109**	0	259
20	17	7	1	5	0	10	26	1	3	1	5	1	3	3	67	4	1	9	18	**47**	229

5.4.2 Discussion

Our overall error rate of the multi-class classification performed with Xerox7 dataset is 13.64% whereas the K-means method of (Willamowski et al., 2004) has an error rate of 15%. Also the error rate of faces in the Xerox7 dataset when we applied our approach is 1.52% whereas (Willamowski et al., 2004) has an error rate of 2%. UDT when applied on this task, performs very similarly in classification rate to DAG with a reduced testing time.

Furthermore, RAC based overall multi-class classification rate on the PASCAL VOC Challenge 2007 dataset is 26.12%, whereas K-means based rate is 26.92%. Even though the overall rate is disappointingly low, RAC is still comparable in performance to the K-means method. The error rate of person in the PASCAL07 dataset when we applied our approach is approximately 48% whereas K-means has an approximate error rate of 55%, which shows a significantly better performance by RAC on the dominating class *person*. UDT slightly performs better than OVA, but it is comparable to DAG and OVO-based classification. In Table 5.11, it can be noticed that UDT performs the testing step in half of the time taken by the DAG-based method.

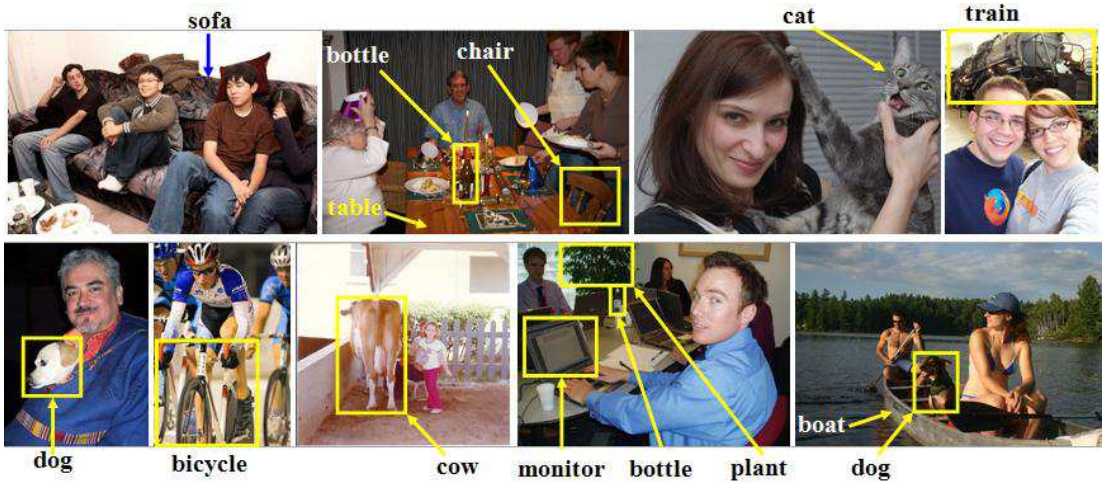


FIGURE 5.8: Presence of the highly dominating class 'person' in every other objects of the PASCAL VOC 2007 dataset

In the case of PASCAL07 multi-class experiments, most of the false positives are around the highly dominating classes: person and car. For example, Figure 5.8 shows the presence of the highly dominating class 'person' with every other objects in the PASCAL07 dataset which makes the discrimination of other classes more difficult. Moreover, the large number of SIFT descriptors that are from these dominating classes (i.e. person and car) can be clearly seen in Figure 5.9.

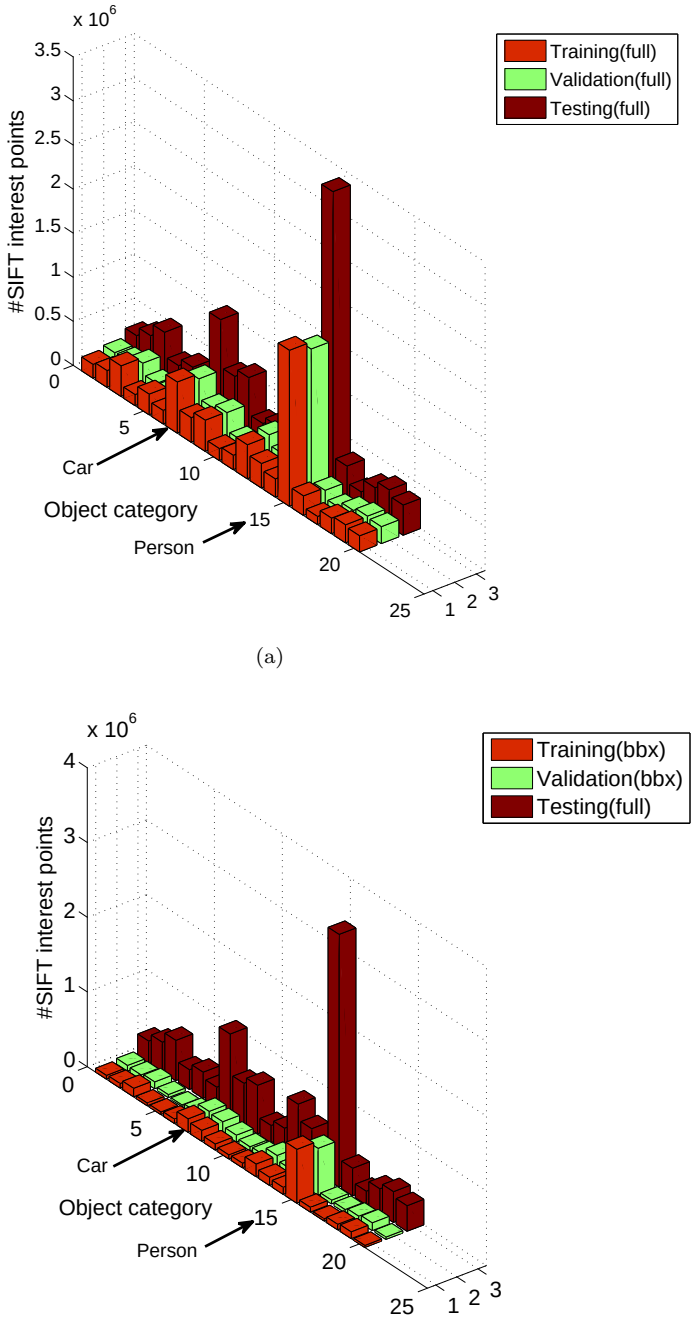


FIGURE 5.9: The number of SIFT descriptors detected in the PASCAL VOC Challenge 2007 dataset. (a) the distribution of descriptors found on the entire dataset (b) the distribution of descriptors extracted within the provided bounding box (bbx) information by ignoring objects marked as 'difficult' for the training and validation sets.

In order to check these false predictions, we looked at the feature space of the union of training and validation ('trainval') images prior to the construction of a codebook. The features were extracted only using the provided bounding box information. We computed the mean of each object categories and plotted the pair-wise correlation between

them. Figure 5.10 shows that the class 15 ('person') and class 7 ('car') are overlapping on other classes which hinders the discriminative power of codebook model based classification. Apart from the presence of multi object categories in a single image, the highly unbalanced distribution of images also makes the classification difficult, as we have discussed in section 5.3.2.

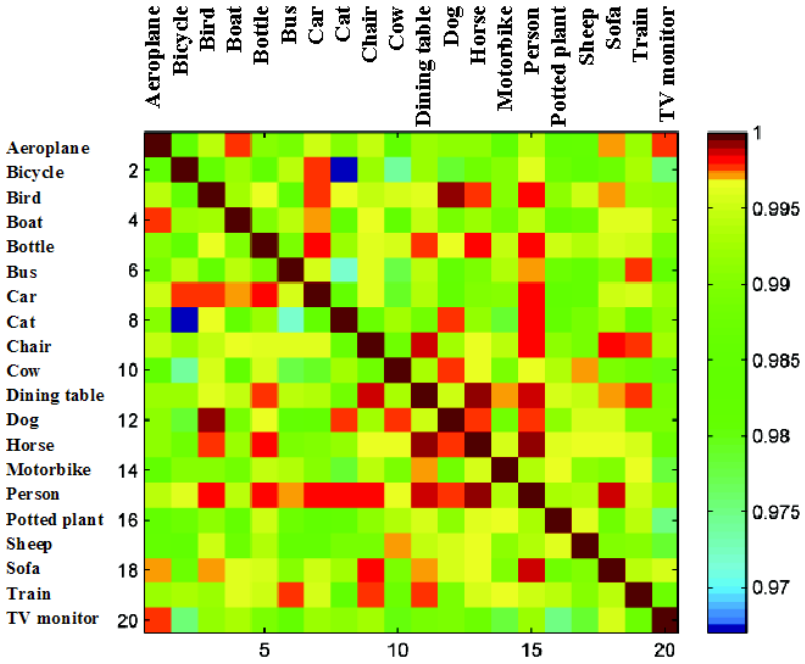


FIGURE 5.10: Pair-wise correlation of SIFT features extracted from the 'trainval' images of the PASCAL VOC 2007 dataset.

All the above mentioned experiments were carried out using scripts in Matlab embedding the SVM^{light} toolkit (Joachims, 2004) that were executed on the Iceberg[1] with the UCI and SCOP sub datasets. Iceberg is a high performance computer server that comprises a head node connected to a farm of execution nodes accessible to networked computers at the University of Sheffield. The visual object recognition tasks were executed on the LEDA3 which is a cluster computer with a dual core Xeon running at 2.6GHz and 48GB of RAM, running in the ISIS research group, University of Southampton.

5.5 Summary

SVMs were originally developed for solving binary classification problems, but binary SVMs have also been extended to solve the problem of multi-class pattern classification. When dealing with multi-classes as in visual object recognition, one needs an appropriate technique to effectively extend these binary classification methods for multi-class classification. We address this issue by demonstrating a novel architecture that we refer to UDT. The UDT is a binary decision tree arranged in a top-down manner, using

[1] http://www.shef.ac.uk/wrgrid/iceberg

the optimal margin classifier at each split to relieve the excessive time in classifying the test data when compared with the DAG-based SVMs. It can be applied to any other application in machine learning in which there are natural groupings among the patterns.

The UDT implements OVA-based concepts at each decision node with a 'knock-out' strategy. A shortcoming of the UDT-based classification is the long training time needed to find the optimal model for each decision node of the tree. This training time can be further improved by fixing the order of the classifiers for the decision tree when finding the optimal node for the root. During the training phase, when determining the optimal model at each node of the decision tree, we use one class lesser than in its previous level, except at the root node. Thus, in each training the number of training data is reduced considerably compared to OVA-based SVMs, which use all the training data. But the number of decision functions required by UDT in the worst case scenario is n-1, and in the best case scenario is one, whereas OVA in any of the cases requires n decision functions. OVO and DAG requires $n(n-1)/2$ decision functions, where n is the number of classes.

In the context of visual object recognition, we have tested the RAC and K-means approaches on the Xerox7 and PASCAL VOC Challenge 2007 datasets. Our experimental results show that the performance of RAC is very similar to or slightly better than K-means based method, but achieved in a tiny fraction of computation time. Furthermore, to the best of our knowledge in the literature, we have first presented the confusion matrix of the multi-class classification results obtained on the widely used PASCAL07 dataset. Even though the overall classification rate is disappointingly low, it gives a baseline performance measure for other researchers to compare their own techniques.

In contrast, UDT is faster in testing compared to DAG, while maintaining accuracy comparable to those standard techniques OVA, OVO and DAG-based SVMs. The visual object recognition in this thesis was performed on image histograms that performed better with linear SVMs. Our future efforts will focus on trying histogram intersection kernel (HIK) in our framework. A description of the HIK is presented in the following chapter of this thesis.

Chapter 6

Conclusions and future directions

This thesis has presented a novel approach to constructing a visual codebook that is computationally effective and discriminant, by means of a one-pass resource-allocating approach, inspired by the resource allocating network algorithms developed in the artificial neural networks literature. Comparing our approach to K-means algorithm, and a closely related mean-shift clustering technique proposed by Jurie and Triggs (2005) we show that this method achieves a recognition performance very similar to or slightly better than those techniques. RAC achieved this comparable performance in a tiny fraction of the computing time because, apart from an initial scan through a small subset to determine length scales, each data item is processed only once. Unlike density preserving clustering techniques, our method spreads out the cluster centres over a wider range of the patch-based feature space, thereby including rare features into the vocabulary. Experimental results in the context of face recognition and visual object classes classification tasks performed with SIFT features support our claims. They further demonstrate the generality of our approach and the ease of implementation that makes this method possible to run on large scale datasets.

By careful examination of the literature, we concluded that several authors make use of different combinations of interest point detectors and descriptors, different features, and various clustering methods to achieve an increased object recognition rate based on visual descriptors of modest size when constructing a visual vocabulary. The reduced size of the descriptors used in the construction of a codebook is due to the bottleneck in the traditional clustering approaches such as K-means and Gaussian mixture models, and the obtained cluster centres are those that have a high probability density which are not necessarily the most discriminative. Hence the work that we demonstrated in this thesis makes an important contribution towards improving current thinking on the subject.

In chapter 4, we also investigated a novel sequential hierarchical clustering technique with nearly the same efficiency as the traditional hierarchical clustering method. In summary,

the bag-of-words approach is a powerful technique that has been widely employed in text classification, visual object recognition, natural scene analysis, video scene analysis, bioinformatics etc. Therefore, our RAC and SHC have a wide range of applications with comparable performance at a drastically reduced computational cost. So far, this idea has been addressed by isolated works that only considered a particular application of the concept. In the following we point out some research areas where RAC can be used to reduce the computational complexity of the proposed methods.

- Recently, Chechik et al. (2009) proposed an online algorithm for scalable image similarity (OASIS) learning that captures both semantic and visual aspects of image similarity. The authors have tested their method for scalability on 2.7 million images collected from the web. Although it is proposed as an online algorithm to cope with images on the web, the codebook that they use is generated in an offline setting using K-means clustering method with K=10000. Thus, their proposed technique can be easily upgraded with the use of our RAC to make it more realistic to update the learnt visual codebook with newly seen images on the web in an online fashion.

- Ballan et al. (2009) proposed an approach to recognise events in video sequences by the bag-of-words approach. In their approach, an action is described by a phrase of variable size, depending on the clip's length, which is able to incorporate temporal relations. They then compare video phrases by computing edit distances between them. K-means was the choice of technique to construct the codebook. SIFT features were used with string kernel SVMs. It is clear that the codebook is limited to the length of a video due to the use of the traditional K-means method. Thus, RAC can be used to solve this limitation as it sequentially processes data.

- Shuiwang et al. (2009) proposed a computational method for automated annotation of *Drosophila* gene expression pattern images using the bag-of-words approach. They applied K-means clustering to generate a codebook with K = 2000, 1000, and 500 for lateral, dorsal, and ventral images, respectively. The features they used are the SIFT descriptors that are then fed to linear SVMs. Considering the number of available images that is rapidly increasing in this context, RAC can deal with this situation efficiently by updating the codebook without retraining the whole training images.

In future, while working with millions of patch-based descriptors, RAC can be used as a fast approximation to determine representative interest points in the feature space that reduce the initial size of a problem on which traditional K-means based methods can be still applied.

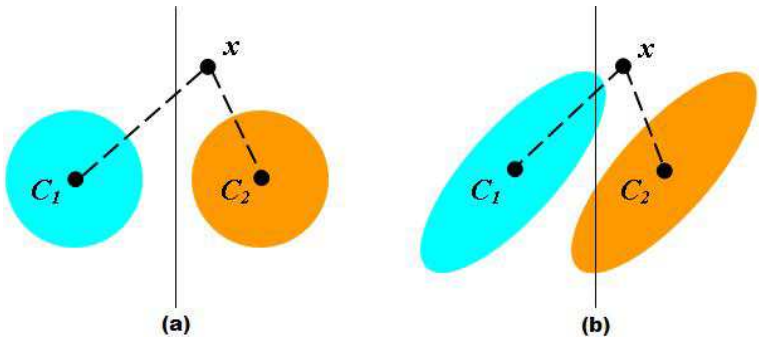


FIGURE 6.1: Distance of an observation to the mean of a distribution (a) **x** assigned to $\mathbf{C_2}$ (b) **x** assigned to $\mathbf{C_1}$

Furthermore, in future research I would be interested to explore the following with the usage of RAC.

- **Mahalanobis distance:** It would be interesting to see the usage of Mahalanobis distance when constructing category-specific codebooks using SIFT features. Sivic and Zisserman (2003) have used this distance measure to construct a codebook for retrieving visual objects and scenes from a movie. They claim that the Mahalanobis distance enables more noisy components of the SIFT features to be weighted down, and also decorrelates the components. It would not be appropriate to use the covariance matrix over the entire feature space, since it is mainly influenced by inter-class variations.

  If two classes are not spherically distributed then it is better to assign $\boldsymbol{x}$ to the class with the largest probability density in $\boldsymbol{x}$, and therefore to the class with the lower value of the Mahalanobis distance of $\boldsymbol{x}$ to the class mean. This scenario is illustrated in Figure 6.1.

- **Learning rate of RAC:** The hyper-parameter r of RAC can be better represented as $r(t)$, the length scale of the input space at the t^{th} input presentation. This can be achieved by starting the span over the input space with $r(t) = r_{max}$, which is the largest length scale of interest. The span then shrinks until it reaches r_{min} which is the smallest length scale of interest. $r(t)$ can be updated as follows:

$$r(t) = \max(\eta_t r_{max}, r_{min}) \tag{6.1}$$

  where η_t is the exponential decreasing learning rate determined by equation 6.2.

$$\eta_t = \eta_0 \left(\frac{\eta_f}{\eta_0}\right)^{\frac{t}{N}} \tag{6.2}$$

  where t is the online iteration index, N is the number of visual descriptors, η_0 and η_f are the initial and final learning rates, respectively.

It can be expected when the size of the patch-based visual descriptors increases, the number of clusters constructed by RAC will grow very slowly making the codebook more compact.

- **Histogram intersection kernel:** It is worth trying the use of the histogram intersection kernel in our framework. In general, visual codebook generation methods used in a bag-of-features model use the Euclidean (l_2) distance for comparing two histograms at the classification step. However, it has been shown that the Euclidean distance is not the most effective method of comparing two histograms (Maji et al., 2008). Instead, the histogram intersection kernel (HIK) was demonstrated to give significantly better classification performance.

  Suppose $\mathbf{h} = (h_1, h_2, \cdots, h_d) \in \mathbb{R}_+^d$ be a histogram. Then HIK is defined as:

$$K_H(\mathbf{h}_1, \mathbf{h}_2) = \sum_{i=1}^{d} \min(h_{1i}, h_{2i})$$

  Odone et al. (2005) show that the HIK forms a positive definite kernel that facilitates the use of HIK in SVMs. Thus there exists a mapping ϕ such that,

$$K_H(\mathbf{h}_1, \mathbf{h}_2) \equiv \phi(\mathbf{h}_1) \cdot \phi(\mathbf{h}_2)$$

  Maji et al. (2008) proposed a technique to accelerate the kernel evaluations for the case of SVMs. In principle, all the experiments performed in this thesis using the bag-of-features model followed by SVM classifiers could be improved through the use of HIK.

In chapter 4, we also presented an object recognition system whose learning is incremental with training images and the output classifier accounts for class specific discriminant features. However, another approach has been explored without the use of cluster analysis upon which the discrimination is based adaptive in size by measuring the Fisher scores between the classes of interest. Our experimental results show that the latter approach can be used effectively with a carefully chosen sub set of features that yields the optimal threshold in advance. In contrast, the Fisher based technique is less expensive in computation as it is free from cluster analysis. A pruning strategy has been employed to detect outliers during the computation of frequency histograms. Our preliminary results show the need for an outlier detection technique to improve the classification accuracy. We also tested the effective number of SIFT features required in achieving reasonable recognition performance and the robustness of RAC with SIFT features to additive noise level. Chapter 5 demonstrated a novel framework that implements OVA-based concept of SVMs to perform multi-class classification in a reduced testing time compared to the DAG-SVMs.

Based on the results presented in this thesis, we believe that the presented resource-allocating codebook methodology is effective for solving visual object recognition problems. We have shown, with extensive results that it is comparable to or outperforms more traditional approaches. We also pointed out some potential methods to be explored with this technique.

Appendix A

Support vector machines

SVMs have been developed by Vapnik (1995). SVMs are supervised learning machines based on statistical learning theory that can be used for pattern recognition and regression. SVMs are typically non-linear in the input space but linear in a higher dimensional 'feature' space. SVMs were originally developed for solving binary classification problems (Cortes and Vapnik, 1995), but binary SVMs have also been extended to solve the problem of multi-class pattern classification (Hsu and Lin, 2002) which is still an ongoing research issue. SVMs deliver state-of-the-art performance in real world applications including regression problems (Vapnik et al., 1997; Meng et al., 2005).

For the pattern classification case, SVMs have been used for scene image classification (Ren et al., 2004; Fei-Fei and Perona, 2005), visual object recognition (Winn et al., 2005; Csurka et al., 2004), pattern detection (Sahbi and Geman, 2006), isolated handwritten digit recognition (Cortes and Vapnik, 1995), and speaker identification (Salomon et al., 2002).

If we consider a set of input vectors $\boldsymbol{x}$: $x_1, x_2, \ldots, x_l$ to an output variable $\boldsymbol{y}$ representing the class labels $y_1, y_2, \ldots, y_l$, where l is the number of training points, then a classifier can be viewed as the overall system that maps from $\boldsymbol{x}$ to $\boldsymbol{y}$.

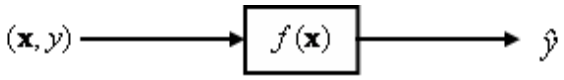


FIGURE A.1: Block diagram of a classifier. $\boldsymbol{x}$ input vectors or features, $\boldsymbol{y}$ the class labels, and f a classifier that maps $\boldsymbol{x}$ to $\boldsymbol{y}$.

The mapping is expressed in terms of a mathematical function that contains a number of adjustable parameters $\mathbf{w}$. This function can be written in the form:

$$f(\mathbf{x}) = y(\mathbf{x}, \mathbf{w}).$$

Linearly separable case

In pattern classification suppose we are given a set of l training points of the form:

$$(x_1, y_1), (x_2, y_2), ..., (x_l, y_l) \in \Re^n \times \{+1, -1\},$$

where x_i is an n-dimensional vector and y_i are their labels such that:

$$y_i = \begin{cases} +1: \text{ if the vector is classified to class } +1 \\ -1: \text{ if the vector is classified to class } -1 \end{cases}$$

we thus try to find a classification boundary function *f(x)=y* that not only correctly classifies the input patterns in the training data but also correctly classifies the unseen patterns.

The classification boundary of all values of $\boldsymbol{x}$ for which *f(x)*=0 is a *hyperplane* defined by its normal vector $\mathbf{w}$, which basically divides the input space into the class +1 vectors on one side and the class -1 vectors on other side.

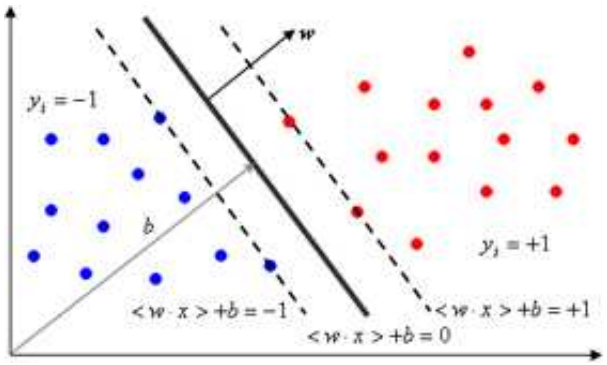


FIGURE A.2: An example of a separable problem in a 2-dimensional space.

Then there exists *f(x)* such that

$$f(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + b, \mathbf{w} \in \Re^n \textit{ and } b \in \Re, \tag{A.1}$$

subject to

$$y_i f(x_i) \geq 1 \text{ for } i = 1, 2, \ldots, l. \tag{A.2}$$

The optimal hyperplane is defined by maximizing the distance between the hyperplane and the data points closest to the hyperplane (called *support vectors*). Mathematically, the weighted sum of the support vectors is the normal vector of the hyperplane.

Then we need to maximize the margin $\gamma = \frac{2}{||\mathbf{w}||}$ or minimise $||\mathbf{w}||$ subject to constraint A.2. This is a quadratic programming (QP) optimization problem that can

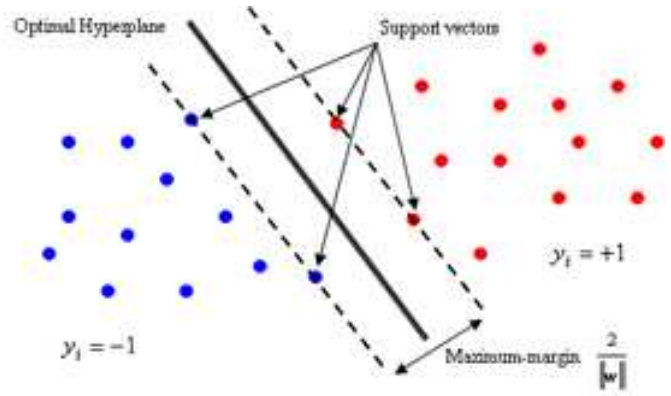


FIGURE A.3: An example of an optimal hyperplane and support vectors.

be expressed as:

$$\min_{w,b} \frac{1}{2}\|\mathbf{w}\|^2 \tag{A.3}$$

The optimisation problem of Equation A.3 can be solved by finding the saddle point of the Lagrangian,

$$L(w,b,\alpha) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{l} \alpha_i (y_i[< \mathbf{w}, \mathbf{x}_i > +b] - 1) \tag{A.4}$$

where α are the Lagrange multipliers. *L(w,b,α)* has to be minimised w.r.t $\mathbf{w}$, b and maximised w.r.t $\alpha \geq 0$.

By setting *L(w,b,α)*= 0, we have, $\mathbf{w} - \sum_{i=1}^{l} \alpha_i y_i \mathbf{x}_i = 0 \quad \Rightarrow \quad \mathbf{w} = \sum_{i=1}^{l} \alpha_i y_i \mathbf{x}_i$ and $\sum_{i=1}^{l} \alpha_i y_i = 0$

The dimension of input space can be replaced by the number of input patterns by representing the SVM problem in the dual form (Cortes and Vapnik, 1995). That is, the objective function in the dual form will be only in terms of α_i.

If we substitute $\mathbf{w}= \sum_{i=1}^{l} \alpha_i y_i \mathbf{x}_i$ to *L(w,b,α)*, we have

$$L(w,b,\alpha) = \frac{1}{2}\sum_{i=1}^{l} \alpha_i y_i \mathbf{x}_i^T \sum_{j=1}^{l} \alpha_j y_j \mathbf{x}_j + \sum_{i=1}^{l} \alpha_i (1 - y_i (\sum_{j=1}^{l} \alpha_j y_j \mathbf{x}_j^T \mathbf{x}_i + b))$$

$$= \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j + \sum_{i=1}^{l} \alpha_i - \sum_{i=1}^{l} \alpha_i y_i \sum_{j=1}^{l} \alpha_j y_j \mathbf{x}_j^T \mathbf{x}_i - b\sum_{i=1}^{l} \alpha_i y_i$$

$$= -\frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j + \sum_{i=1}^{l} \alpha_i$$

This is a function of α_i only.

The dual problem is a QP problem as follows:

$$\max. \quad W(\alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i=1,j=1}^{l} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$$

subject to

$$\alpha_i \geq 0, \quad \text{and} \quad \sum_{i=1}^{l} \alpha_i y_i = 0$$

Non-separable case

In practice, data sets are often not linearly separable in the input space. To deal with this situation slack variables (ξ_i) are introduced (Cortes and Vapnik, 1995) into A.3, where ξ_i are a measure of the misclassification errors. The constraints of Equation A.1 in A.2 now becomes:

$$y_i f(x_i) \geq 1 - \xi_i \quad \text{for } i = 1, 2, \ldots, l. \tag{A.5}$$

where $\xi_i \geq 0$. Thus, the QP optimisation problem in Equation A.3 becomes,

$$\min_{w,b} \frac{1}{2} \|\mathbf{w}\|^{\mathbf{2}} + \mathbf{C} \sum_{\mathbf{i}}^{\mathbf{l}} \xi_{\mathbf{i}} \tag{A.6}$$

where C is the parameter that determines the tradeoff between the maximization of the margin and minimization of the classification error.

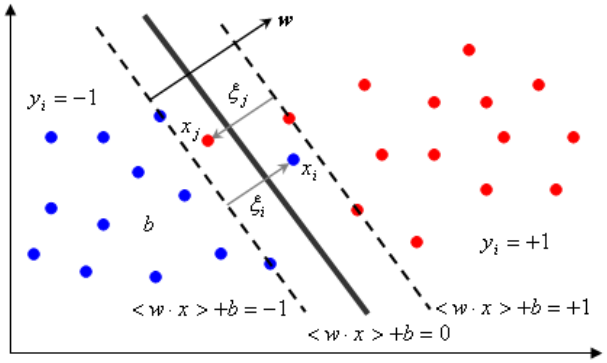


FIGURE A.4: Misclassified samples in classification.

The solution to the optimisation problem of Equation A.6 can be solved by finding the saddle point of the Lagrangian,

$$L(w, b, \alpha, \xi, \beta) = \frac{1}{2} \|\mathbf{w}\|^2 - C \sum_{i=1}^{l} \xi_i - \sum_{i=1}^{l} \alpha_i (y_i [< \mathbf{w}, \mathbf{x}_i > + b] - 1 + \xi_i) + \sum_{i=1}^{l} \beta_i \xi_i \tag{A.7}$$

where α, β are the Lagrange multipliers. *L(w,b,*α, ξ, β*)* has to be minimised w.r.t $\mathbf{w}$, b, $\boldsymbol{x}$ and maximised w.r.t α, β. The solution to the above optimization problem has the form:

$$f(\mathbf{x}) = \mathbf{w} \cdot \phi(\mathbf{x}) + b = \sum_{i=1}^{l} c_i \phi(x_i) \cdot \mathbf{w} + b \tag{A.8}$$

where $\phi(.)$ is the mapping function that transforms the vectors in input space to feature space.

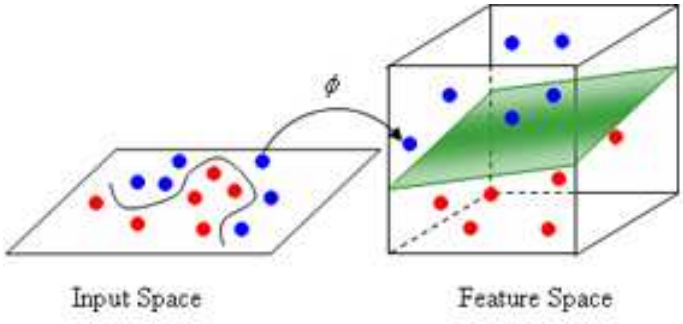


FIGURE A.5: An example of transforming the data from input space to feature space.

The dot product in A.8 can be computed without explicitly mapping the points into feature space by using a kernel function (some example kernels are presented in Table A.1, where γ, r and d are kernel parameters), that can be defined as the dot product of two points in the feature space:

$$K(x_i, x_j) + b \equiv \phi(x_i) \cdot \phi(x_j) \tag{A.9}$$

Thus the solution to the optimization problem has the form:

$$f(\mathbf{x}) = \sum_{i=1}^{l} c_i \cdot K(x_i, x_j) + b \tag{A.10}$$

where most of the coefficients c_i are zero except for the coefficients of support vectors.

The dual problem of this new constrained optimisation problem with kernel function becomes,

$$\max. \quad W(\alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i=1,j=1}^{l} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)$$

subject to

$$C \geq \alpha_i \geq 0, \quad \text{and} \quad \sum_{i=1}^{l} \alpha_i y_i = 0$$

Generally, $K(x_i,x_j)$ satisfies the Mercer's theorem (Cortes and Vapnik, 1995). Mercer's theorem guarantees the dot product in the equation can be substituted by kernel function. Unfortunately the Sigmoid kernel does not satisfy the Mercer condition on all γ and r (Lin and Lin, 2003).

Kernel methods were introduced in 1990s with SVMs and it provides a general purpose toolkit for pattern analysis. Kernel algorithms can perform linear regression in much higher dimensional spaces efficiently by using the kernel trick. It can also work in infinite dimensional spaces, e.g. using the Gaussian kernel (Shawe-Taylor and Cristianini, 2004).

TABLE A.1: Examples of kernels employed by SVMs

Linear kernel	$K(\mathbf{x},\mathbf{y}) = \mathbf{x}^T \cdot \mathbf{y}$
Polynomial kernel	$K(\mathbf{x},\mathbf{y}) = (\gamma \mathbf{x}^T \cdot \mathbf{y} + r)^d, \gamma > 0$
Radial Basis Function kernel	$K(\mathbf{x},\mathbf{y}) = \exp(-\gamma \|\mathbf{x} - \mathbf{y}\|)^2, \gamma > 0$
Sigmoid kernel	$K(\mathbf{x},\mathbf{y}) = \tanh(\gamma \mathbf{x}^T \cdot \mathbf{y} + r)$, $\gamma > 0$ and $r < 0$

Having the generalised optimal hyperplane, the expectation value of the probability of committing an error on a test example is as follows:

$$E[P(\text{error})] \leq \frac{E[\text{number of support vectors}]}{(\text{number of training vectors}) - 1} \tag{A.11}$$

Bibliography

E. Achtert, C. Bohm, H-P. Kriegel, and P. Kröger. Online Hierarchical Clustering in a Data Warehouse Environment Data Mining. In *Proceedings of the Fifth IEEE International Conference on Data Mining*, pages 10–17, 2005.

A. Agarwal and B. Triggs. Hyperfeatures - Multilevel Local Coding for Visual Recognition. In *Proceedings of the European Conference on Computer Vision (ECCV'06)*, pages 30–43. Springer, 2006.

S. Agarwal, A. Awan, and D. Roth. Learning to Detect Objects in Images via a Sparse, Part-Based Representation. In *IEEE Transactions on Pattern Analysis and Machince Intelligence*, volume 26, pages 1475–1490, 2004.

L. Ballan, M. Bertini, A. D. Bimbo, and G. Serra. Video Event Classification using Bag-of-Words and String Kernels. In *Proceedings of the Image Analysis and Processing (ICIAP'09)*, volume 5716, pages 170–178, 2009.

H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool. SURF: Speeded Up Robust Features. In *Computer Vision and Image Understanding*, volume 110, pages 346–359, 2008.

G. Blanchard and D. Geman. Hierarchical Testing Designs for Pattern Recognition. volume 33, pages 1155–1202. The Analysis of Statistics, 2005.

D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, (3):993–1022, 2003.

J. Brank, M. Grobelnik, N. Milic-Frayling, and D. Mladenic. Interaction of Feature Selection Methods and Linear Classification Models. In *Proceedings of the ICML-02 Workshop on Text Learning*. Forthcoming, 2002.

K. M. Branson and S. Agarwal. Structured Principal Component Analysis. In *UCSD Technical Report*, 2003.

L. Breiman. Random Forests. *Machine Learning*, 45(1):5–32, 2001.

M. J. Brusco and H. F. Köhn. Comment on "Clustering by Passing Messages between Data Points". *Science*, 319(5864), 2008. ISSN 1095-9203.

C. J. Burgues. A Tutorial on Support Vector Machines for Pattern Recognition. volume 2, pages 121–167. Knowledge Discovery and Data Mining, 1998.

J. Canny. A Computational Approach to Edge Detection. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 8, pages 544–567, 1986.

G. Chechik, V. Sharma, U. Shalit, and S. Bengio. An Online Algorithm for Large Scale Image Similarity Learning. In *Advances in Neural Information Processing Systems (NIPS'09)*, 2009.

J. Chen, T. N. Pappas, A. Mojsilovic, and B. E. Rogowitz. Adaptive Perceptual Color-Texture Image Segmentation. In *IEEE Transactions on Image Processing*, volume 14, pages 1524–1536, 2005.

Y. Cheng. Mean Shift, Mode Seeking, and Clustering. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 17, pages 790–799, 1995.

D. Comaniciu and P. Meer. Mean Shift: A Robust Approach toward Feature Space Analysis. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 24, pages 603–619, 2002.

C. Cortes and V. N. Vapnik. *Support-Vector Networks*, volume 20. Machine Learning, 1995. ISBN 20:273-297.

G. Csurka, C. R. Dance, L. Fan, J. Willamowski, and C. Bray. Visual Categorization with Bags of Keypoints. In *Workshop on Statistical Learning in Computer Vision, ECCV'04*, pages 1–22, 2004.

N. Dalal and B. Triggs. Histograms of Oriented Gradients for Human Detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'05)*, pages 886–893, 2005.

J. Davis and M. Goadrich. The Relationship between Precision-Recall and ROC Curves. In *Proceedings of the 23rd International Conference on Machine Learning (ICML'06)*, pages 233–240, New York, NY, USA, 2006. ACM Press.

R. Debnath, N. Takahide, and H. Takahashi. A Decision based One-Against-One Method for Multi-class Support Vector Machine. volume 7, pages 164–175. Pattern Analysis Application, 2004.

H. Deng, W. Zhang, E. Mortensen, T. Dietterich, and L. Shapiro. Principal Curvature-based Region Detector for Object Recognition. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR'07)*, pages 1–8, 2007.

C. H. Ding and I. Dubchak. Multi-class Protein Fold Recognition using Support Vector Machines and Neural Networks. In *Bioinformatics*, volume 17, pages 349–358, 2001.

G. Dorko, C. Schmid, and P. Lear. Object Class Recognition using Discriminative Local Features. Technical report, RR-5497, INRIA -Rhone-Alpes, 2005.

R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification (2nd Edition)*. Wiley-Interscience, November 2000. ISBN 0471056693.

M. B. Eisen, P. T. Spellman, P. O. Brown, and D. Botstein. Cluster Analysis and Display of Genome-wide Expression Patterns. In *Proceedings of the National Academy of Sciences USA*, volume 95, pages 14863–14868, 1998.

Y. El-Sonbaty and M. A. Ismail. On-line Hierarchical Clustering. In *Pattern Recognition Letters*, volume 19, pages 1285–1291, 1998.

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 2009 (VOC2009) Results. http://www.pascal-network.org/challenges/VOC/voc2009/workshop/index.html, 2009.

J. D. R. Farquhar, S. Szedmak, H. Meng, and J. Shawe-Taylor. Improving "Bag-of-Keypoints" Image Categorisation: Generative Models and PDF-Kernels. In *LAVA Report, University of Southampton, UK.*, 2005.

B. Farran and C. Saunders. Voted Spheres: An Online Fast Approach to Large Scale Learning. In *IEEE International Conference on Advanced Information Networking and Applications Workshop*, pages 744–749, 2009.

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 IEEE Workshop on Generative-Model based with Vision with CVPR'04*, 2004.

L. Fei-Fei and P. Perona. A Bayesian Hierarchical Model for Learning Natural Scene Categories. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 2, pages 524–531, 2005.

B. J. Frey and D. Dueck. Clustering by Passing Messages between Data Points. In *Science AAAS*, volume 315, pages 972–976, 2007.

K. Fukunaga and L. D. Hostler. The Estimation of the Gradient of a Density Function, with Application in Pattern Recognition. In *IEEE Transactions on Information Theory*, volume 21, pages 32–40, 1975.

A. S. Georghiades, P. N. Belhumeur, and D. J. Kriegman. From Few to Many: Illumination Cone Models for Face Recognition under Variable Lighting and Pose. In *Pattern Analysis and Machine Intelligence*, volume 23, pages 643–660, 2001.

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.

M. Girolami and M. Zhong. Data Integration for Classification Problems Employing Gaussian Process Priors. In *Twentieth Annual Conference on Neural Information Processing Systems (NIPS'07)*, pages 465–472. MIT Press, 2007.

K. Grauman and T. Darrell. The Pyramid Match Kernel: Discriminative Classification with sets of Image Features. In *Proceedings of the Tenth IEEE International Conference on Computer Vision (ICCV'05)*, volume 2, pages 458–1465, 2005.

I. Guyon, Ulrike von Luxburg, and R. C. Williamson. Clustering: Science or Art? In *Neural Information Processing Systems (NIPS'09) Workshop on Clustering Theory*, pages 179–186, 2009.

C. Harris and M. Stephens. A Combined Corner and Edge Detector. In *Proceedings of Alvey Vision Conference*, pages 147–151, 1988.

J. A. Hartigan and M. A. Wong. A K-Means Clustering Algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 28(1):100–108, 1979.

M. Hasan and J. Jue. Online Clustering for Hierarchical WDM Networks. In *IEEE/OSA Conference on Optical Fiber Communication*, pages 1–3, 2008.

C.-W. Hsu and C.-J. Lin. A Comparison of Methods for Multi-class Support Vector Machines. In *IEEE Transactions on Neural Networks*, volume 13, pages 415–425, 2002.

A. K. Jain, M. N. Murty, and P. J. Flynn. Data Clustering: A Review. In *ACM Computing Surveys*, volume 31, pages 264–323, 1999.

T. Joachims. Text Categorization with Support Vector Machines: Learning with Many Relevant Features. In *Proceedings of the European Conference on Machine Learning (ECML'98)*, pages 137–142, 1998.

T. Joachims. SVMlight - a Software Package for Support Vector Learning. 2004. http://svmlight.joachims.org/.

L. Juan and O. Gwun. A Comparison of SIFT, PCA-SIFT and SURF. In *International Journal of Image Processing*, volume 3, pages 143–152, 2009.

F. Jurie and B. Triggs. Creating Efficient Codebooks for Visual Recognition. In *Proceedings of the Tenth IEEE International Conference on Computer Vision (ICCV'05)*, volume 1, pages 604–610, 2005.

T. Kadir and M. Brady. Saliency, Scale and Image Description. *International Journal of Computer Vision*, 45(2):83–105, 2001.

V. Kadirkamanathan and M. Niranjan. A Function Estimation Approach to Sequential Learning with Neural Networks. In *Neural Computation*, volume 5, pages 954–975, 1993.

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 (CVPR'04)*, pages 511–517, 2004.

S. Kim and I.-S. Kweon. Biologically Motivated Perceptual Feature: Generalized Robust Invariant Feature. In *Proceedings of the Asian Conference on Computer Vision (ACCV'06)*, pages 305–314, 2006.

S. Kim and I.-S. Kweon. Object Categorization Robust to Surface Markings using Entropy-guided Codebook. In *IEEE Workshop on Applications of Computer Vision*, page 22, 2007.

N. Larios, H. Deng, W. Zhang, M. Sarpola, J. Yuen, R. Paasch, A. Moldenke, D. A. Lytle, S. R. Correa, E. N. Mortensen, L. Shapiro, and T. Dietterich. Automated Insect Identification through Concatenated Histograms of Local Appearance Features. *Machine Vision and Applications*, 19(2):105–123, 2007.

D. Larlus and F. Jurie. Latent Mixture Vocabularies for Object Categorization. In *Proceedings of the British Machine Vision Conference (BMVC'06)*, pages 959–968, 2006.

S. Lazebnik, C. Schmid, and J. Ponce. A Maximum Entropy Framework for Part-based Texture and Object Recognition. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV'05)*, volume 1, pages 832–838, 2005a.

S. Lazebnik, C. Schmid, and J. Ponce. A Sparse Texture Representation using Local Affine Regions. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 27, pages 1265–1278, 2005b.

S. Lazebnik, C. Schmid, and J. Ponce. Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'06)*, pages 2169–2178, 2006.

P. Legendre and L. Lengendre. *Numerical Ecology.* Elsevier Science B.V., 2003.

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, 2008.

B. Leibe and B. Schiele. Interleaved Object Categorization and Segmentation. In *Proceedings of the British Machine Vision Conference (BMVC'03)*, pages 759–768, 2003.

T. Li, T. Mei, and I.-S. Kweon. Learning Optimal Compact Codebook for Efficient Object Categorization. In *IEEE Workshop on Applications of Computer Vision*, pages 1–6, 2008.

H.-T. Lin and C.-J. Lin. A Study on Sigmoid Kernels for SVM and the Training of non-PSD Kernels by SMO-type Methods. Technical report, National Taiwan University, 2003.

L. Lo Conte, B. Ailey, T. J. Hubbard, S. E. Brenner, A. G. Murzin, and C. Chothia. SCOP: a Structural Classification Of Proteins Database. *Nucleic Acids Research*, 28 (1):257–259, January 2000.

Y. Loewenstein, E. Portugaly, M. Fromer, and M. Linial. Efficient Algorithms for Accurate Hierarchical Clustering of Huge Datasets: Tackling the Entire Protein Space. In *Bioinformatics*, volume 24, pages 41–49, 2008.

D. Lowe. Object Recognition from Local Scale-Invariant Features. In *The Proceedings of the Seventh IEEE International Conference on Computer Vision (ICCV'99)*, volume 2, pages 1150–1157, 1999.

D. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. In *International Journal of Computer Vision*, volume 60, pages 91–110, 2004.

J. Luo, Y. Ma, E. Takikawa, S. Lao, M. Kawade, and B-L. Lu. Person-specific SIFT Features for Face Recognition. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, volume 2, pages 593–596, 2007.

S. Maji, A. C. Berg, and J. Malik. Classification using Intersection Kernel Support Vector Machines is Efficient. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'08)*, pages 1 – 8, 2008.

G. Martínez-Muñoz, W. Zhang, N. Payet, S. Todorovic, N. Larios, A. Yamamuro, D. Lytle, A. Moldenke, E. Mortensen, R. Paasch, L. Shapiro, and T. G. & Dietterich. Dictionary-free Categorization of Very Similar Objects via Stacked Evidence Trees. In *Proceedings of the IEEE Conference in Computer Vision and Pattern Recognition (CVPR'09)*, 2009.

J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide Baseline Stereo from Maximally Stable Extremal Regions. In *British Machine Vision Conference (BMVC'02)*, volume 1, pages 384–393, 2002.

H. Meng, J. Shawe-Taylor, S. Szedmak, and J. D. R. Farquhar. Support Vector Machines to Synthesise Kernels. pages 242–255. Springer-Verlag Berlin Heidelberg, 2005.

A. Meyerson, L. O'Callaghan, and S. Plotkin. A K-Median Algorithm with Running Time Independent of Data Size. *Machine Learning*, 56:61–87, 2004.

K. Mikolajczyk, B. Leibe, and B. Schiele. Multiple Object Class Detection with a Generative Model. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 1, pages 26–36, 2006.

K. Mikolajczyk and C. Schmid. Indexing based on Scale Invariant Interest Points. In *Proceedings of the 8th IEEE International Conference on Computer Vision (ICCV'01)*, pages 525–531, 2001.

K. Mikolajczyk and C. Schmid. An Affine Invariant Interest Point Detector. In *Proceedings of the European Conference on Computer Vision (ECCV'02)*, pages 128–142. Springer Verlag, 2002.

K. Mikolajczyk and C. Schmid. A Performance Evaluation of Local Descriptors. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 27, pages 1615–1630, 2005.

F. Moosmann, B. Triggs, and F. Jurie. Fast Discriminative Visual Codebooks using Randomized Clustering Forests. In *Neural Information Processing Systems (NIPS'07)*, pages 985–992, 2007.

D. J. Newman, S. Hettich, C. L. Blake, and C. J. Merz. UCI Repository of Machine Learning Database. http://www.ics.uci.edu/ mlearn/MLRepository.html, 1998.

D. Nister and H. Stewenius. Scalable Recognition with a Vocabulary Tree. In *IEEE Computer Society Conference on Computer*, volume 2, pages 2161–2168, 2006.

M. S. Nixon and A. S. Aguado. *Feature Extraction & Image Processing*. Elsevier Academic Press, Second Edition, 2008. ISBN 978-0-1237-2538-7.

F. Odone, A. Barla, and A. Verri. Building Kernels from Binary Strings for Image Matching. *IEEE Transactions on Image Processing*, 14:169–180, 2005.

C. Papageorgiou and T. Poggio. A Trainable System for Object Detection. *International Journal of Computer Vision*, 38(1):15–33, 2000.

F. Perronnin. Universal and Adapted Vocabularies for Generic Visual Categorization. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 30, pages 1243–1256, 2008.

G. Peterson and H. Barney. Control Methods used in a Study of the Vowels. In *Journal of the Acoustical Society of America*, volume 24, pages 175–184, 1952.

N. Pinto, D. D. Cox, and J. J. DiCarlo. Why is Real-World Visual Object Recognition Hard? *PLoS Computational Biology*, 4(1):e27, 2008.

J. C. Platt. A Resource-Allocating Network for Function Interpolation. *Neural Computation*, 3:213–225, 1991.

J. C. Platt. Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods. In *Advances in Large Margin Classifiers*, pages 61–74. MIT Press, 1999.

J. C. Platt, N. Cristianini, and J. Shawe-Taylor. Large Margin DAGs for Multiclass Classification. In *Advances in Neural Information Processing Systems (NIPS'00)*, volume 12, pages 547–553, 2000.

J. Ponce, T. Berg, M. Everingham, D. Forsyth, M. Hebert, S. Lazebnik, M. Marszalek, C. Schmid, B. Russell, A. Torralba, C. Williams, J. Zhang, and A. Zisserman. Dataset Issues in Object Recognition. In *Toward Category-Level Object Recognition*, pages 29–48. 2006.

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.

J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers, 1993. ISBN 1-55860-238-0.

J. Ren, Y. Shen, S. Ma, and L. Guo. Applying Multi-class SVMs into Scene Image Classification. In *Proceedings of the 17th International Cconference on Innovations in Applied Artificial Intelligence: in IEA/AIE'2004*, pages 924–934. Springer Verlag Inc, 2004.

R. Rifkin and A. Klautau. In Defense of One-vs-All Classification. *Journal of Machine Learning Research*, 5:101–141, 2004.

Y. Rubner, C. Tomasi, and J. Guibas. A Metric for Distributions with Applications to Image Databases. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV'98)*, 1998.

H. Sahbi and D. Geman. A Hierarchy of Support Vector Machines for Pattern Detection. volume 7, pages 2087–2123. Journal of Machine Learning Research, 2006.

J. Salomon, S. King, and M. Osborne. Framewise Phone Classification using Support Vector Machines. pages 2645–2648. ICSLP, Denver, Colorado, USA, 2002.

F. S. Samaria and A. Harter. Parametrization of a Stochastic Model for Human Face Identification. In *IEEE Workshop Applications Computer Vision*, pages 138–142, 1994.

R. S. Shadafan and M. Niranjan. A Dynamic Neural Network Architecture by Sequential Partitioning of the Input Space. In *Neural Computation*, volume 6, pages 1202–1222. MIT Press, 1994.

J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, Cambridge, UK, 2004.

J. Shuiwang, Y-X. Li, Z-H. Zhou, S. Kumar, and J. Ye. A Bag-of-Words Approach for Drosophila Gene Expression Pattern Annotation. *BMC Bioinformatics*, 10(1):119, 2009.

J. Sivic and A. Zisserman. Video Google: A Text Retrieval Approach to Object Matching in Videos. In *Proceedings of the Ninth IEEE International Conference on Computer Vision (ICCV'03)*, pages 1470–1478, 2003.

E. B. Sudderth, A. Torralba, W. T. Freeman, and A. S. Willsky. Describing Visual Scenes Using Transformed Objects and Parts. *International Journal of Computer Vision*, 77(1-3):291–330, 2008.

N. Tishby, F. C. Pereira, and W. Bialek. The Information Bottleneck Method. In *The 37th Annual Allerton Conference on Communication, Control, and Computing*, pages 368–377, 1999.

T. Tuytelaars and K. Mikolajczyk. Local Invariant Feature Detectors: A Survey. Now Publishers Inc, 2008. ISBN 1601981384.

K. E. A. van de Sande, T. Gevers, and C. G. M. Snoek. Evaluating Color Descriptors for Object and Scene Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (in press), 2010.

Jan C. van Gemert, Jan-Mark Geusebroek, Cor J. Veenman, Cees G. M. Snoek, and Arnold W. M. Smeulders. Robust Scene Categorization by Learning Image Statistics in Context. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR'06) Workshop*, page 105, 2006.

Jan C. van Gemert, Cees G.M. Snoek, Cor J. Veenman, Arnold W.M. Smeulders, and Jan-Mark Geusebroek. Comparing Compact Codebooks for Visual Categorization. *Computer Vision and Image Understanding*, 114(4):450 – 462, 2010. Special issue on Image and Video Retrieval Evaluation.

V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer, N.Y., 1995.

V. N. Vapnik, S. Golowich, and A. Smola. Support Vector Method for Function Approximation, Regression Estimation, and Signal Processing. pages 281–287. Advances in Neural Information Processing Systems (NIPS'97), 1997.

L. Wang. Toward a Discriminative Codebook: Codeword Selection across Multi-resolution. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'07)*, pages 1–8, 2007.

L. Wang, L. Zhou, and C. Shen. A Fast Algorithm for Creating a Compact and Discriminative Visual Codebook. In *Proceedings of the Tenth European Conference on Computer Vision (ECCV'08)*, pages 719–732. LNCS 5305, 2008.

J. Willamowski, D. Arregui, G. Csurka, C. R. Dance, and L. Fan. Categorizing Nine Visual Classes using Local Appearance Descriptors. In *ICPR Workshop on Learning for Adaptable Visual Systems*, 2004.

J. Winn, A. Criminisi, and T. Minka. Object Categorization by Learned Universal Visual Dictionary. In *Proceedings of the Tenth IEEE International Conference on Computer Vision (ICCV'05)*, volume 2, pages 1800–1807, 2005.

A. P. Witkin. Scale-space Filtering. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 1019–1022, 1983.

J. Wu and J. Rehg. Beyond the Euclidean Distance: Creating Effective Visual Codebooks using the Histogram Intersection Kernel. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV'09)*, 2009a.

J. Wu and J. Rehg. CENTRIST: A Visual Descriptor for Scene Categorization. Technical report, GIT-GVU-09-05, GVU Center, Georgia Institute of Technology,, 2009b.

J. Yang, D. Zhang, A. Frangi, and J. Yang. Two-dimensional PCA: A New Approach to Appearance-based Face Representation and Recognition. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 26, pages 131–137, 2004.

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 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'08)*, pages 1–8, 2008.

L. Yingwei, N. Sundararajan, and P. Saratchandran. A Sequential Learning Scheme for Function Approximation by using Minimal Radial Basis Function Neural Networks. In *Neural Computation*, volume 9, pages 461–478, 1997.

J. Zhang, M. Marszalek, S. Lazebnik, and C. Schmid. Local Features and Kernels for Classification of Texture and Object Categories: A Comprehensive Study. In *International Journal of Computer Vision*, volume 73, pages 213–238, 2007.

W. Zhang, A. Surve, X. Fern, and T. Dietterich. Learning Non-Redundant Codebooks for Classifying Complex Objects. In *Proceedings of the 26th International Conference on Machine Learning (ICML'09)*, 2009.

Y. Zhao, G. Karypis, and U. Fayyad. Hierarchical Clustering Algorithms for Document Datasets. In *Data Mining and Knowledge Discovery*, volume 10, pages 141–168, 2005.