Classification, Detection and Segmentation of Deformable Animals in Images

Thesis submitted in partial fulfillment
of the requirements for the degree of

Master of Science (by Research)
in
Computer Science Engineering

by

Omkar M Parkhi
200807012
omkar.parkhi@research.iiit.ac.in

Center For Visual Information Technology
International Institute of Information Technology
Hyderabad - 500 032, INDIA

International Institute of Information Technology
Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled " Classification, Detection and Segmentation of Deformable Animals in Images " by Omkar M. Parkhi, has been carried out under my supervision and is not submitted elsewhere for a degree.

Date	Adviser: Prof. C. V. Jawahar
Date	Adviser: Prof. A. P. Zisserman

To my family

Acknowledgements

It would not have been possible to produce this work without the guidance of my advisors, Prof. C. V. Jawahar and Prof. A. P. Zisserman. Their encouragement and support has been the key motivating factor during these years. Many thanks to Dr. Andrea Vedaldi for his invaluable support during this work. It would not have been the same without him.

I am grateful to CVIT's annotations team, Phani, Rajan, Nandini and Girish who provided tremendous help by tirelessly annotating the images. Thanks to Neela for testing annotation interfaces in her vacation time. Thanks to R. S. Satyanarayana for all the administrative work throughout these years. I am grateful to Prof. P. J. Narayanan, Prof. Jayanthi Sivaswamy and Prof. Anoop Namboodiri for their help in CVIT.

I am thankful to everyone in CVIT and VGG, Oxford for providing a wonderful atmosphere during the work. Special thanks to Marcin, Matthew, Victor, Patrick, James, Florian, Varun and Relja for their help. Thanks to my IIIT friends, Wasif, Laxit, Mrityunjay, Soumith, Raman, Rahul, Chandrika, Karthika, Anand, Gopal, Nataraj, Pramod, Mayank and Siddharth for their support. Special thanks to Sreekanth and Mihir for their friendly advice and support.

I would also like to express my gratitude towards everyone in my family. It would not have been possible to even dream of studying further without the support of my mother. I am indebted to her forever...

Abstract

This thesis deals with the problem of classification, detection and segmentation of objects in images. We focus on classes of objects which have considerable deformations, with the categories of cats and dogs as a case study. Many state of the art methods which perform well on the task of detecting a rigid object category, like bus, airplane, boats etc., have a poor performance on these deformable animal categories. The well known difficulty in automatically distinguishing between cats and dogs in images, has been exploited in web security systems: ASIRRA, a system developed by Microsoft Research, requires users to correctly select all cat images from the 12 cats and dogs images shown to them to gain access to a web service. Beyond this, the problem of classifying these animals into their breeds is a challenging problem even for humans. Developing machine learning methods to solve these challenging problems requires the availability of reliable training data. Here, the popularity of cats and dogs as pets provides a chance of collecting this data from various sources on the internet where they are often present in images and videos (together with people). As a part of this work, we propose: a novel method for detecting cats and dogs in the image; and, a model for classifying images of cats and dogs according to the species. We also introduce a dataset for fine grained classification of pet breeds, and develop models to solve the problem of classifying pet images according to the breed. In the process we also segment these objects.

For detecting animals in an image, we propose a mechanism based on a combination of a template based object detector and a segmentation algorithm. The template based detector of Felzenszwalb *et al.* [43] is used to first detect the distinctive part of the object, and then an iterative segmentation process extracts the animal by minimizing an energy function based over a conditional random field using GraphCuts. We show quantitatively that our method works well and substantially outperforms whole-body template-based detectors for these highly deformable object categories, and indeed achieves accuracy comparable to the state-of-the-art on the PASCAL VOC competition, which includes other models such as bag-of-words.

For the task of subcategory classification, a novel dataset for pet breed discrimination, the IIIT-OXFORD PET dataset is introduced. The dataset contains 7,349 annotated images of cats and dogs of 37 different breeds. These images were selected from various sources on the internet. In addition to the pet breed, annotations include a pixel level segmentation of the body of each animal, and a bounding box marking its head. This data set is the first of its kind for pet breed classification and should provide an important benchmark for researchers working on fine grained classification.

For the classification task, we propose a model to estimate a pet breed automatically from an image. The model combines shape, captured by a deformable part model detecting the pet face, and appearance, captured by a bag-of-words model that describes the pet fur. Two classification approaches are discussed: in a hierarchical approach a pet is first classified into the dog or cat species, and then classified into its corresponding breed; and a flat one, in which the breed is obtained directly. For the task of breed classification, on our 37 class dataset, an average accuracy of about 60% was achieved, a very encouraging result considering the difficulty of the problem. Also, these models are shown to improve probabilities of breaking the challenging Asirra test by more than 30%, beating all previously published results.

Contents

List of Figures

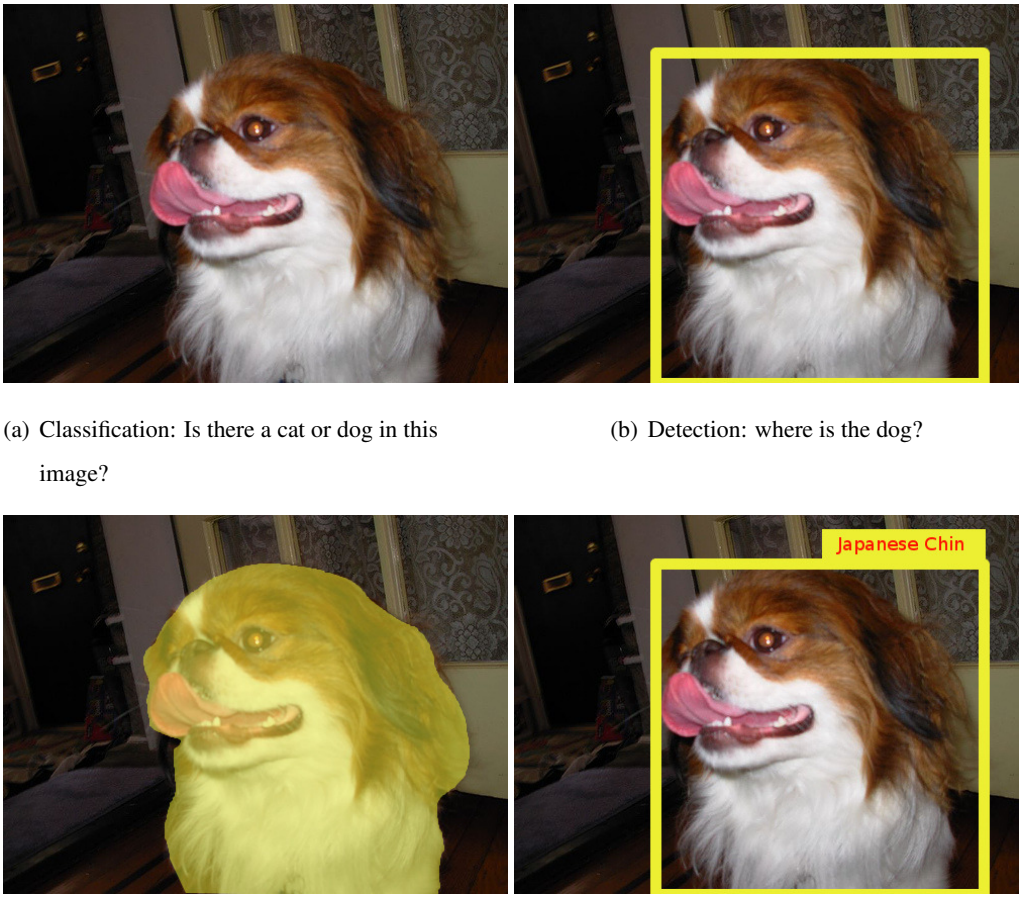
List of Tables

Chapter 1

Introduction

The objective our work is to classify, segment and detect deformable objects in images. Specifically, we focus our attention on detecting Cats and Dogs in the images and classifying them into their sub categories. Figure 1.1 describes these tasks with reference to our work. Given an image, classification system deals with the problem of classifying it as cat or dog (figure 1.1a). Algorithms for object detection localize the object in an image by providing rectangular bounding box around its body (figure 1.1b). Tasks of a segmentation mechanism is to label the pixels of images as belonging to object and background. (figure 1.1c). In Fine grained classification, object is classified according to its breed(figure 1.1d). In the past, researchers have attempted to recognise different categories in images and videos. PASCAL VOC challenge [35] provides data for 20 different object categories such as airplane, bus, boat, train, sheep, birds etc. Caltech 256 dataset [54] is another popular benchmark providing images belonging to 256 different categories. Using such datsets, researchers have tried to build efficient classifiers, detectors and segmentation frameworks for multiple categories together [94]. Many datasets and methods have also been introduced for focused object recognition problems. Researchers have proposed solutions for detection of cars [12], recognizing faces [19], detecting humans [27] etc. in the past. Several focused datasets have also been published. E.g. [18] provides data for animal classification, dataset for retrieving buildings was introduced in [81], dataset focused on birds was introduced in [96], work on fine grained categorization of flowers was discussed in [76, 77, 78]. These datasets and methods are integral parts of contemporary object recognition research. These recognition methods and datasets provide useful insight and motivations for solving problems for other specific categories. Taking the same trend forward, we choose to work on classifying, detecting and segmenting cats and dogs in the images.

Choice of cats and dogs as categories is not at all random. Animal categories have been known to be difficult to detect automatically in images. In PASCAL VOC challenge, performance of state-of-the-art algorithms have been consistently lower on animal categories than most of the other categories [34]. In 2010 edition of the challenge, average precisions reported by state-of-the-art methods for detecting objects such as aeroplane, bus and bicycle were 58.4%, 55.3% and 55.5% respectively and those on cat and dog categories were 47.7% and 37.2% respectively. In another interesting work, researchers



(a) Classification: Is there a cat or dog in this image?

(b) Detection: where is the dog?

(c) Segmentation: Which image pixels belong to dog?

(d) Sub-categorization: What is the breed of this dog?

Figure 1.1 Visual Recognition Problems: Different subtasks.

exploited inability of computer vision and machine learning systems to separate cats from dogs in the images to provide security to webpages on the internet. The ASIRRA animal CAPTCHA developed by Microsoft Research [33, 15] provides security to websites from fraudulent access. The system shows 12 images of cats and dogs to a user and he/she has to select all images of cat to gain access to a website. This system argued to provide more security to the website than text based CAPTCHA mechanisms simply because optical character recognition systems are far better capable of automatically recognising characters than their animal recognition counterparts. These pets are difficult to classify and detect due to the large intra class variations and inter class similarities between different categories. Deformations in shape, size and fur textures, variations in lighting conditions etc. are some of the reasons behind these problems.

Although our methods are directly applied to specific categories, it is applicable to a wider range of categories, especially animals. Also it should provide useful insight into problems of fine grained classification of visually similar objects.

1.1 Problem Definition and Contributions

In this work, we develop a system for a) Detecting cats and dogs in the images. b) Classifying images according to species *i.e.* cats and dogs. c) Classifying images according to subcategories of cats and dogs. For first two tasks, we evaluate performance of our system on PASCAL VOC 2010 and ASSIRA datasets. For the third task, we introduce our own dataset, The IIIT-OXFORD PET dataset which consists of carefully annotated images of cats and dogs subcategories. To classify a given image into its category, we first detect the distinctive part of an animal. Taking clues from this detected part, an entire object is detected using segmentation. Then category specific classifiers are trained on every category. Detected object is represented using powerful feature descriptors and it is classified into its appropriate category with the help of classification techniques.

Dataset Creation In this work, we introduce a novel dataset for pet breed discrimination, the IIIT-OXFORD dataset. The dataset contains $7,349$ annotated images of cats and dogs of 37 different breeds. Images are carefully selected from various sources on the internet. In addition to the pet breed, annotations include a pixel level segmentation of the body of each animal, and a bounding box marking its head. Dataset is discussed in detail in chapter 3.

Detection On highly flexible and deformable objects such as cats and dogs, template-based object detectors such as the deformable parts model by [43] are typically outperformed by simpler bag-of-words models [63, 45, 68, 57, 94, 34]. In this work, we try to enhance performance of these template based detectors on such categories. We use the template-based model to detect a distinctive part for the animal, and detect the rest of the body via segmentation. Iterative segmentation procedure discussed in [85] was used to segment the body of an animal from given image. We use image specific information learnt from detected distinctive part to assist segmentation process. Our method improves upon the performance of the part based models and achieves accuracy comparable to the state-of-the-art on the PASCAL VOC competition. Details of our method and its performance are discussed in chapter 4.

Classification Another contribution of this work is a model to estimate pet breed automatically from an image. The model combines shape, captured by a deformable part model detecting the pet face, and appearance, captured by a bag-of-words model that describes the pet fur. We compare two approaches: a hierarchical one, in which a pet is first assigned to the cat or dog category and then to a breed, and a flat one, in which the breed is obtained directly. We evaluate our model on MSR ASIRRA dataset and The IIIT-OXFORD PET dataset. More detailed discussion can be found in chapter 5.

Figure 1.2 Object Deformations. Images showing variation in the appearance of cats due to deformations.

1.2 Challenges

In this section we look at the challenges faced during the design our system. Lack of reliable data for training and testing models is a fundamental problem for any machine learning algorithm. Additionally, variations in the appearance of the object due to body deformations, inter-class similarities and variations in imaging conditions pose non trivial challenges from object recognition point of view.

Dataset Creation Although there is abundant data available on the internet, each source of data has its own set of problems. Images from social networks for cats and dogs owners were of considerably low quality than those obtained from Flickr. While accumulating breed annotations, some pure breed animals were found to be assigned to wrong labels and cross breed animals were found to be assigned to any one of the category forming cross. On the other hand, images provided by highly reliable institutions were very less in number. Flickr images, though better in image quality posed problem of object repetition as people tend to upload multiple pictures of their pet from different viewpoints. Duplicate images with difference in aspect ratios was a problem observed in images downloaded from Google. Many people upload pictures of their pet wearing clothes and other accessories such as belts, goggles and hats etc. Occlusions due to these accessories provide less visual information for learning the models. This problem was observed in images obtained from all the sources. These problems compelled us to undertake heavy filtering process and ultimately, less than 10% of the downloaded images were selected for final annotations.

Object Deformations Significant change in the appearance can be observed in cases of deformable objects such as animals. Due to the deformations, object can appear to be of different size and shape. Also some parts might appear to be occluded or missing. This is a very serious problem for object detection, especially for part based detectors. Difficulty in detecting such deformable objects is one of the key motivating factor for work presented in this thesis. Figure 1.2 shows some examples of this nature.

Intra-class variations and Inter-class similarities Intra-class variations are variations between objects within a category. Variations in imaging conditions such as lighting, scale, view point are factors

contributing to this problem making task of classification difficult. Object deformations discussed above also contribute to intra-class confusions. Besides, there can be some naturally occurring variations such as color and texture of fur in the objects of the same category. Problem of inter-class similarities arises due to similarities in the appearance of cats of different breeds. Figure 1.3 shows some examples from different but visually similar categories of cats.

(a) (b) (c)

Figure 1.3 Interclass Similarity. Cats of different breeds appear very similar. Images shown in the figure belong a) Bengal b) Occicat and c) Egyptian Mau categories. Yet, they look very similar in appearance.

Clutter and Occlusions Background clutter and occlusions can cause similar effects to that of deformations. They alter shape, make parts disappear and add noise to the data. This problem is of particular importance to cats and dogs as people often tend to upload pictures of their pet wearing some clothes and accessories. Several times, color and texture of animal fur can appear similar to that of the background making automatic segmentation difficult. Some examples of such pictures can be seen in figure 1.4.

Figure 1.4 Clutter and Occlusions. Images uploaded on the web often have pictures of pet wearing clothing.

1.3 Thesis Outline

Brief outline of text in this thesis is as follows, in chapter 2, we present some of the related Machine Learning and Computer Vision literature. Chapter 3 introduces datasets and evaluation protocol used

for assessing our performance. This chapter also describes our own contribution, the IIIT-OXFORD pet dataset. Having reviewed necessary work, datasets and evaluation benchmarks, we explain our method for detection of cats and dogs in chapter 4. In chapter 5, we present a method to classify detected cats and dogs into their appropriate subcategories. Our attempt to break the MSR ASIRRA challenge is also discussed in this chapter. Finally chapter 6 ends the thesis with concluding remarks.

Chapter 2

Background

Classification and Detection of objects in images is considered to be a very challenging task. Problem of recognizing objects has been of interest to the researchers in the community for a long time. Initially, research was focused on solving the problem of recognizing specific objects in constrained set-ups. Lack of powerful computing resources and digital imaging equipments were main reasons for obstacles in the progress of research in this field. During the past decade, we have witnessed tremendous growth in power of computing devices as well as in digital imaging machinery. As a result, researchers have started working on challenging problems like classifying object in natural settings [91, 82] and classifying many categories of objects together [22, 37, 55, 68, 92]. This chapter reviews some of these computer vision and machine learning topics relevant to this work.

This chapter is divided into four parts: the first part (section 2.1) reviews methods of representing images in the form of features, the second part (section 2.2) reviews the relevant classification techniques, third part (section 2.3) reviews relevant work on object segmentation and fourth and the final part (section 2.4) discusses relevant object detection methods.

2.1 Image Representation

Human visual system is very complex system. Even with the all the scientific advances, a very little is known about functioning of this system. Humans can easily discriminate between different objects they observe irrespective of their sizes, sources, lighting conditions, view points, quality and various such attributes. A computer vision system designed to recognize objects also needs to be invariant to these parameters. E.g., a person detector is supposed to detect people despite the variations in clothing and surroundings. Similarly, a face detector should detect a face regardless of person's ethnicity and other facial attributes. Images in their raw pixel representations are often not useful to achieve this invariance. Good features representations are therefore necessary to describe characteristics of objects properly [31, 98]. A good feature representation should be invariant to different sources of variations in the imaging conditions and also in the appearance of the objects they represent. The process of representing images in a meaningful way is called feature extraction. There are many feature representations developed for

computer vision tasks [17, 71, 80, 27]. In following text, we describe some of the feature representations used in this thesis.

2.1.1 Appearance Based Representation

In this section, we will review Scale Invariant Feature Transform [71] and its representation using Bag Of Words Model [89] and Spatial Pyramid Representation [23]. "Bag of Words" or "Bag of Features" builds upon the research in the fields of Natural Language Processing and Information Retrieval. A text document consists of several words from a set defined by a vocabulary. These words or collection of words can then be used to identify the topic and contents of the given document. Indexing based on words helps in retrieving relevant documents based on a query. In computer vision domain, an image is analogous to a document, and visual words in the image are analogous to the words in the document. These visual words are obtained from vector quantizing local features computed in the image. Visual vocabulary necessary for this quantization is generated by clustering local features. Bag of words feature computation consists of following steps:

1. Finding regions of interests
2. Computation of local descriptors
3. Vector quantization of descriptors to form a vocabulary
4. Computation of histogram of visual words.

Process of computing Bag of Words features is schematically shown in figure 2.1.

Finding Regions of Interest: Process of finding regions of interest is known as "feature detection". These are the regions in the image which are invariant to scale variation, rotation and affine transformations. Local descriptors are computed on these regions which inherit and enhance these properties. Several region detectors have been introduced in the past. Some examples of these region detectors are i) Harris Points ii) Harris-Laplace regions iii) Hessian- Laplace regions iv) Harris-Affine v) Hessian-Affine vi) Maximally Stable Extremal Regions. These detectors use low level image processing operations to find the regions in the image. Smoothing by a Gaussian kernel in a scale-space representation followed by local derivative operations are typically the steps in finding such regions. These regions are mainly used for solving correspondence problems. More detailed discussion and comparison of these detectors can be found in [74].

For classification tasks, uniform sampling of the points from an image was shown to be beneficial than using sparse detected points [38]. In uniform sampling, a spatial grid is laid on top of the image and interest points are chosen to be the intersection points of this grid. 5×5 pixels spacing of the grid is common in practice but this parameter can also be determined experimentally. Figure 2.3[a-d] shows the examples of sparse and dense regions and computation of local descriptor.

Local Descriptor Computation: In this step, regions of interest detected in the previous step are represented in the form of local descriptors. In our experiments, we use Scale Invariant Feature Transform (SIFT) [71] as a local feature descriptor. SIFT descriptor is a histogram of gradient location and orientation and is computed on normalized image patches. The location is quantized into a 4×4 location grid and the gradient direction is quantized into 8 orientation bins. This results in a 128 dimensional feature vector. In our experiments, we use VLFeat [93] implementation of SIFT. Figure 2.3[e] shows approach for computing SIFT descriptors.

Vocabulary Construction: Having computed feature descriptor, vocabulary is computed by collecting similar features into clusters. Cluster centers representing similar features are called as Visual Words. Clustering algorithms such as K-Means are used for finding the cluster centers. Number of cluster centers are determined experimentally. In our work, fast algorithm for clustering introduced in [32] was used to compute cluster centers.

Histogram Computation: To compute feature histograms, local feature descriptors are mapped to their nearest visual word by using some distance metric. This mapping assigns a visual word to every feature descriptor. Histograms of visual words are then computed by assigning a bin to every visual word. Thus an image with varying number of local descriptors always results in the same feature vector dimension as long as the underlying vocabulary remains constant. Loss of spatial information is one of the drawback of this scheme. In practice, it is often important to encode spatial information into features. To preserve the spatial structure, spatial pyramid representation is used for computing histograms [23].

In this approach, an image is represented as collection of cells on different coarse-fine levels. On the basic level, image is taken as one unit entity producing single histogram as previously discussed. However on subsequent levels, the image is spatially divided into different cells and histograms are computed for every cell individually. Number of divisions increase as the level is increased. All computed histograms are concatenated to form a final feature vector. There is no fixed formula for determining number of cells on every level. Levels of the pyramid are also determined experimentally. However, it is common practice to divide an image into 4^l cells at every level l and representations going upto 2 levels. Figure 2.2 shows the concept of spatial pyramids diagrammatically.

2.1.2 Shape Descriptors

As their name suggests, these descriptors convey information about shape of the object and are used very regularly in the object detection and classifications tasks. For the experiments in this thesis, we have also used Histogram of Oriented Gradients (HOG) feature descriptor. Since their introduction [27], these features have been widely used by many successful object detection and classification systems [41]. The descriptor describes an object by the distribution of gradients intensity and edge directions. These descriptors are computed by dividing the image into small connected regions, called cells, and computing a histogram of gradient directions or edge orientations for the pixels within the cell. These

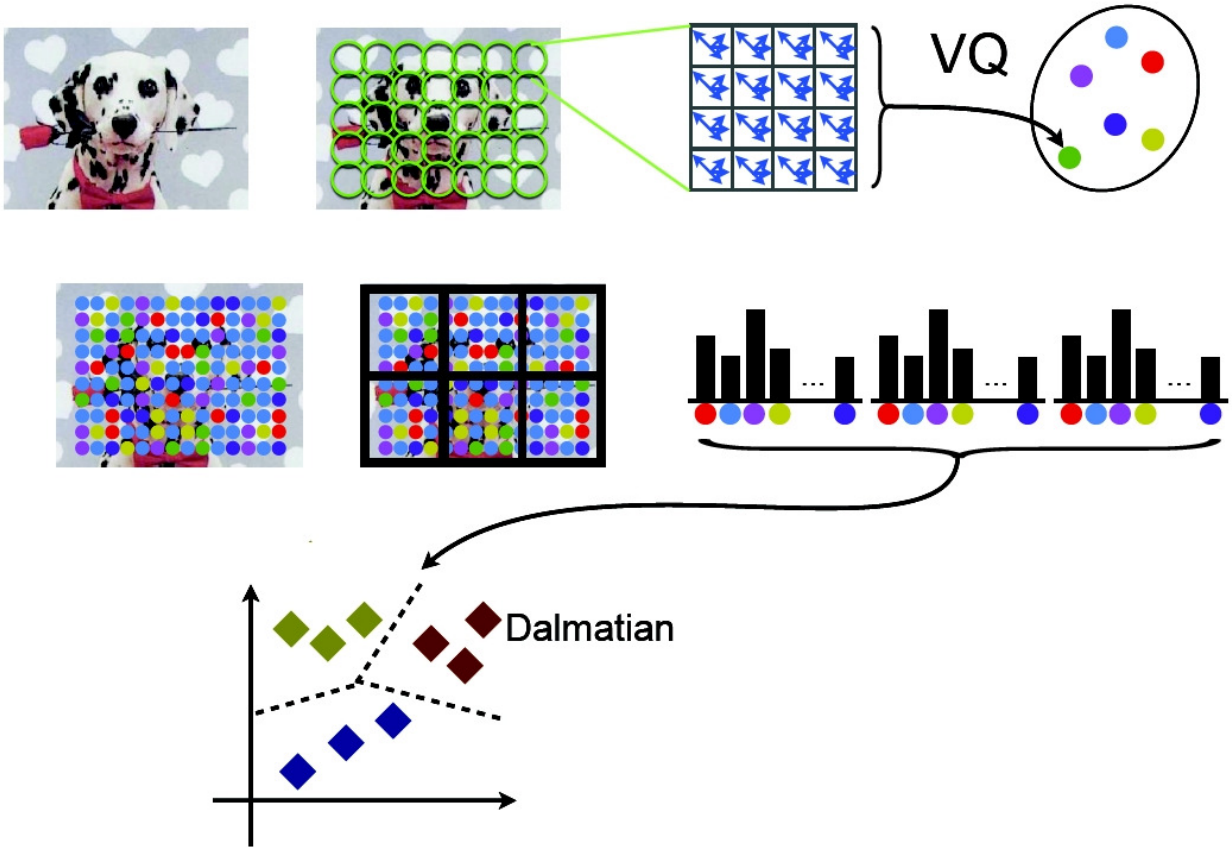


Figure 2.1 Bag of words feature computation. Figure showing procedure for computing Bag of Words features on image and classifying it according to its subcategory. Figure courtesy [93].

histograms are then combined to represent the descriptor. Descriptors computation consists of 4 major steps described below:

1. Gradient computation:

   Gradients are computed by applying derivative masks on the image. 1-D centered derivative mask ($[-1\ 0\ 1]$ and $[-1\ 0\ 1]^T$) is applied to an image in both horizontal and vertical direction. Images are sometimes preprocessed for gamma and color normalization. More complex masks such as 3×3 sobel operator are also used in some implementations. Output of this step is the computation of magnitude and direction of the gradient at every pixel in the image.

2. Cell Histogram:

   In this step, pixels belonging to same cell (say 5×5 neighbourhood) vote for forming orientation histogram. Each pixel in the cell votes for orientation bin according to its gradient direction. Weight of the vote is dependent on the magnitude of the gradient.

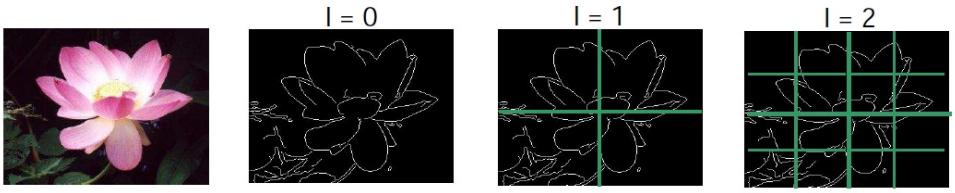


Figure 2.2 Spatial Pyramid Representation. Spatial pyramid representation of [23] Image showing an image and grids for $l = 0$ to $l = 2$.

3. Descriptor Block:

   Cells are grouped locally to form larger spatially connected blocks to account for the illumination and contrast changes. The HOG descriptor is then formed as a vector of the components of the normalized cell histograms from all the block regions. Circular and Rectangular geometry has been discussed in the literature for construction of these blocks. The block vectors are normalized for better performance.

   Figure 2.4 shows schematic of HOG feature computation. In our experiments, we have used HOG for distinctive parts detection in chapters 4 and 5.

2.2 Classification

Given any image and a set of categories, say cats and dogs, task of a classifier is to assign the image to its appropriate class. Ideally, posterior probability can be used to carry out this classification. To classify a given image x into one of the K classes, a classifier would compute a posterior probability $P(c_k|x)$ and assign the image to class resulting in highest posterior probability. In practice, we do not have prior knowledge about $P(c_k|x)$. This means that a classifier has to produce output by learning some statistics from the data. Depending on the way in which classifier learns these statistics, it can be grouped as a generative classifier or a discriminative classifier.

2.2.1 Generative Classifiers

These type of classifiers first model conditional probability $P(x|c_k)$ and then produce posterior probability with the use of Bayes' rule. Idea here is to model commonalities of instances belonging to same class. Gaussian mixture models is an example of generative classifiers. Given M Gaussians, mixture of Gaussians can be written as

$$P(x|c_k) = \sum_{m=1}^{M} \Pi_m N(x|\mu_m, \Sigma_m) \tag{2.1}$$

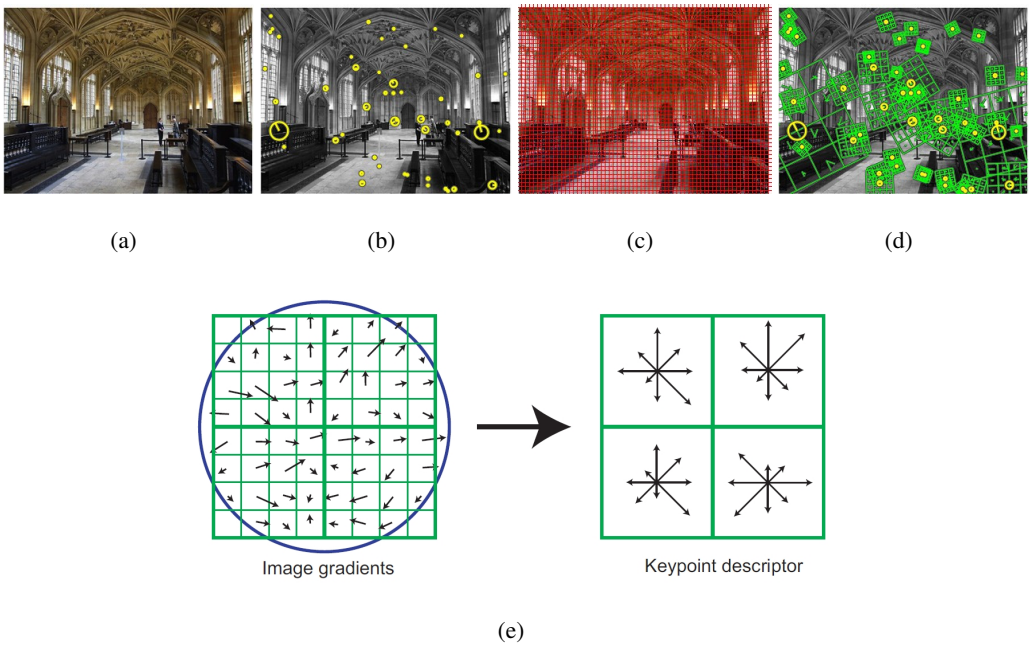


Figure 2.3 Appearance Feature Computation [a-d] Steps involved in computation of local features. [a] Input Image [b] Sparse region of interests [c] Dense region of interest on 5×5 grid. [d] SIFT feature computation. [e] The SIFT descriptor of [71]. On the left are the gradients of an image patch. The blue circle indicates the Gaussian center-weighting. These samples are then accumulated into orientation histograms summarizing the contents over 4×4 subregions, as shown on the right, with the length of each arrow corresponding to the sum of the gradient magnitudes along that direction within the region. A 2×2 descriptor array computed from an 8×8 set of samples is shown here. Figures courtesy [93].

Where the mixing coefficient Π_m must satisfy $0 \leq \Pi_m \leq 1$ and $\sum_m \Pi_m = 1$. The expectation-maximization algorithm is a maximum likelihood algorithm that can be used to fit this model to the training data.

2.2.2 Discriminative Classifiers

These classifiers attempt to model the differences between the categories. It is done by finding a discriminant surfaces separating the classes. As a result, these classifier directly model the posterior probability, $P(c_k|x)$. Linear classifier, nearest neighbour classifiers, Support Vector Machines are all examples of discriminative classifiers.

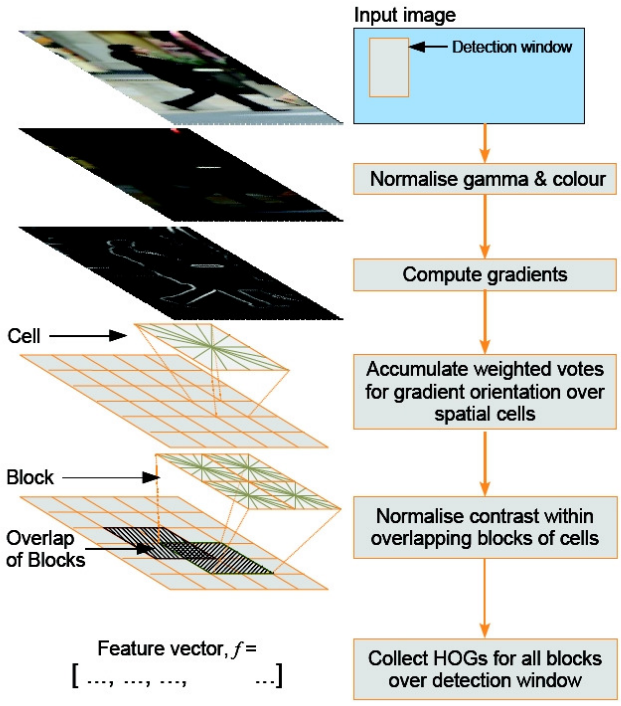


Figure 2.4 HOG Feature Computation An overview of static HOG feature extraction [27]. The detector window is tiled with a grid of overlapping blocks. Each block contains a grid of spatial cells. For each cell, the weighted vote of image gradients in orientation histograms is performed. These are locally normalised and collected in one big feature vector.

2.2.2.1 Support Vector Machines

Support Vector Machine is a popular discriminative classifier which learns the decision boundary with the largest margin to the training data. On many classification problems SVMs have been shown to be very robust and yielding high empirical performances. They belong to the category of supervised learning methods. The underlying idea is to separate p dimensional data points using $p-1$ dimensional hyperplane. Since there can be many such hyperplanes, there exists a problem of selecting the hyperplane which provides the maximum separation of the data. SVM tries to select a hyperplane such that its distance from nearest data points on each side is maximized. Such a hyperplane is known as the "maximum-margin hyperplane" and the classifier it defines is known as a "maximum margin classifier".

Consider the problem of finding a maximum margin hyperplane for data samples D which classifies given sample x into its class. Such a hyperplane can be written as,

$$w \cdot x + b = 0$$

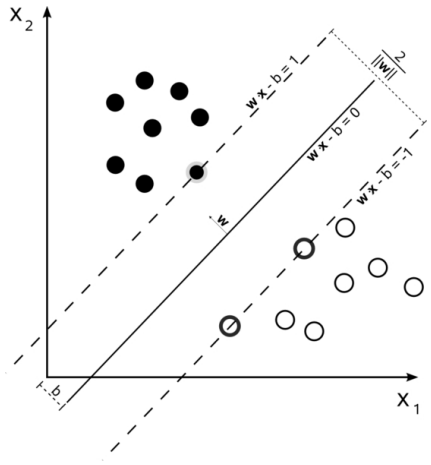


Figure 2.5 Support Vector Machines Maximum-margin hyperplane and margins for an SVM trained with samples from two classes. Figure courtesy wikipedia.

w is a normal vector perpendicular to the hyperplane and b is a bias term. We want to choose the w and b to maximize the margin, which is the distance between the two parallel hyperplanes that are as far apart as possible while still separating the data. These hyperplanes can be described by the equations,

$$w \cdot x + b = 1 \text{ and } w \cdot x + b = -1$$

Distance between these two planes is given by $(\frac{2}{\|w\|})$. To maximize this distance we need to minimize $\|w\|$. Hence the optimization problem becomes,

$$\begin{gathered} \text{Minimize in } (w, b), \\ \frac{1}{2} \|w\|^2 \\ \text{subject to conditions, } y_i(w \cdot x_i + b) \geq 1 \end{gathered} \tag{2.2}$$

Solving this we get,

$$w = \sum_{i=1}^{n} \alpha_i y_i x_i$$

And the resultant classification y for input sample x is given by,

$$y = sign(\sum_{i=1}^{n} \alpha_i y_i x_i^T \cdot x + b) \tag{2.3}$$

Figure 2.5 shows the concept of svm in 2-D space. SVM model described here is called " Hard Margin Model ". "Soft Margin Methods" have also been presented by the researchers making SVMs more powerful. It is important to note that these forms of SVM find a linear hyperplane separating the data points. To separate data with more complex distributions, a 'kernel trick' was researched allowing

hyperplanes to be non-linear. An algorithm for "Kernel SVM" is similar to explained above except that dot product of $x_i^T \cdot x$ is replaced by non-linear kernel function $K(x_i, x)$. The resultant classification y for input sample x is given as,

$$y = sign(\sum_{i=1}^{n} \alpha_i y_i K(x_i, x) + b). \tag{2.4}$$

2.2.2.2 Kernels in Computer Vision

Today, kernel methods are widely used for solving computer vision problems. They have been applied to solve problems such as Object Classification, Object Detection, Optical Character Recognition, Image and Video Retrieval etc. For all these problems, algorithms based on kernel methods have been shown to achieve state-of-the- art performance.

Some of the popular kernels used are,

- *Linear Kernel:*
  It has very simple formulation and can be computed very quickly. It is the basic form of a kernel. Kernel function is a dot product of feature vectors.

$$k(x, y) = \langle x, y \rangle \tag{2.5}$$

- *Radial Basis Function Kernel:*

$$k(x, y) = exp^{\frac{-\|x-y\|^2}{2\sigma^2}} \tag{2.6}$$

- *Generalised Intersection Kernel:*

$$k(x, y) = \sum_i min(x_i, y_i)^\gamma \tag{2.7}$$

- *Exponential χ^2 Kernel:*

$$k(x, y) = e^{-\gamma \sum_i \frac{(x_i - y_i)^2}{(x_i + y_i)}} \tag{2.8}$$

- *χ^2 Kernel:*

$$k(x, y) = 2 \sum_i \frac{x_i y_i}{x_i + y_i} \tag{2.9}$$

While using spatial pyramid features, these kernel functions can be applied to entire feature vector or can also be applied to individual cells forming the pyramid. Several of these kernels can be combined together to form a classifier in method known as "Multiple kernel learning".

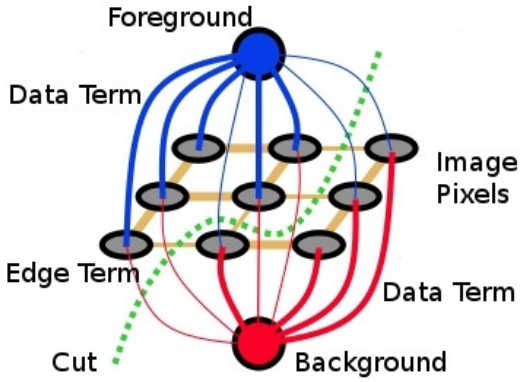


Figure 2.6 Graph Cut Setup. Image pixels (gray) are connected to foreground (blue) and background (red) nodes by data terms. Edge terms (brown) connect pixels to their neighbouring pixels. Algorithm then provides an optimum cut (green) generating a segmentation.

2.2.2.3 Multiple Kernel Learning

It is often difficult to build a classifier using just a single feature representation as a source of information. *E.g.* While classifying fruits, it is difficult to separate apples from oranges just using shape features or to distinguish between guava and pear using just the color. It is indeed useful to combine information coming from different sources to solve a given problem. The challenge is to select the importance of an individual feature. This problem is solved by multiple kernel learning [16].

$$k = \sum_{f=1}^{F} d^f k_f \tag{2.10}$$

Here, k_f is the individual feature kernel and d_f denotes its importance (weight). Kernel method based on support vector machines was shown to perform better on object classification tasks [92]. MKL was also shown to achieve state-of-the-art performance for object detection by [94].

2.3 Segmentation

In this thesis we address the problem of automatically separating animal (foreground) pixels from the background pixels in the images. GraphCut energy minimization technique has been a very popular technique used for these kinds of segmentation tasks. In this technique, properties of an image are represented in the form of a graph. Each graph has a foreground and background node. Every pixel in the image represents a node in the graph and is connected to foreground and background nodes by the "data term" or the "smoothness term" and is also connected to other neighbouring nodes by the "edge term" or the "pairwise term". Figure 2.6 shows the graphcut representation discussed above.

In typical formulation, an image is considered to be an array of gray scale values, $z = (z_1, z_2..., z_N)$. Segmentation of an image is considered to be an array of labels $\underline{\alpha} = (0, 1)$. Label 0 is assigned to the background pixels while label 1 is assigned to the foreground pixels.

The data term is modeled as a probability of pixel belonging to foreground and background. Total energy represented by the data terms can be given as,

$$U(\underline{\alpha}, z) = \sum_i \sum_j -log(h(z_i, \alpha_j)) \tag{2.11}$$

Where h is histogram build for foreground and background pixels separately and used to model likelihood of a given pixel belonging to foreground and background. Pairwise term is modeled as similarity preserving function between pixel and its neighbour. Total energy of the pairwise terms is given as,

$$V(\underline{\alpha}, z) = \gamma \sum_{(m,n)\in C} D(m, n)^{-1} \left[\alpha_n \neq \alpha_m\right] e^{-\beta(z_m - z_n)^2} \tag{2.12}$$

Where C is set of neighbouring pixels and D is an Euclidean distance between the two negibouring pixels. This form of energy function encourages a uniform label assignment in the region of similar gray level. Parameter γ determines the total contribution of the edge terms into the total energy calculations and is determined experimentally. Total energy to be minimised is given as

$$E(\alpha, z) = U(\underline{\alpha}, z) + V(\underline{\alpha}, z) \tag{2.13}$$

A cut producing the segmentation is then provided by minimizing this energy. In this thesis, we use GrabCut segmentation method [85] which provides segmentation by iterative energy minimization. In this method, data terms are modeled by the Gaussian mixture models instead of the histograms described above. In the beginning of the process, initializations in the form of labeled pixels are provided to build initial models for the data terms. These initializations are called as "seeds". GraphCut algorithm is then executed multiple times . After every iteration, GMMs for foreground and background are re-parametrised based on the segmentation output of previous step. The iterations stop after a predetermined criterion is satisfied. Some examples showing use of seeds and resulting segmentation can be seen in figure 2.7. In the original work, seeds were provided by the user interaction. In our case, seeds are selected based on distinctive part detection to facilitate automatic object detection.

2.4 Detection

Detecting and localizing objects in images is one of the very challenging problems in Computer Vision. Objects in such categories can vary greatly in appearance. The variance can arise from changes in illumination, viewpoint, deformations of the object, variations in the shape and sizes in the objects of same category etc. Several researchers have tried to solve these problem in many different ways. Sliding window mechanism is an example of powerful detection methods. A sliding window detector probes

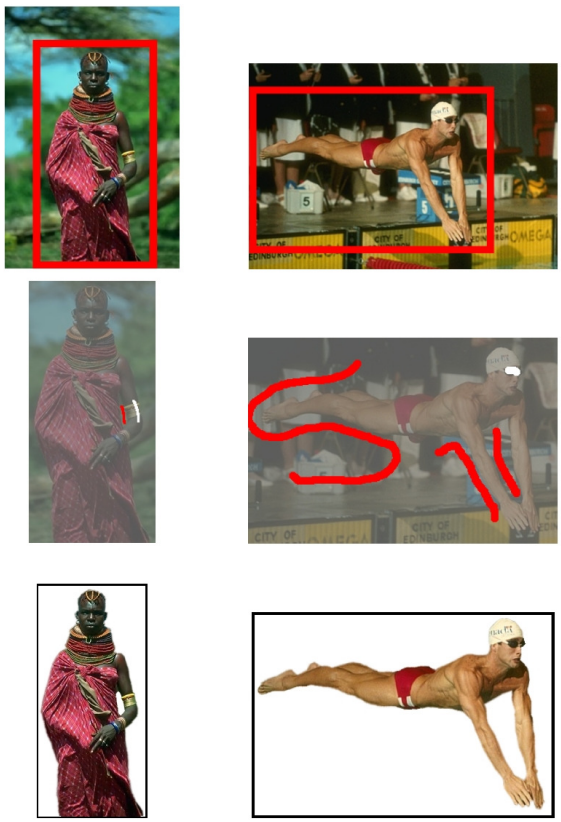


Figure 2.7 GrabCut Segmentation First row shows the original images with superimposed user input (red rectangle). The second row displays all user interactions: red (background seeds), white (foreground seeds). The results obtained by GrabCut [85] are visualized in the third row.

for presence of the object on multiple scales and locations. The Dalal-Triggs detector [27], used sliding window mechanism for detecting humans in an image. This method used a single filter on histogram of oriented gradients (HOG) features to represent an object category. A filter is then applied at all positions and scales of an image. In general, a detector can be seen as a classifier which takes as input an image, a position within that image, and a scale. The classifier determines whether or not there is an instance of the target category at the given position and scale.

Part based methods provide another way of representing objects. These methods represent an object by a collection of parts arranged in a deformable configuration. Pictorial structures framework [40, 47] and grammar based models [75] are examples of such methods. In our work, we use part based method proposed in [41, 43]. We call this method the "Deformable Parts Model".

2.4.1 Deformable Parts Model

This section describes Deformable Parts models discussed in [41, 43, 39] . In our work, we use the implementation provided by authors on their website [42]. This method has been very popular in the computer vision community and has won PASCAL VOC Lifetime Achievement Award. The Deformable parts model represents objects using variable hierarchical structures. This method builds up on the pictorial structures and grammar based models and has been shown to be very effective in detecting objects in the images. This method essentially uses a star-structured part-based model defined by a "root" filter plus a set of parts filters and associated deformation models. The object is represented by one global and several part models. Location of the parts of the object are labeled automatically as manual labeling can be consuming, expensive and most importantly, suboptimal. Latent variable formulation [14] is used to automatically select size and to locate positions of part filters. This is conceptually similar to describing an animal by different parts such as head, torso, legs etc. Only difference is that here the parts sizes and locations are learnt automatically.

Combination of the root filter and part filters is called as a "component". Once the root filter, part filters and their locations are learnt, an object is searched in the test image on different scales and positions. The score at a particular position and scale within an image is given as the score of the root filter at the given location plus the sum over parts of the maximum, over placements of that part, of the part filter score on its location minus a deformation cost measuring the deviation of the part from its ideal location relative to the root. Both root and part filter scores are defined by the dot product between a filter (a set of weights) and a sub window of a feature pyramid computed from the input image. To model visual appearance at multiple scales, the part filters capture features at twice the spatial resolution relative to the features captured by the root filter. The entire process is explained in figure 2.8.

Since different objects of a category can differ greatly in shapes and sizes, single star structure model is often not enough to model these variations. Hence, instead of using one component model, a multi component mixture of star models was developed. The score of a mixture model at a particular position and scale is then computed as maximum over all the components learnt for that category.This way, one can model objects with different aspect ratios and appearance of the object without having to fit a single appearance model for all variations. Some of the example detections produced by this method are shown in figure 2.9.

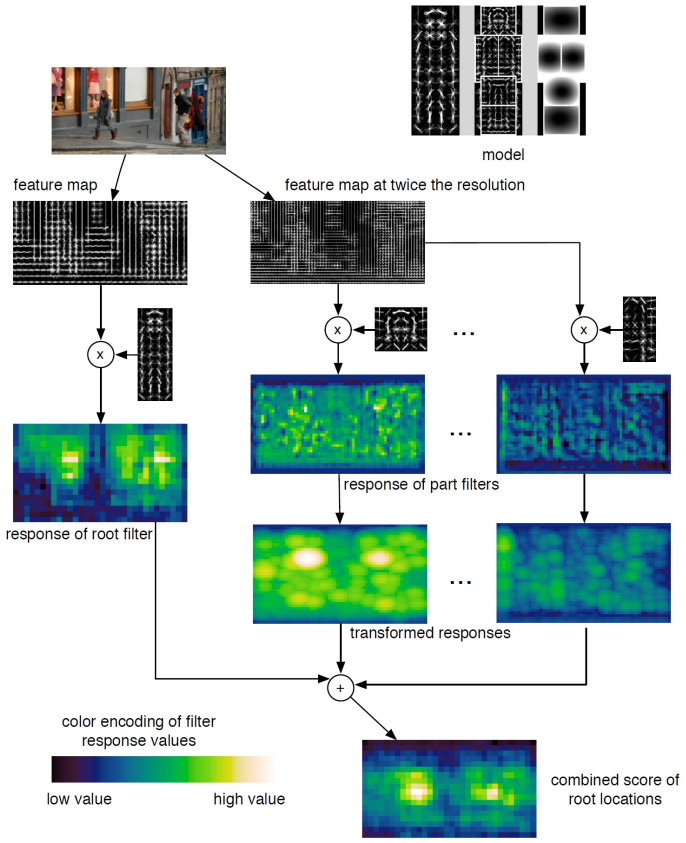


Figure 2.8 Deformable Parts Model The matching process at one scale. Responses from the root and part filters are computed at different resolutions in the feature pyramid. The transformed responses are combined to yield a final score for each root location.

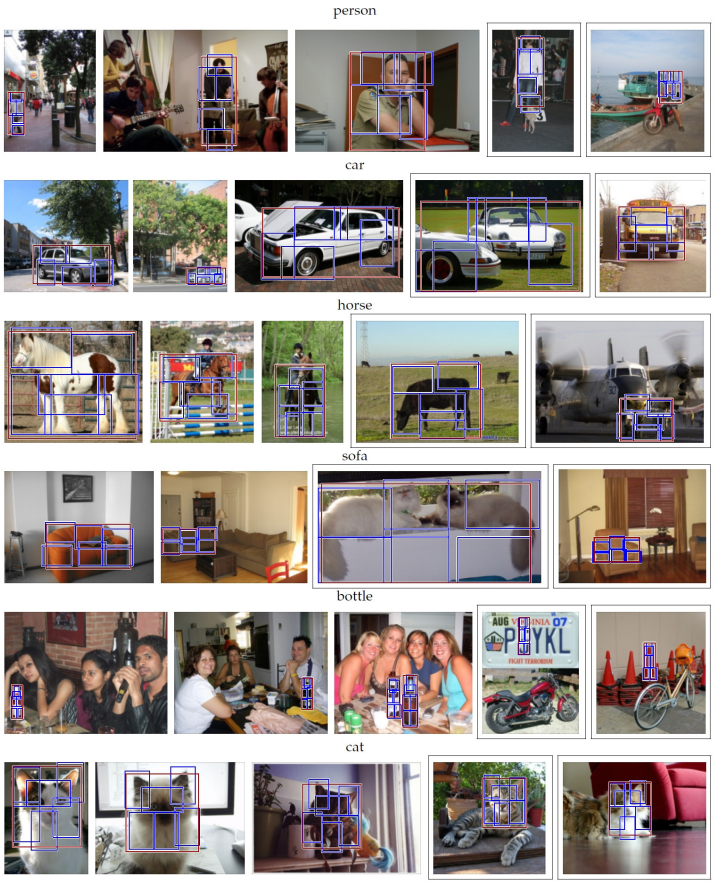


Figure 2.9 Example results of Deformable Parts Model method Examples of high-scoring detections from [43] on the PASCAL 2007 dataset, Last two in each row illustrate false positives for each category

Chapter 3

Dataset

Image databases are the essential part of object recognition research. They are required for learning visual object models and for benchmarking performances of classification, detection, and segmentation algorithms. Several datasets are publicly available for these tasks. For our work, we have used three different datasets. All of them are designed for solving different type of problems. In this chapter we also introduce our new dataset contribution, the IIIT-OXFORD PET dataset and briefly review other two datasets used in this thesis viz. VOC 2010 Challenge dataset and MSR Asirra dataset.

3.1 PASCAL VOC 2010 Dataset

VOC 2010 dataset is the key dataset used in this thesis. Majority of work discussed in the thesis is modeled and evaluated on the images in this dataset.

Background There are several datasets made publicly available to provide base for vision related tasks. Some of the examples are UIUC Car Dataset [12], Caltech 4 [46], Caltech 256 [54] etc. For a long time, these datasets served as perfect testing platforms providing some of the classic works in the field of computer vision and machine learning. These datasets however, presented very little variance in terms of factors such as appearance, shape, viewpoints and orientations of objects of different categories they represent. *E.g.* All photographs in UIUC car dataset are side views of cars, caltech series has just one object of interest present per image and there is very little background clutter. There are some problems with such restricted datasets. (i) some algorithms may exploit these restrictions (for example near-global descriptors with no scale or rotation invariance may perform well on such images), yet will fail when the restrictions do not apply (ii) the images are not sufficiently challenging for the benefits of more sophisticated algorithms (e.g., scale invariance) to make a difference. This means that progress in algorithm capability cannot be assessed. Such limitations were addressed in the design of datasets like PASCAL VOC dataset.

Dataset The PASCAL Visual Object Classes Challenge (PASCAL VOC) [35] started in 2006, making set of annotated images available for research purpose. VOC provides twofold benefits for researchers. First it provides them with highly annotated dataset for developing vision algorithms and secondly, it provides competent benchmark for testing performance of these algorithms. The dataset presents images which are taken in natural settings downloaded from Flickr. We use 2010 edition of PASCAL VOC dataset. The VOC2010 database contains a total of 21,738 annotated images. The data is split in two parts: (i) training and validation data with annotation (ii) test data without annotation. Training and validation data is used for training and initial testing of algorithms while test data provides important benchmark to evaluate their performance. Table 3.1 shows statistics of VOC 2010 dataset. Figure 3.1 shows some of the example images from this dataset.

3.1.1 Ground Truth Annotation

Object level annotations are provided for every image in the dataset as 'Ground Truth'. For each object, following attributes are provided as part of annotations:

- *class:*

  The object class e.g. 'car' or 'bicycle'. Every annotated object is assigned with one of 20 classes.

- *bounding box:*

  An axis-aligned rectangle specifying the extent of the object visible in the image. This is provided in the form of top-left and bottom-right coordinates of rectangle perfectly encapsulating the object. Bounding boxes are useful for training and validation of object detection algorithms.

- *view:*

  Specified orientation of the object such as 'frontal', 'rear', 'left' or 'right'. The views are subjectively marked to indicate the view of the 'bulk' of the object. Some objects have no view specified.

- *truncated:*

  This attribute indicates that the bounding box specified for the object does not correspond to the full extent of the object e.g. an image of a person from the waist up, or a tail or body of an animal extending outside the image.

- *occluded:*

  An object marked as 'occluded' indicates that a significant portion of the object within the bounding box is occluded by another object.

- *difficult:*

Category	**Train**		**Val**		**TrainVal**	
	Img	Obj	Img	Obj	Img	Obj
Aeroplane	283	369	296	369	579	738
Bicycle	228	305	243	309	471	614
Bird	340	486	326	485	666	971
Boat	222	345	210	342	432	687
Bottle	300	507	283	507	583	1014
Bus	180	245	173	253	353	498
Car	523	892	507	882	1030	1774
Cat	502	563	503	569	1005	1132
Chair	469	946	456	944	925	1890
Cow	125	228	123	236	248	464
Diningtable	209	234	206	234	415	468
Dog	591	707	608	709	1199	1416
Horse	209	306	216	315	425	621
Motorbike	225	306	228	305	453	611
Person	1717	3559	1831	3737	3548	7296
Pottedplant	225	408	225	413	450	821
Sheep	152	344	138	357	290	701
Sofa	205	224	201	227	406	451
Train	226	261	227	263	453	524
Tvmonitor	247	342	243	341	490	683
Total	4998	11577	5105	11797	10103	23374

Table 3.1 PASCAL VOC 2010 data composition. The table lists the object categories and for each category the number of images and objects in the training and validation, and test sets in the PASCAL VOC 2010 dataset.

An object marked as 'difficult' indicates that the object is considered difficult to recognize, for example an object which is clearly visible but unidentifiable without substantial use of context. Objects marked as difficult are currently ignored in the evaluation of the challenge.

These annotations are provided in the XML format. An XML file is generated per image, providing information about every annotated object present in that image. Object level annotations are useful for tasks of object detection and image classification. In addition to these annotations, pixel level annotations are provided for some images in the dataset. These annotations are useful for object segmentation task. Object level annotations can be seen in figure 3.1.

Additional Annotations To train and evaluate our distinctive part detector (section 3.2), we annotated every cat and dog category image in the dataset with tight bounding box around the head of the animal. Also for segmentation task, we annotated these images with pixel level annotations. Examples of these annotations can be seen in figure 3.2.

3.1.2 Tasks

Three main tasks are defined on this dataset. One can build a system to perform these tasks and evaluate their performance on test data. It is not mandatory to use VOC training data and annotations for performance evaluation. However VOC expects independent training data used in such cases to be disjoint from testing data. One may choose to tackle all, or any subset of object classes, for example "cars only" or "motorbikes and cars". VOC definitions of "classification","detection" and "segmentation tasks are given as:

- *classification:*

  For each of the twenty object classes predict the presence/absence of at least one object of that class in a test image. The output from a system should be a real-valued confidence of the object's presence so that a precision/recall curve can be drawn.

- *detection:*

  For each of the twenty classes predict the bounding boxes of each object of that class in a test image (if any). Each bounding box should be output with an associated real-valued confidence of the detection so that a precision/recall curve can be drawn.

- *segmentation:*

  For each test image pixel, predict the class of the object containing that pixel or 'background' if the pixel does not belong to one of the twenty specified classes. The output from your system should be an indexed image with each pixel index indicating the number of the inferred class (1-20) or zero, indicating background.

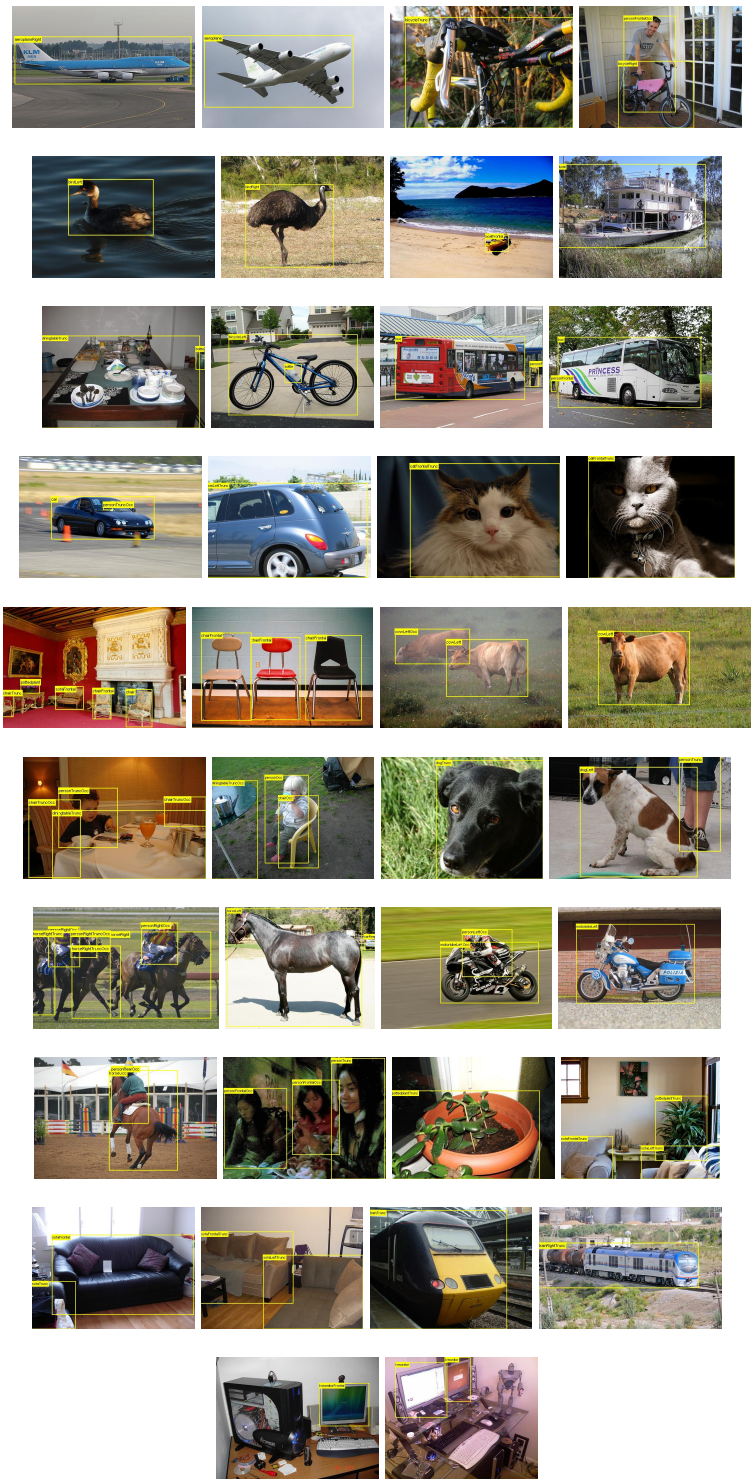
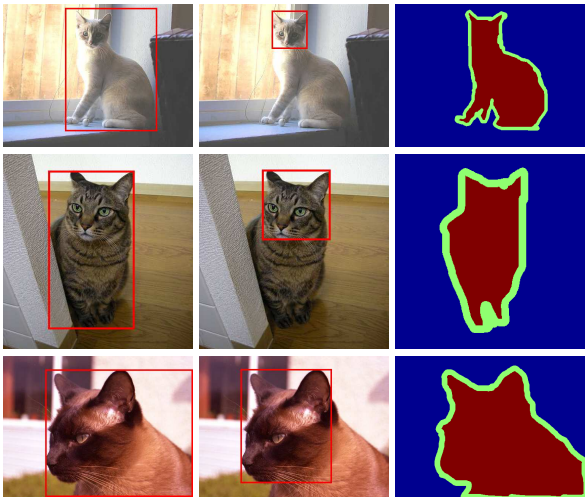


Figure 3.1 PASCAL VOC 2010 dataset. Example images and annotations from the dataset.



(g) Object (h) Distinctive Part (i) Trimap

Figure 3.2 Annotations. (a) The PASCAL VOC annotations are tight bounding boxes around the object instances. (b) Additional annotations for the distinctive object part, in this case cat/dog heads. (c) Pixel-level segmentation of the object also provided by PASCAL VOC.

In chapter 4 we use the VOC dataset images with additional annotations to build our object detector. We mainly present results for detection challenge explained above. However, object segmentation is an integral part of our detection system and will also be discussed in chapter 4.

3.2 The IIIT-OXFORD PET Dataset

As a part of our work to build models for fine grained categorization of pet breeds, we introduce the IIIT-OXFORD PET dataset, set of extensively annotated images collected from the internet. The motivating challenge for this problem is the fact that breeds may differ only by a few subtle details. Additionally, it is difficult to measure these details automatically due to the highly deformable nature of the bodies of such animals. Beyond the technical interest, extracting information from images of pets has a practical side too. People devote a lot of attention to their domestic animals, as suggested by the large number of social networks dedicated to the sharing of images of cats and dogs: PetFinder [10], Catster [4], Dogster [5], My Cat Space [8], My Dog Space [9], The international cat association [7], and several others [2, 1, 11, 3]. In fact, the bulk of the data used to build this dataset has been extracted from tagged images posted daily by the users of these websites.

Background The research on object category recognition has largely focused on the discrimination of well distinguished object categories (*e.g.* , airplane vs cat). Most of the popular international benchmarks discussed previously, (*e.g.* , Caltech-101 [36], Caltech-256 [55] and PASCAL VOC [34]) contain

a few dozens object classes that, *for the most part, are visually dissimilar*. Even in the much larger ImageNet database [29], categories are defined based on a high-level ontology and, as such, any visual similarity between them is more accidental than systematic.

Similar to flower categorization problem discussed in [76, 77, 78], our work concentrates on the problem of *discriminating different breeds of cats and dog*, a challenging example of fine grained object categorization. For this purpose, we introduce a large annotated collection of images of 37 different breeds of cats and dogs. This data constitutes the first benchmark for pet breed classification, and, due to its focus on fine grained categorization, is complementary to the standard object recognition benchmarks. The data, which is publicly available, comes with rich annotations: in addition to a breed label, each pet has a pixel level segmentation and a rectangle localizing its head. A simple evaluation protocol, in the line of the PASCAL VOC challenge, is also proposed to enable the comparison of future methods on a common ground.

3.2.1 Dataset

The *IIIT-OXFORD PET dataset* is a collection of images of cats and dogs of 37 different breeds. It contains 7,349 images of cats and dogs, each annotated with a breed label, a pixel level segmentation marking the body, and a rectangle localizing the head (figure 3.3). Of the 37 breeds, 25 are dogs and 12 are cats. Images are divided into training, validation, and test sets, similarly to the PASCAL VOC data. The dataset contains about 200 images for each breed (of which about 50 for training, 50 for validation, and 100 for testing). A detailed list of breeds is given in table 3.2.1, and example images are given in figure 3.4 and figure 3.5.

Dataset construction. All the images used in the dataset were downloaded from various sources on the internet. Some images were downloaded from specific websites related to cats and dogs such as Catster [4], Dogster [4], AKC [1], CFA [2] etc. Additional images were downloaded by querying on general image collections such as Flickr [6] groups, Google images [52]. Our efforts of tagging images with breeds were simplified by breed information avaliable along with images on social networks and category specific Flickr groups. For each of the 37 breeds, about 2,000 – 2,500 images were downloaded from these data sources to form a pool of candidates for inclusion in the dataset. From this candidate list, images were dropped if any of the following conditions applied, as judged by the annotator: (i) the image was gray scale (ii) another image portraying the same animal existed (which happens frequently in Flickr), (iii) the illumination was poor, (iv) the pet was not centered in the image, or (v) the pet was wearing clothes. The most common problem in all the data sources, however, was found to be errors in the breed labels. Thus labels were reviewed by the human annotators and fixed whenever possible. When fixing was not possible, for instance because the pet was a cross breed, the image was dropped. Overall, up to 200 images for each of the 37 breeds were obtained.

Breed	Tr.	Val.	Ts.	Tot.	Breed	Tr.	Val.	Ts.	Tot.
Abyssinian	50	50	98	198	English Setter	50	50	100	200
Bengal	50	50	100	200	German Shorthaired	50	50	100	200
Birman	50	50	100	200	Great Pyrenees	50	50	100	200
Bombay	49	47	88	184	Havanese	50	50	100	200
British Shorthair	50	50	100	200	Japanese Chin	50	50	100	200
Egyptian Mau	47	46	97	190	Keeshond	50	50	99	199
Maine Coon	50	50	100	200	Leonberger	50	50	100	200
Persian	50	50	100	200	Miniature Pinscher	50	50	100	200
Ragdoll	50	50	100	200	Newfoundland	50	46	100	196
Russian Blue	50	50	100	200	Pomeranian	50	50	100	200
Siamese	50	49	100	199	Pug	50	50	100	200
Sphynx	50	50	100	200	Saint Bernard	50	50	100	200
American Bulldog	50	50	100	200	Samoyed	50	50	100	200
American Pit Bull Terrier	50	50	100	200	Scottish Terrier	50	50	99	199
Basset Hound	50	50	100	200	Shiba Inu	50	50	100	200
Beagle	50	50	100	200	Staffordshire Bull Terrier	50	50	89	189
Boxer	50	50	99	199	Wheaten Terrier	50	50	100	200
Chihuahua	50	50	100	200	Yorkshire Terrier	50	50	100	200
English Cocker Spaniel	50	46	100	196	Total	1846	1834	3669	7349

Table 3.2 The IIIT-OXFORD PET dataset data composition. The table lists the pet breeds (cats first) and for each the number of images in the training, validation, and test sets in the PET dataset.

Annotations In addition to breed label, every image is manually segmented into three regions: foreground (marking the pet body), background, and ambiguous (marking the pet body boundary and any accessory such as collars). Additionally, every image in the training and validation sets was annotated with a tight bounding box around the head of the pet. Figure 3.3 shows examples of these annotations.

In chapter 5, we present benchmark results on problem of Cat Vs Dog classification and on fine grained categorization using this dataset.

3.3 The MSR ASIRRA Dataset

Asirra (Animal Species Image Recognition for Restricting Access) [33, 15] is a Human Interactive Proof developed by Microsoft Research that works by asking users to identify photographs of cats and

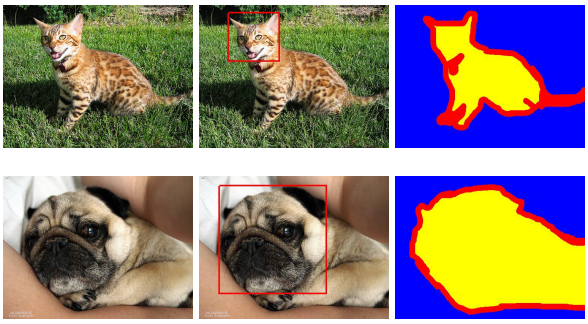
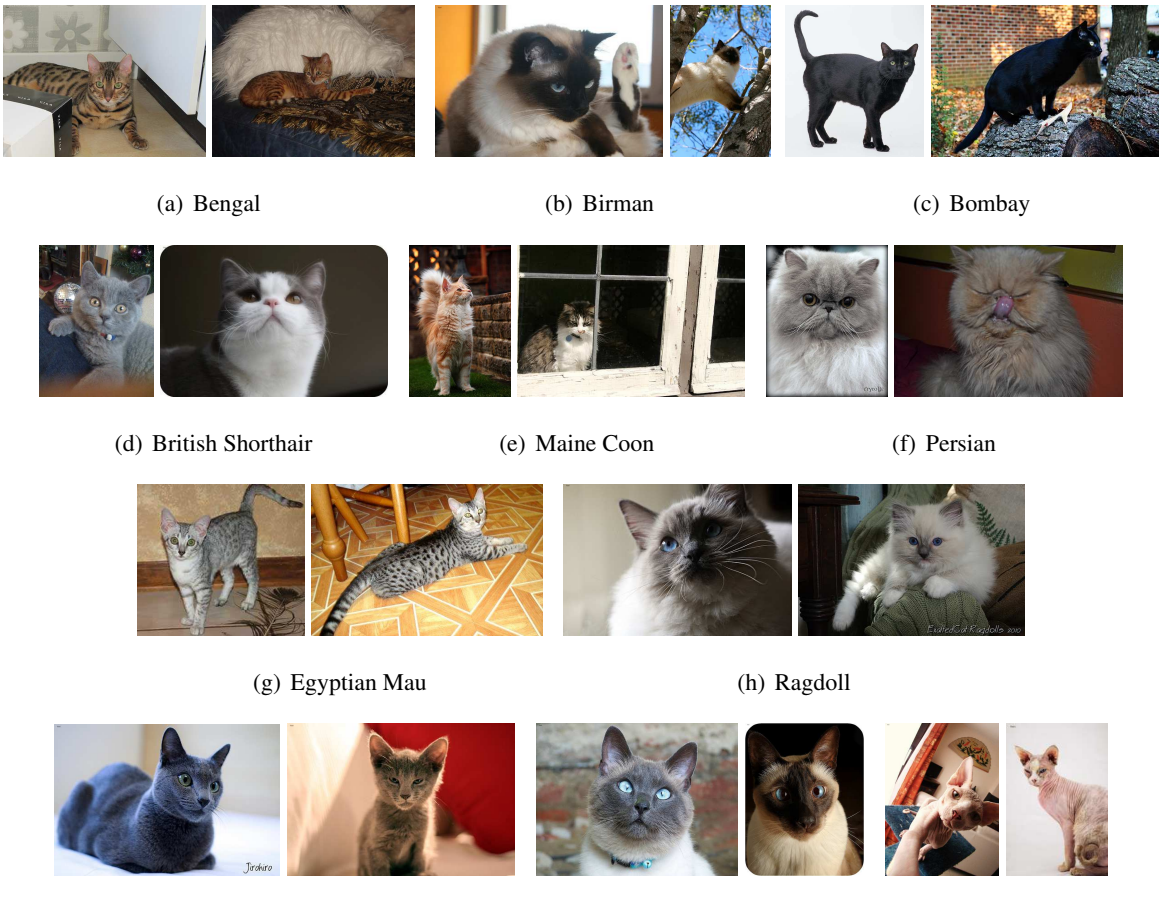


Figure 3.3 Annotations in the PET data. From left to right: pet image, head bounding box, and trimap segmentation (*blue*: background region; *red*: ambiguous region; *yellow*: foreground region).



(a) Bengal (b) Birman (c) Bombay

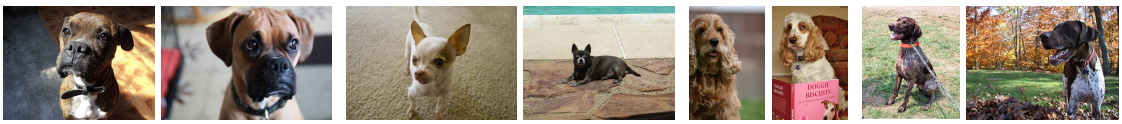
(d) British Shorthair (e) Maine Coon (f) Persian

(g) Egyptian Mau (h) Ragdoll

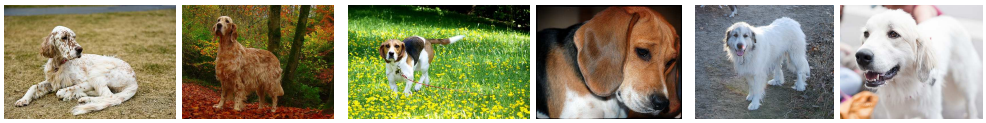
(i) Russian Blue (j) Siamese (k) Sphynx

Figure 3.4 Example cat images from the PET data. Two images per breed are shown side by side to illustrate the data variability.

(a) Am. Bull Dog (b) Am. Pit Bull Terrier (c) Basset Hound (d) Japanese Chin (e) Mini Pinscher



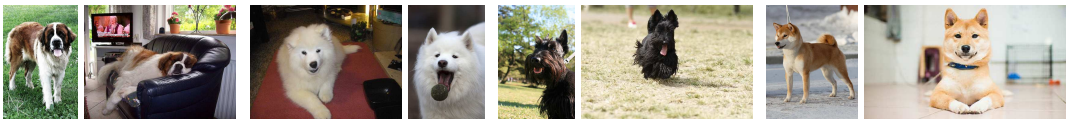
(f) Boxer (g) Chihuahua (h) Eng. Cocker (i) German Shorthaired



(j) Eng. Setter (k) Beagle (l) Great Pyrenees

(m) Havanese (n) Keeshond (o) Leonberger

(p) New Found Land (q) Pomeranian (r) Pug



(s) Saint Bernard (t) Samoyed (u) Scottish Terrier (v) Shiba Inu

(w) Staff. Bull Terrier (x) Wheaten Terrier (y) Yorkshire Terrier

Figure 3.5 Example dog images from the PET data. Two images per breed are shown side by side to illustrate the data variability.

dogs. In our work, we try to break the ASIRRA challenge using shape and appearance based models. Our models are evaluated on the ASIRRA Dataset provided for performance evaluation.

Background Web services are often protected with a challenge that is supposed to be easy for people to solve, but difficult for computers. Such a challenge is often called a CAPTCHA (Completely Automated Public Turing test to tell Computers and Humans Apart) or HIP (Human Interactive Proof). HIPs are used for many purposes, such as to reduce email and blog spam and prevent brute-force attacks on web site passwords. Today, the most common HIPs ask users to identify text that has been distorted or obscured. Unfortunately, such challenges can be difficult and frustrating for people, yet are often easily solved by computers.

Asirra works by asking users to identify photographs of cats and dogs. As shown in figure 3.6, user is presented with 12 images of cats and dogs. In order to gain access to the web service, user has to select all images containing cat from these 12 images. This task is difficult for computers, but user studies have shown that people can accomplish it quickly and accurately. Asirra is developed in partnership with Petfinder.com, a website devoted to finding homes for homeless pets. This website provides over three million images of cats and dogs, manually classified by people at thousands of animal shelters across the United States. These images form the backbone of Asirra system.

3.3.1 Dataset

To help computer vision researchers interested in "breaking" Asirra challenge, MSR made a corpus of 30,000 labelled images of cats and dogs available to the public. Each image is labeled according to species, cat or dog. This set of images is representative of the images used by the Asirra CAPTCHA, which comes from Petfinder.com. Asirra corpus is slightly biased relative to random images directly found on Petfinder. Asirra's back-end retrieves all of Petfinder's images, then filters out images that are considered unusable for example, images that are below a certain resolution, have an aspect ratio that differs too much from 1, or depict animals other than cats or dogs. The corpus offered for development is a random, unbiased sample of the images that have passed above mentioned acceptance criteria. Figure 3.7 shows example cat and dog images from this dataset.

This dataset is used for task of binary classification. Model built for this classification and its performance is discussed in chapter 5.

3.4 Evaluation

Having discussed the datasets, we now turn our attention to evaluation protocols used for assessing the performance. For our work, we use PASCAL VOC evaluation guidelines. Procedure for evaluating performance for different tasks is given below.

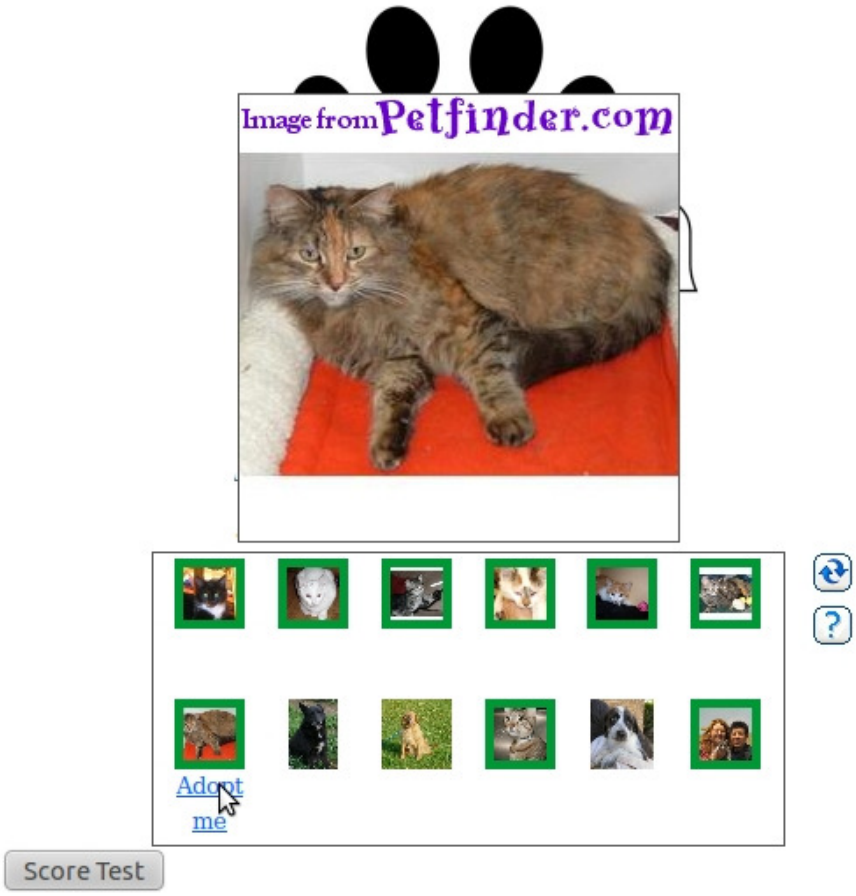


Figure 3.6 ASIRRA Interface. Asirra shows 12 images to user to gain access to a website user has to select all images having one or more cat in them.

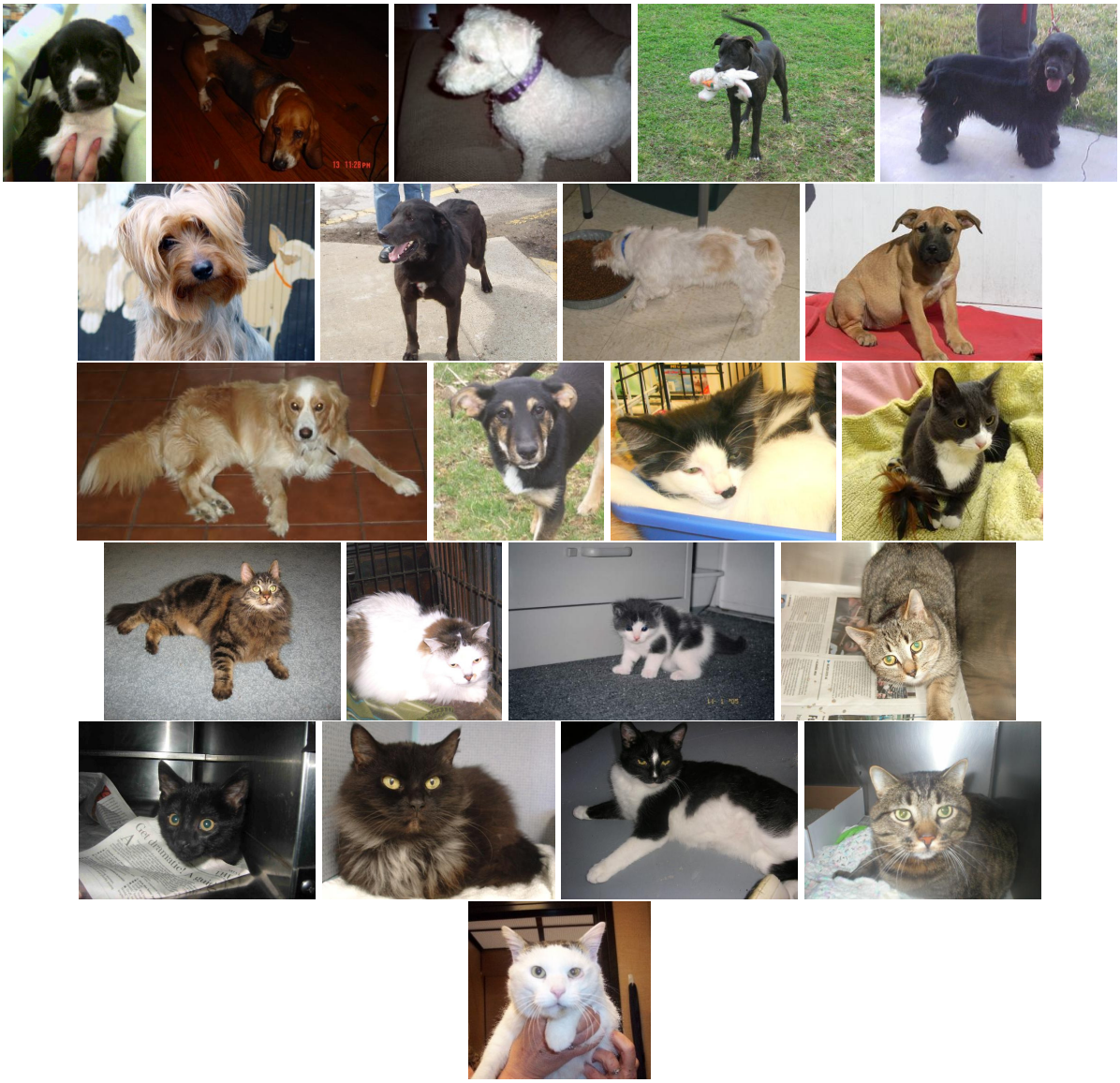


Figure 3.7 ASIRRA Dataset. Example images from MSR ASIRRA dataset.

- *classification:*

  Performance of classification tasks is judged by precision/recall curve. The principal quantitative measure is the average precision (AP). Each instance classified is assigned with a score by a classifier. These instances are ranked according to the score and precision/recall curve is plotted using maximum precision at every recall. AP is then area under this curve computed using numerical integration. Value of AP ranges from 0 to 1.

- *detection:*

  Evaluation of detection task is carried out same as that of classification. Only difference is, detections are considered true or false positives based on the area of overlap with ground truth bounding boxes. To be considered a correct detection, the area of overlap a_o between the predicted bounding box B_p and ground truth bounding box B_{gt} must exceed 50% by the formula:

$$a_o = \frac{area(B_p \bigcap B_{gt})}{area(B_p \bigcup B_{gt})} \tag{3.1}$$

- *segmentation:* The segmentation accuracy for a class is assessed using the intersection/union metric, defined as the number of correctly labelled pixels of that class, divided by the number of pixels labelled with that class in either the ground truth labelling or the inferred labelling. Equivalently, the accuracy is given by the equation:

$$\text{segmentation accuracy} = \frac{\text{true positives}}{\text{true positives} + \text{false positives} + \text{false negatives}} \tag{3.2}$$

Having looked at background work, datasets and evaluation protocols used for this thesis, we now look at problem of detecting and segmenting cats and dogs in the images in chapter 4. Later in chapter 5, we look at problem of classifying them into their respective breeds.

Chapter 4

Detection

In this chapter, we explain our model for detecting highly flexible objects such as cats and dogs. In this work, we propose to use the template-based model to *detect a distinctive part* for the class, followed by detecting the rest of the object via segmentation on image specific information learnt from that part. Our approach is motivated by two observations: (i) many object classes contain distinctive parts that can be detected very reliably by template-based detectors, whilst the entire object cannot; (ii) many classes (e.g. animals) have fairly homogeneous coloring and texture that can be used to segment the object once a sample is provided in an image.

4.1 Introduction

The vast majority of current methods for object category detection use some form of sliding window classifier. In particular, template-based models such as the Deformable Parts Model (DefPM) by [43] currently achieve state-of-the-art performance for the majority of the object classes in international benchmarks such as the PASCAL VOC 2010 [34]. The success of these methods emphasizes the importance of geometry in the description of most visual categories. But, for highly flexible and deformable objects such as cats and dogs (figure 4.1), DefPMs and other template-based models are still outperformed by a large margin by simpler bag-of-words models, which have a much weaker notion of geometry [34]. Several authors [48, 100] advocate the study of these object categories as prototypical cases for which geometric modelling is challenging. The question we address here is whether it is possible to extend template-based models such as DefPM to be competitive for these highly flexible categories as well. The key insight is that for many objects, color and texture are fairly uniform across the entire body, or vary in a manner that can be learnt; and also that many objects have a distinctive part that can be detected well with the current generation of template-based detectors, even though their overall appearance is highly variable. The idea is then to detect first a distinctive part of the category, and second, to segment the category instance primarily using image specific features learnt from that part. We call this a Distinctive Part Model (DisPM, section 4.2). For example, for a cat the head is a distinctive part and can be detected well by a template detector such as DefPM. The detected head

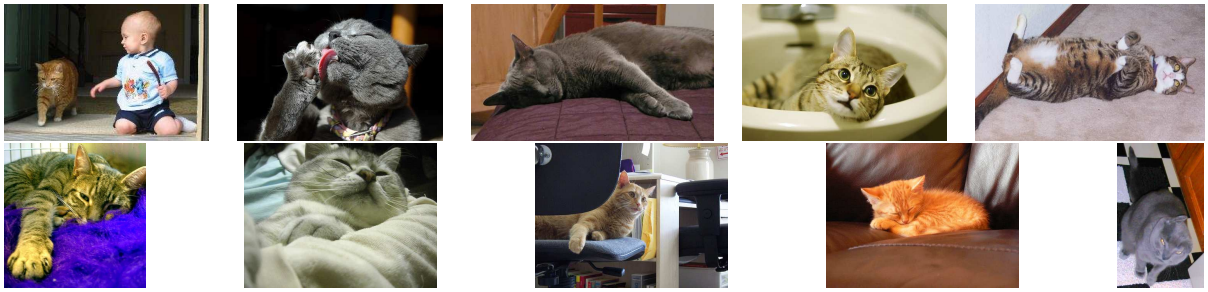


Figure 4.1 The deformable and truncated cat. Cats exhibit (almost) unconstrained variations in shape and layout. The cat examples shown here are detected by our Distinctive Part Model, but missed by the template based method of [43].

then provides the cat's fur color and texture, and, in turn, these color/texture distributions can be used to segment out the cat's body. These assumptions are satisfied for instance by numerous animal classes, such as sheep, cows, zebras, horses, elephants. A similar approach can be applied to naked humans (e.g. using face detection to learn an image specific skin color [49]), but clothing makes the model less applicable in this case. The resultant model is quite powerful in detecting such deformable animals. As can be seen by performance of our model on the PASCAL VOC 2010 detection competition [34] in section 4.3 the performance surpasses existing template models trained on the whole body by far. DisPM is in fact able to detect cats and dogs in quite variable poses, and under considerable partial occlusions and truncations (figure 4.1).

4.1.1 Related work.

Our approach extends template-based detectors such as DefPM, which, by allowing only for limited geometric variability, usually do not work well for highly deformable objects. Similarly, articulated models, such as the pictorial structures [40] typically used for human layout detection, are not appropriate for objects such as cats and dogs as they do not capture the deformation and limb occlusions that they exhibit.

Our method is also directly related to [64] and [100], that have designed and evaluated cat head detectors; section 4.3 finds DefPM much better at this task. Fleuret and Geman [48] have also proposed an interesting cat detector, but unfortunately did not evaluate their algorithm on public benchmarks, making a direct comparison difficult.

Previous work has combined object category detection and segmentation in various ways [53, 58, 61, 84, 86]. However, often the goal of these methods has been segmentation of the entire image, rather than object category detection, whilst others [20, 21, 59, 60, 69, 70, 88] have generally targeted typical views of vehicles and animals (e.g. side views of horses) that are suited to template based detectors. Their aim has not been to handle the variety in appearance and deformation that it is our goal for the new DisPM

detector. In fact, a significant difference is that DisPM restricts the use of the template detector to extract just an object part and then relying on segmentation to extend it to the whole deformable object.

Finally, this work is generally related to sliding-window object detectors. Within the window there may be be a single feature type represented, such as HOG [27] or HOG parts [43], or a bag of visual words [63], or a grid or pyramid of visual words [45, 68], or a combination of such features and kernels [57, 94]. In the recent PASCAL VOC 2010 object detection competition [34] all the top methods were of this kind. There are a number of methods for object detection that start from bottom up segmentation, rather than sliding/jumping windows [13, 56, 66], but they are yet to be competitive with the window based detectors.

4.2 The distinctive part model

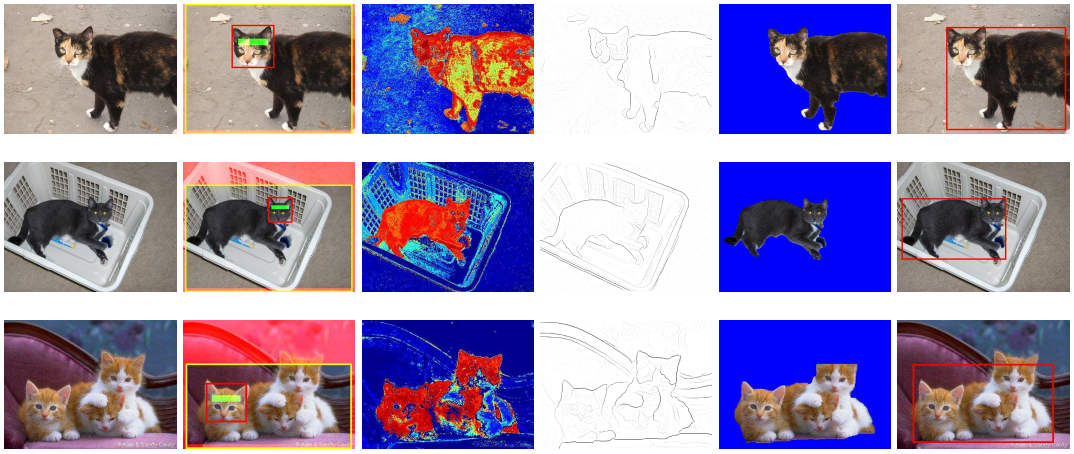
The DisPM extends template-based models to the detection of highly deformable object categories. Consider the case of cats, which we will use as our running (actually sitting) example for describing the new model: extreme articulations, atypical viewpoints, and partial occlusions induce variations of the appearance of a cat that cannot be captured by a template-based model. This is true even for models such as the DefPM detector that account explicitly for deformations of the template.

The DisPM works around this problem by detecting first a stable and distinctive object part, such as the cat head, for which a template-based detector is appropriate. It then uses the detected part to initialize and constrain the segmentation of the rest of the object. DisPM is therefore composed of three elements, illustrated in figure 4.2: (i) a template-based detector of the distinctive object part, (ii) a model of the object body appearance (color or texture), and (iii) a segmentation algorithm.

The next three sections describe in detail the three components of the model. For the template-based detector (i) we use the DefPM model based on the implementation publicly available from the author's website (section 4.2.1). For the local appearance model (ii) we model colors by histograms in RGB space, along with an object boundary detector to aid segmentation (section 4.2.2). For the segmentation algorithm (iii) we use the standard graph cut model of Boykov *et al.* [24] (section 4.2.3). Since the appearance model is learned from the object region itself (starting from the distinctive part), graph cut and estimation of the appearance model are alternated to refine the segmentation result (GrabCut [85]).

4.2.1 Part model

The distinctive object part is detected by means of the DefPM. As will be shown in section 4.3, this model is excellent for structures that are relatively stable, such as, for example, the face of a cat, but is relatively poor for highly deformable objects, such as the cat body. The detected part is used to determine an *image-specific* color model for the cat, and also to predict a (maximal) bounding box for the entire cat.



(g) Original image (h) Head detection (i) Foreground posteriors (j) Berkeley edge detection (k) Foreground segmentation (l) Output bounding box

Figure 4.2 Overview of the model. A distinctive part, the head in this case, is detected using the DefPM model [43]. (b) The detected part ROI (red rectangle) is used to define a search region for the object (yellow rectangle), and also seeds the foreground color distribution (green rectangular region). The background color distribution is learnt from the red area. (c) the foreground posterior, computed using the seed and background data (red is high, blue is low probability). These posteriors form the unary term of the energy function used in segmentation. The pairwise terms use the Berkeley edge detector response (d). A graph cuts binary optimization gives the foreground segmentation (e). The detection result is a tight bounding box around the foreground segment (f).

The DefPM detector is a mixture of templates, each of which is a collection of parts connected by springs. Parts are described by linear filters on top of low level features such as HOG [27] and the model is learned by means of a latent SVM. See section 4.3 for further details and figure 4.2(b) for example detections.

4.2.2 Whole object model

The object appearance model captures the material of the object (color) and the object discontinuities (edges). For the object color, there are two source of information that can be used. First, some colors cannot belong to any of the object instances (e.g. there are no green, blue, or purple cats), which is used to construct a *color prior for the category*. This is learned from the trimap object segmentations (figure 3.2) by computing color histograms of the foreground (cat) and background (non-cat) regions. Second, the color of the specific object *instance* being detected, and of the background scene in which it is found, can be estimated from the distinctive part. For cats and dogs, the head provides a cue on

the color of the fur, and image pixels far enough from the head are used to estimate the color of the background.

Category color prior. Colors are modelled by means of histograms. We use a relatively high dimensional histogram $\mathbf{h} \in \mathbb{R}^{32\times32\times32}$ but smooth it by a small Gaussian kernel (of isotropic standard deviation $\sigma = 0.025$) in order to reduce the variance of the estimator. The global foreground/background color histograms $\mathbf{h}^0_{\text{fg}}, \mathbf{h}^0_{\text{bg}}$ are obtained from all the foreground/background regions in the training set.

Instance-specific color. The distinctive part of the object is used to obtain an instance-specific foreground $\mathbf{h}_{\text{fg}}$ and background $\mathbf{h}_{\text{bg}}$ color models. The foreground color is estimated by sampling the pixels contained in the *foreground seed*. The seed is a rectangular sub-region of the distinctive part that is contained in the foreground region with very high probability in the training data. For instance, the foreground seed of cats roughly corresponds to the forehead. The background color is estimated from the pixels that are outside a *maximal bounding box*, i.e. a bounding box that contains almost surely the entire object. The maximal bounding box is obtained by aligning and scaling a template box to the rectangle of the distinctive part detection. The dimensions of the template itself are learned by requiring it to be the smallest box that contains 99% of the object pixels for all training images. To handle the case where no part of the image is inside the maximal bounding box, a thin strip of pixels around the image (20 pixels wide) is always included to estimate the background color. Examples of the seed and of the bounding box are shown in figure 4.2(b) (these regions will be used in section 4.2.3 to further constrain the segmentation geometrically).

Foreground and background posteriors. In the section our intention is to find a posterior probabilities of a pixel from a given image beloging to object and background. Let $\mathbf{x}$ be an image and $\mathbf{y}$ be a partition of the image into foreground (object) and background components. In particular, let $\mathbf{x}_i \in \mathbb{R}^3$ denote the color of the i-th pixel (in RGB space) and let y_i be equal to $+1$ if the pixel belongs to the object and to -1 otherwise. Given the color histogram $\mathbf{h}_{\text{fg}}, \mathbf{h}_{\text{bg}}, \mathbf{h}^0_{\text{fg}}, \mathbf{h}^0_{\text{bg}}$, we can define three likelihoods:

$$p(\mathbf{x}|y=+1,\text{fg}) = \mathbf{h}_{\text{fg}}(\mathbf{x}), \qquad p(\mathbf{x}|y=-1,\text{bg}) = \mathbf{h}_{\text{bg}}(\mathbf{x}),$$
$$p(\mathbf{x}|y=+1,\text{fg}^0) = \mathbf{h}^0_{\text{fg}}(\mathbf{x}), \qquad p(\mathbf{x}|y=-1,\text{bg}^0) = \mathbf{h}^0_{\text{bg}}(\mathbf{x}),$$

First all the pixels belonging to foreground and background from training images are quantized to form two histograms. Similar histograms are built for a given test image from the head detection region and predicted bounding box region. Then given a pixel, its appropriate bin in the histogram gives likelihood of pixel coming from foreground and background. By assuming $P[y=+1] = P[y=-1] = 1/2$, these are combined into two posteriors probabilities by using Bayes' theoram.

$$p_1(y|\mathbf{x}) = \frac{p(\mathbf{x}|y=+1,\text{fg})}{p(\mathbf{x}|y=+1,\text{fg}) + p(\mathbf{x}|y=-1,\text{bg})}, \tag{4.1}$$

$$p_2(y|\mathbf{x}) = \frac{p(\mathbf{x}|y=+1,\text{fg}^0)}{p(\mathbf{x}|y=+1,\text{fg}^0) + p(\mathbf{x}|y=-1,\text{bg}^0)}. \tag{4.2}$$

Above equations give the posterior probability of a given pixel belonging to foreground region. Since two different likelihoods are used, in the form of local and global information, two different posteriors are obtained. fg and bg are refer to object and background pixels in the given image respectively. While fg^0 and bg^0 refer to object and background pixels from all training images (global model). These posterior probabilities are used to form data terms in equation 4.4. For a pixel belonging to object, probability of its color being observed in object foreground colors should be high. Similarly, pixel belonging to background should have high background posterior probability. The first one discriminates between the color of the object instance and the color of its surrounding (as estimated from the seed and the maximal bounding box), and the second one between that of the object and of generic clutter. The latter helps eliminating impossible colors (e.g. green cats) that may not be sampled outside the maximal object bounding box. These two are combined into a unique posterior by additive combination ($p(y|\mathbf{x}) \propto c_1 p_1(y|\mathbf{x}) + c_2 p_2(y|\mathbf{x})$ where the weights c_i are learnt from validation data (and have the values $c_1 = 1/10, c_2 = 9/10$). Example foreground posteriors, $p(y = 1|\mathbf{x})$, are shown in figure 4.2(c).

Modelling edges. In addition to color, the model also uses an edge detector in order to further improve the quality of the final object segmentation. The edge map will be used to encourage the segmentation boundaries to match discontinuities of image edges. In this work we leverage on the powerful Berkeley PB edge detector [73]. Compared to other detectors such as Canny, PB is designed to suppress intensity discontinuities which correspond to texture rather than actual object boundaries. See figure 4.2(d).

4.2.3 Segmentation model

Once the distinctive object part has been detected, it must be extended to a segmentation of the entire object (see figure 4.2(e)). As we expect the object to be highly deformable but to have a distinctive material, this can be achieved by a well designed segmentation algorithm.

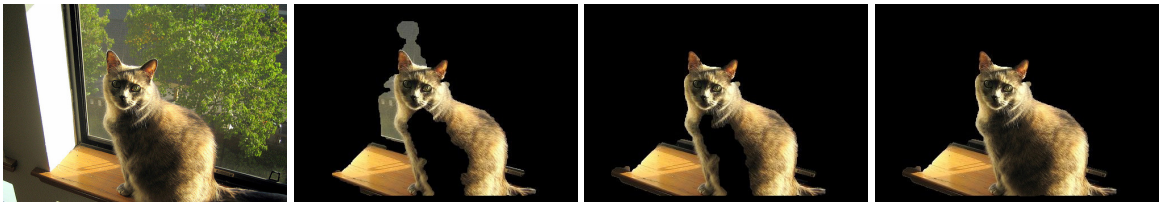
For segmentation we use a graph cut [24] based energy minimization formulation. The cost function is given by

$$E(\mathbf{x}, \mathbf{y}) = -\sum_i \log p(y_i|x_i) + \sum_{(i,j)\in\mathcal{E}} S(y_i, y_j|\mathbf{x}) \tag{4.3}$$

This is the standand graphcut enegry minimization formulation. Energy is minimized using max-flow/min-cut technique. First term on the right hand side is called 'unary term/data term' and second term is called 'pairwise term/edge term'. The data term is modelled as a likelyhood of a pixel belonging to foreground or background while parwise term encourages similarities between neighbouring pixels. The edge system $\mathcal{E}$ determines the pixel neighbourhoods and here is the standard eight-way connectivity scheme. The pairwise potential $S(y_i, y_j|\mathbf{x})$ favours neighbouring pixels to have the same label unless a PB edge separates them:

$$S(y_i, y_j|\mathbf{x}) = \gamma \exp(-e_j(\mathbf{x})/\beta) \tag{4.4}$$

where $e_j(\mathbf{x})$ is the PB edge intensity at pixel j and $\beta = \langle e_j(\mathbf{x}) \rangle$ is the average edge intensity in the image. Note that the edge is measured only at pixel j, as defined by the edge system $\mathcal{E}$ (here j is the



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

Figure 4.3 Effect of various parameters on segmentation.

(a) Original image. (b) Output of GrabCut segmentation described in [24]. (c) Effect of adding berkeley edge detector response as pairwise terms. A sharper cut is obtained as a result of using edge detector response. (d) Effect of posterior probabilities in data terms of the segmentation model. Global posterior probabiltie helps in modelling foreground model by considering colors which were missed by local foreground model.

pixel more on the right/south).The parameter γ controls the importance of edge terms in the energy calculations and is learnt experimentally on the validation data.

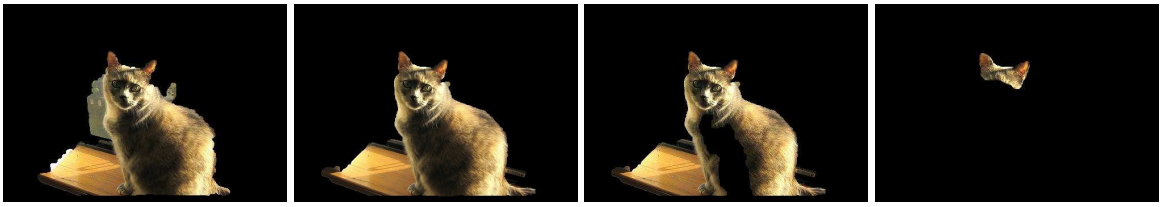
The distinctive part detection is used to fix the values of some labels $\mathbf{y}$ (clamping) as follows: (i) the foreground seed region must be labelled as foreground, and (ii) the region outside the maximal bounding box must be background. These two regions were defined above in section 4.2.2.

The segmentation is defined as the minimizer $\arg\min_{\mathbf{y}} E(\mathbf{x}, \mathbf{y})$ of the energy using graph cut. In fact, since the color of foreground and background can be estimated more accurately as a better segmentation of the object becomes available, GraphCut is alternated to re-estimate the color model, in the manner of GrabCut [85]. In section 4.3 we show that initializing from the posteriors of section 4.2.2, yields a substantial improvement in detection performance over simply initializing from the clamped regions.

Cleaning-up and detection. Given the segmentation result from GrabCut, this is cleaned-up by preserving only the connected foreground component that intersects with the distinctive part and discarding the others. The final object bounding box is estimated as the smallest box that fully contains the segmented foreground region (see figure 4.2(f)). The detector score is obtained from a combination of the DefPM score and size of the distinctive part detection.

4.2.4 Examples

In this subsection, we provide qualitative analysis of different techniques used for segmentation. First we show effect of variation in modelling such as berkeley edges, posterior probabilities etc. described in 4.2.2 and then we will describe effect of γ term in equation 4.4.



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

Figure 4.4 Effect of γ on segmentation.

(a) (γ =0.1) Lower γ value results in adding more information to foreground. (b) (γ = 5) Increase γ value helps in removing background pixels. (c) (γ = 50) Further increase γ results in algorithm providing a cut at sharp edge responses (d) (γ = 300). Very high values of γ force algorithm to providing a cut at weaker edge responses.

Effect of modelling variations. Figure 4.3 shows effect of modelling techniques on on the segmentation output. Figure 4.3(a) shows the original image considered for segmentation. Figure 4.3(b) shows the output of standard GrabCut technique described in [24] as per equation 4.3. When berkeley edge detector responsed are used as edge term in the same equation, sharp cuts are observed as shown in figure 4.3(c) removing additional area from the foreground. As we can see the slightly darker area on the cat's body is still not marked as foreground. Color models built from the seed information obtained from distinctive part detection are inadequate to model the change in the fur color. To provide sufficient color information and to produce better segmentation, we model colors based on colors learnt from training images. The posterior posterior probabilities obtained by using equations 4.1 and 4.2 are used to model data terms in equation 4.4. Resultant segmentation is shown in figure 4.3(d). It can be seen that additional color information helps in forming better color models and hence better segmentation.

Effect of modelling variations. Figure 4.4 shows effect of changing contribution of edge terms from equation 4.4. This can be done by chaning values of parameter γ. Lower values of gamma result in algorithm providing cuts only at very strong edge responses. As the value is increased, cut is produced at weaker edge detector response. Higher values of gamma force the cut to take place even at very low edge detector responses. In practice, one has to set optimal value of gamma to achieve better performance. In our work, this value is set experimentally by analysing performance of the system on validation data.

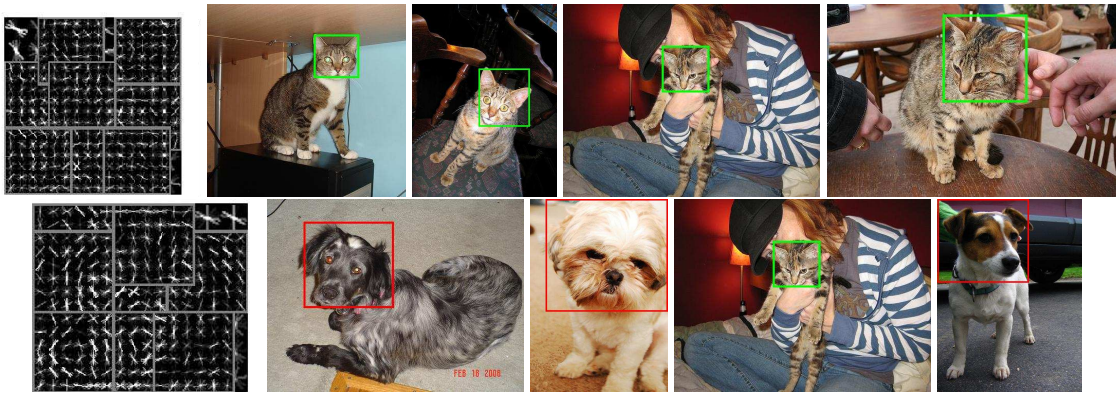


Figure 4.5 Distinctive part detector. First row: The DefPM model [43] for the cat head, which is used as a distinctive part for this object, and example detections. Second row: the same for dog.

4.3 Results

Following the PASCAL VOC best practices [34], the various components of the model are evaluated in detail on the PASCAL VOC 2010 train/validation sets and overall results for the complete model are given on the test set to allow for a direct comparison with other published methods.

The performance of a detector is evaluated in term of the Average Precision (AP) of the ranked list of detections, where a detection is considered to be correct if its overlap ratio with a ground truth bounding box is at least 0.5 and if it is not a duplicate (see [35] for details).

Learning the distinctive part. The distinctive part annotations are used to train a DefPM model for the part (figure 4.5) with one aspect, eight high resolution parts, and a low resolution one (root filter). The low level image features are HOG [27, 43] (capturing shape) and LBP [79] (capturing texture). The DefPM detector supports multiple components, but in our experiments we use a single one as we found empirically that this worked better in our case. Figure 4.5 shows examples of the detected cat/dog heads with variations in pose, appearance, and size.

Precision-recall curves for the DefPM detector for the cat heads in the VOC 2010 validation data are given in figure 4.6(b). With the standard PASCAL VOC overlap ratio of 0.5, the detector AP is 45% with HOG features only, and this improves to 49% when the LBP features are added. Since the DisPM uses the distinctive part as a seed to obtain a segmentation for the whole object, a less strict (than 0.5) overlap ratio often suffices for this purpose (as will be seen below). Thus it is interesting to note that for a (looser) overlap ratio of 0.2, the AP of the head detector is 61% with a recall of 80%. The recall-precision curve for this overlap ratio is also shown in figure 4.6(b). The DefPM performs much better than alternative cat head detectors available in the literature. Specifically, when trained and tested on the VOC 2007 cat heads, DefPM achieves an AP of 54.6%, while the detector of Zhang *et al.* [100]

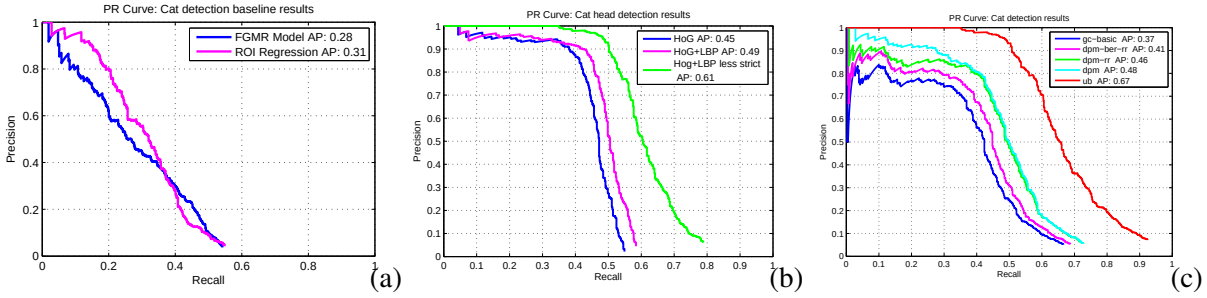


Figure 4.6 Performance of the model components for cat detection on the VOC2010 Validation data. (a) Baseline cat ROI detection results – the DefPM model (trained on the whole object) and regression on the head detections. (b) Head ROI detection results using the DefPM model (trained only on heads) with and without LBP. (c) Components of the DisPM model: `gc-basic` (GrabCut initialized from the clamped regions, without Berkeley edges nor reranking), `dpm-ber-rr` (as previous case, but GrabCut with the posterior from Sect. 4.2.2), `dpm-rr` (with Berkeley edges), `dpm` (with reranking). Finally, *ub* shows the upper bound on the detection AP.

obtains 34.4% with the same data. The detector of Laptev [64] obtains 18.7% on the VOC 2007 test data (when trained on the VOC 2006 training data).

Whole object detectors: baselines. The first baseline is the standard DefPM model trained to detect the whole object. For cats, training on the VOC training data and testing on the validation data gives an AP of just 29% (figure 4.6(a)). Based on the PASCAL VOC 2010 results, the performance of the newest DefPM version (which is not yet available to the public) is, on the VOC 2010 test data, about the same (31.8%). This level of performance is relatively poor compared to other classes (e.g. the performance of the DefPM detector on the VOC vehicles is around 50% AP). The model does not seem capable of capturing the variability of the cat *bodies*. To verify this, consider as a second baseline a simple head-to-cat regressor. This regressor is obtained by computing the average ratios between the size of the cat head and the margins between the cat head bounding box and the bounding box of the whole cat. These ratios are then used to predict a bounding box for the cat given a novel head detection. This simple head-to-cat regressor has 31.1% AP, which *already exceeds the performance of the DefPM detector trained on the whole cat*.

Whole object detectors: upper-bounds. Given the detections for the distinctive part, it is easy to compute an upper bound for the performance of the DisPM by mapping each part detection to its corresponding ground-truth object bounding box, if there is one (we say that the part corresponds to the object if more than 50% of the area of its bounding box is included in the object bounding box). In this

way, one obtains a cat detector with AP of 67% (figure 4.6(c)). While this is an ideal result, it is worth noting that the performance is more than twice that of the standard DefPM detector.

Postprocessing. All the top methods in the PASCAL Challenge [34] rerank detections based on global image cues and other statistics. In our case, the final scoring for a candidate detection is obtained by combining, by means of a linear SVM, the following seven features. The first feature is the DefPM score for the distinctive part; the second and third features are the output of image-level bag-of-word classifiers [97], trained to detect cats and dogs respectively (the inclusion of both animals helps disambiguating between them, similarly to [43]); the fourth and fifth features are also two global cat and dog scores, obtained as the maximum response of the DefPM detectors within each image. The last two features are the size of the detection relative to the image size and its aspect ratio, which capture weak pose information. Similar to what observed by [43], post processing improves the results by 2-3% AP points.

Results on cats and dogs. Having defined and measured upper and lower bounds (from the baselines), we now turn to the performance of the DisPM itself. This is shown in figure 4.6(c), where the contribution of the various components of the model are detailed *for the VOC 2010 validation data*: (i) the most basic (damaged) form of the model is to segment using GrabCut but with the foreground and background regions defined only by the clamped areas, and without using the Berkeley edge detector (instead the pair wise term (4.4) measures neighbouring image intensity differences directly as in [24]). This is shown as `gc-basic` and has an AP of only 37%. Adding in the posterior computation from section 4.2.2 to initialize the GrabCut (`dpm-ber-rr`) increases the AP to 41%. A further increase is obtained by using Berkeley edges instead of image differences in (4.4), and the performance reaches 46% (`dpm-rr`). Finally, the full DisPM including the re-ranking step (`dpm`) achieves 48%, which surpasses the baselines (DefPM and regressor) by about 20% AP. A similar analysis holds for dogs, for which the final AP of the DisPM detector is 36%, which also about 20% better than both the baselines (the upper bound being 51%). While the performance of the DisPM exceeds both the baselines by a wide margin, there is still a significant gap to the upper bound. We describe the reasons for this gap below, and in section 4.4 discuss how the gap can be reduced. Examples detections are shown in figures 4.7 and 4.8 for cats and dogs respectively.

Finally, on the VOC 2010 *test* data the performance of the cat and dog detectors are respectively 45.3% and 36.8%, both of which improve significantly on the latest DisPM results (31.8% and 21.5%) and are very close to the state of the art (47.7% and 37.2%) [34].

4.4 Summary

Given the current performance of the DisPM detector, we can conclude that starting from a distinctive part it is possible to detect far more and varied instances than can be obtained with a whole body template

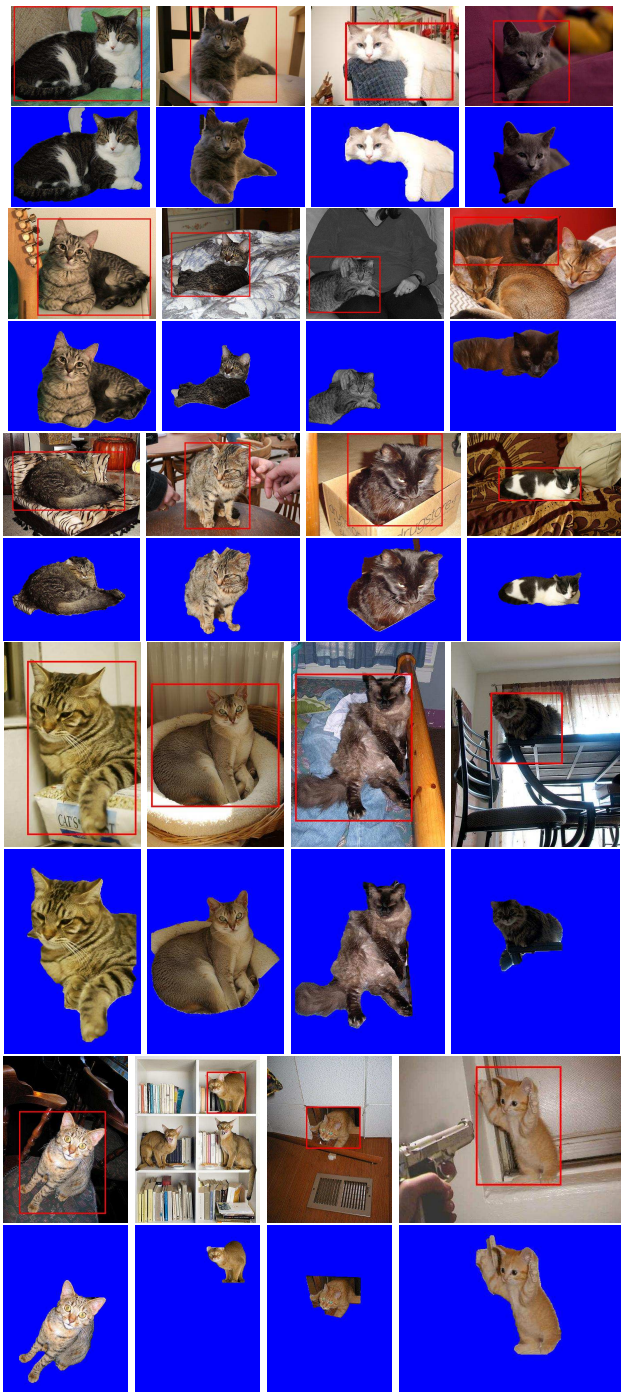


Figure 4.7 Cat detections. A sample of the detections and segmentations produced by the DisPM detector (VOC 2010 validation data). It can be seen that cats are successfully detected despite having different fur colors, and appearing in a variety of postures etc.

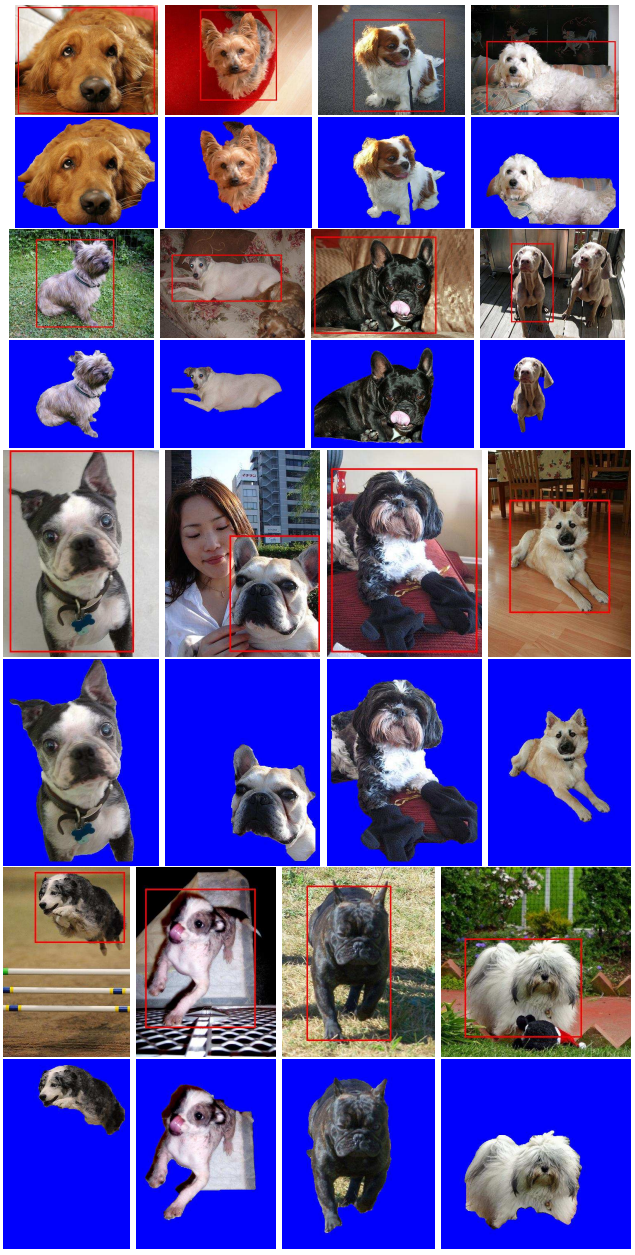


Figure 4.8 Dog detections. A sample of the detections produced by the DisPM detector (VOC 2010 validation data).

detector. Indeed, the DisPM detector is comparable to the state of the art. This is remarkable – a simple model using only two feature types (HOG and LBP for the distinctive part) and image specific color, matches the performance of algorithms using multiple features, including pyramids and kernels (e.g. the PASCAL VOC 2010 winner for this class). Further improvements to the model, mainly in segmentation, needs to be researched so as to outperform the BOW based methods.

Chapter 5

Classification

Having dealt with problem of detecting cats and dogs in the images, in this chapter we shift our attention to classifying them according to their species and breed. Given an image, we first detect the object in the image and then classify the objects by their breeds. Performance of our methods is tested on our IIIT-OXFORD PET dataset and MSR Asirra dataset. Our method significantly outperforms previously known attempts to break Asirra system.

5.1 Introduction

A model for pet breed discrimination (Sect. 5.2) will be discussed in this chapter. The model captures both shape (by a deformable part model [44, 95] of the pet face) and texture (by a bag-of-visual-words model [90, 26, 67, 99] of the pet fur). Unfortunately, current deformable part models are not sufficiently advanced to represent satisfactorily the highly deformable bodies of cats and dogs; nevertheless, they can be used to extract reliably stable and distinctive *components* of the body, such as the pet face. This observation is validated by demonstrating that two deformable part models, one for the cat faces and one for the dog faces, are sufficient to break the Asirra test (Sect. 5.3.1), that uses the ability of discriminating between cats and dogs to tell humans from machines. Two natural ways of combining the shape and appearance features are then considered and compared: a flat approach, in which both features are used to regress the species and the breed simultaneously, and a hierarchical one, in which the species is determined first based on the shape features alone, and then appearance is used to predict the breed conditioned on the species.The model is validated experimentally on the task of discriminating the 37 pet breeds (Sect. 5.3), obtaining very encouraging result, especially considering the toughness of the problem.

Related work. Authors have often focused on cats and dogs as example of highly deformable objects for which recognition and detection is particularly challenging [65, 100, 48]. Breaking the Asirra test by machine learning tools was attempted in [51], but obtaining results significantly inferior to the ones pre-

sented here. Other works addressing fine grained object categorization include discriminating between flower [76, 92, 78] and bird species [62, 96, 25].

5.2 A model for breed discrimination

The breed of a pet affects its size, shape, and fur type and color. Since it is not possible to measure the pet size from an image without an absolute reference, the models focus on capturing the pet shape (Sect. 5.2.1) and the appearance of its fur (Sect. 5.2.2). Given an image, location of the pet is determined by using our detection method discussed in chapter 4. Pet head detector based on deformable part model of [44] is used to model the shape. Appearance model is based on SIFT BOW Approach. SVM and MKL approaches are used for classification.

5.2.1 Shape model

To represent shape, we use the deformable part model of [44]. In this model, an object is given by a root part connected with springs to eight smaller parts at a finer scale. The appearance of each part is represented by a HOG filter [28], capturing the local distribution of the image edges; inference (detection) uses dynamic programming to find the best trade-off between matching well each part to the image and not deforming the springs too much.

While powerful, this model is insufficient to represent the flexibility and variability of a pet body. This can be seen by examining the performance of this detector on the cats and dogs in the recent PASCAL VOC 2010 challenge data [34]: The deformable parts detector [44] obtains an Average Precision (AP) of only 31.8% and 13.5% on cats and dogs respectively [34]; by comparison, an easier category such as bicycle has AP of 54% [34]. However, in the PASCAL VOC challenge the task is to detect the *whole body* of the animal; we found that, instead, deformable part models are much better at detecting certain stable and distinctive *components* of the body. In particular, the head annotations included in the IIIT-OXFORD PET data are used to learn a deformable part model of the cat faces, and one of the dog faces ([48, 65, 100] also focus on modelling the faces of pets). Sect. 5.3.1 shows that these models are in fact very good.

5.2.2 Appearance model

To represent texture, we use a bag-of-words [26] model. Visual words [90] are computed densely on the image by extracting SIFT descriptors [72] with a stride of 6 pixels and at four scales, defined by setting the width of the SIFT spatial bins to 4, 6, 8, and 10 pixels respectively. The SIFT features have constant orientation (*i.e.* , they are not adapted to the local image appearance). The SIFT descriptors are then quantized based on a vocabulary of $4{,}000$ visual words. The vocabulary is learned by using k-means from features randomly sampled from the training data. In order to obtain a descriptor for the

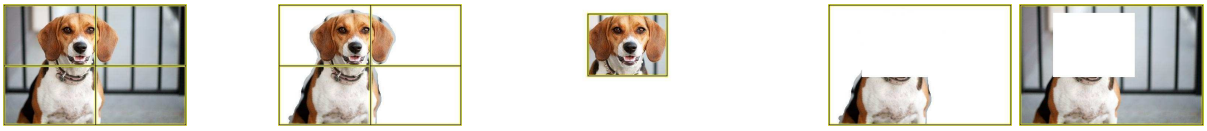


Figure 5.1 Spatial histogram layouts. From left to right: *image, body, head, body-head* and *image-head* layouts.

image, the quantized SIFT features are pooled into a spatial histogram [67], which has dimension equal to $4{,}000$ times the number of spatial bins.

Different variants of the spatial histograms can be obtained by placing the spatial bins in correspondence of particular geometric features of the pet. We consider five different types of layouts for the spatial bins and divide them into three groups, depending on the type of geometric information that they require. These groups and the layouts therein are described next and in figure 5.1:

No geometric features. This group includes only the *image layout*, which consists of five spatial bins organized as a 1×1 and a 2×2 grids covering the entire image area, as in [67].

Head bounding box. This group includes the *head layout*, a spatial bin matching the head bounding box (obtained from the deformable part model of the pet faces), and the *image-head layout*, which consists of a bin covering the entire image minus the head bounding box.

Body segmentation. This group includes the *body layout*, obtained by intersecting the five bins from the image layout with the foreground region (obtained either from the ground truth annotations or automatically by using a segmentation algorithm described next), and the *body-head layout*, which is the same as the image-head layout but restricted to the foreground region.

The segmentations needed for the last group of layouts are computed by using the grab-cut algorithm [85]. Our grab-cut implementation describes the foreground (pet) and background regions by corresponding appearance models (each of which is a Gaussian mixture model of the color and gradient orientation at each pixel) and an edge model (based on the Berkeley edge detector [73]). The foreground appearance model is initialized on the pet head bounding box (obtained from the pet face detector), and the background model from a strip of ten pixels around the image.

5.2.3 Classification

Two approaches are considered for classification, using shape and appearance information individually and combining them in Multiple Kernel Learning framework. Head detection scores of two seperate detectors trained to detect cats and dog heads are used for binary classification based on shape. For appearance model, Histograms computed in 5.2.2 are then l^1 normalized and used in a support vector machine (SVM) based on the exponential-χ^2 kernel [99] for classification. One versus rest multiclass

n.	shape	appearance					classif. accuracy (%)				
		ROI			segmentation		species	breed (S. 5.3.3)		both (S. 5.3.4)	
		img.	head	img.−hd.	body	body−hd.	(S. 5.3.1)	cat	dog	hier.	flat
1	✓	–	–	–	–	–	94.23	NA	NA	NA	NA
2	–	✓	–	–	–	–	81.94	53.84	39.08	NA	41.73
3	–	✓	✓	✓	–	–	87.28	63.88	53.87	NA	53.90
4	–	✓	✓	–	✓	✓	89.77	67.66	56.80	NA	56.99
5	–	✓	✓	–	GT	GT	91.10	69.79	60.07	NA	60.75
6	✓	✓	–	–	–	–	95.36	53.57	43.52	41.73	44.85
7	✓	✓	✓	✓	–	–	95.81	64.53	56.37	54.88	57.16
8	✓	✓	✓	–	✓	✓	95.96	66.90	58.91	57.87	59.42
9	✓	✓	✓	–	GT	GT	95.80	69.23	62.09	60.74	62.76

Table 5.1 Comparison between different models. The table compares different models on the three tasks of discriminating the species, the breed given the species, and the breed and species of the pets in the IIIT-OXFORD PET dataset (Sect. 3.2). Different combinations of the shape features (deformable part model of the pet faces) and of the various appearance features are tested (Sect. 5.2.2). A dash – means that a feature is not used, a checkmark ✓means that the feature is used and that its computation does not use ground truth information, and a GT symbol means that the feature is used with ground truth segmentations.

classification technique is used for 37 class classification problem. For combining shape and appearance information, detector scores are used as features to form the shape kernel. This was then combined with appearance kernel by simple addition and SVM was used for solving binary and multi class problems.

5.3 Experiments

The models are evaluated first on the task of discriminating the species of the pets (Sect. 5.3.1), then on the one of discriminating their breed given the species (Sect. 5.3.3), and finally discriminating both the species and the breed (Sect. 5.3.4). For the latter task, both hierarchical classification (*i.e.* , determining first the species and then the breed) and flat classification (*i.e.* , determining the species and the breed simultaneously) are evaluated. Training uses the PET train and validation data and testing uses the PET test data. All these results are summarized in table 5.1 and further results for species discrimination on the Asirra data are reported in Sect. 5.3.1. Failure cases are reported in figure 5.3.

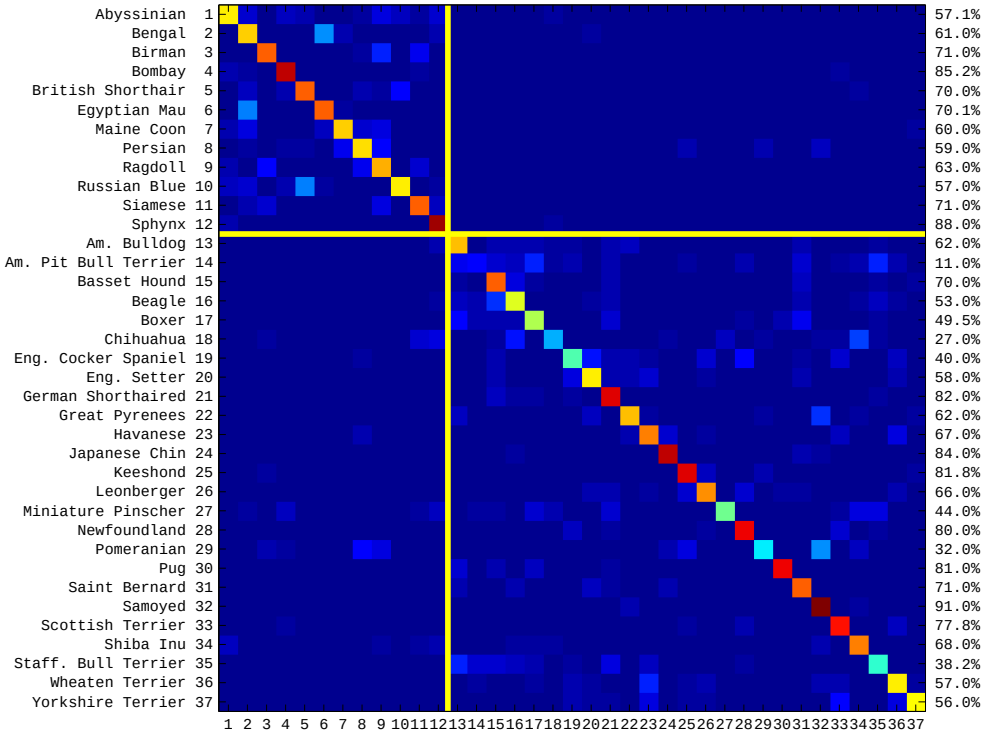


Figure 5.2 Confusion matrix for breed discrimination. The vertical axis reports the ground truth labels, and the horizontal axis to the predicted ones (the upper-left block are the cats). The matrix is normalized by row and the values along the diagonal are reported on the right. The matrix corresponds to the breed classifier using shape features, appearance features with the *image, head, body, body-head layouts* with ground truth segmentations, and a 37-class SVM. This is the best result for breed classification, and corresponds to the last entry of row number 9 in table 5.1.

5.3.1 Species discrimination

This section evaluates the models on the task of discriminating the species of a pet (*i.e.* , cat vs dog classification). Each model uses a different combination of features:

Shape only. The maximum response of the cat face detector (Sect. 5.2.1) on an image is used as an image-level score for the class cat. The same is done to obtain a score for the class dog. Then a linear SVM is learned to discriminate between cats and dogs based on these two scores. The classification accuracy of this model on the PET test data is 94.69%.

Appearance only. Spatial histograms of visual words are used in a non-linear SVM to discriminate between cats and dogs, as detailed in Sect. 5.2.2. The accuracy depends on the type of spatial histograms considered, which in turn depends on the layout of the spatial bins. On the PET test data, the *image layout* obtains an accuracy of 81.94%; adding to the latter the *head* and *image-head layouts* yields an accuracy of 87.28%. Combining the *image, head, body*, and *body-head layouts* improves the results by a further 2.5%, which becomes 3.8% if the ground-truth segmentations are used in place of the ones estimated by grab-cut (Sect. 5.2.2). This progression indicates that the more accurate is the localization of the pet body, the better is the classification accuracy.

Shape and appearance. The appearance and shape information are combined by summing the exp-χ^2 kernel for the appearance part (Sect. 5.2.2) with a linear kernel on the cat scores and a linear kernel on the dog scores. The combination boosts the performance by an additional 4%, yielding approximately 96% accuracy, with all the variants of the appearance model performing similarly.

5.3.2 Cracking Asirra

MSR proposed the problem of discriminating cats from dogs as test to tell humans from machines, and created the *Asirra* test ([33], figure 3.7). The assumption is that, out of a batch of twelve images of pets, any machine would predict incorrectly the species of at least one of them, while humans would make no mistakes. The Asirra test is currently used to protect a number of web sites from the unwanted access by Internet bots. However, the reliability of this test depends on the classification accuracy α of the classifier implemented by the bot. For instance, if the classifier has accuracy $\alpha = 95\%$, then the bot fools the Asirra test roughly half of the times ($\alpha^{12} \approx 54\%$). More details about Asirra dataset are provided in section 3.3.

Classification accuracy of 92% was achieved using the shape model discussed previously on the Asirra data. This corresponds to a 42% probability of breaking the test in one try. Our method outperforms best accuracy reported by a long margin, improving performance by 10% compared to 82% reported in [50]. This improvement in binary classification results in significant improvement in probabilities of breaking the test improving probability from just 9.2% to 42%. With 9.2% probability reported previously, a machine take more than 10 attempts to break the test. With our 42% probability, a machine can crack asirra test in every third attempt. This is a significant achievement. It is interesting to

note that just the shape of the head of an animal is a powerful enough to classify it correctly according to its species.

5.3.3 Breed discrimination

This section evaluates the models on the task of discriminating the different breeds of cats and dogs given their species. This is done by learning multi-class SVM by using the 1-vs-rest decomposition [87] (this means learning 12 binary classifiers for cats and 25 for dogs). The relative performance of the different models is similar to the one observed for species classification in Sect. 5.3.1. The best breed classification accuracies are 66.90% and 58.91% for cats and dogs respectively, which become 69.23% and 62.09% when the ground truth segmentations are used.

5.3.4 Species and breed discrimination

This section investigates classifying both the species and the breed. Two approaches are explored: *hierarchical classification*, in which the species is decided first as in Sect. 5.3.1, and then the breed is decided as in Sect. 5.3.3, and *flat classification*, in which a 37-class SVM is learned directly, using the same method discussed in Sect. 5.3.3. The relative performance of the different models is similar to the one observed in Sect. 5.3.1 and 5.3.3. Flat classification is better than the hierarchical one, but the latter requires less work at test time, due to the fact that fewer SVM classifiers need to be evaluated. For example, using the appearance model with the *image, head, image-head layouts* has accuracy 53.90%, adding shape information hierarchically improves this to 54.88%, and using flat classification has accuracy 57.16%. The confusion matrix for the best case is reported in figure 5.3.

5.4 Summary

A number of different models have then been proposed and tested on the tasks of species and breed classification. These models incorporate both shape (in term of a deformable part model of the cat and dog faces) and appearance (in term of geometrically-adapted histograms of visual words). Results on the IIIT-OXFORD PET dataset test data are very encouraging. The performance on the Asirra test data is excellent as well, which is remarkable considering that this dataset was *designed* to be challenging for machines.

Figure 5.3 Failure cases for the model using appearance only (*image layout*) in Sect. 5.3.3. *First row:* Cat images that were incorrectly classified as dogs and vice versa. *Second row:* Bengal cats (b–e) classified as Egyptian Mau (a). *Third row:* English Setter (g–l) classified as English Cocker Spaniel (f).

Chapter 6

Conclusions and future work

In this thesis we have addressed the problem of automatically detecting, classifying and segmenting deformable animals from images. We have proposed machine learning methods for detecting and classifying these objects and also introduced a dataset to evaluate our performance. Major contributions of this work are as follows:

Dataset In chapter 3, we have introduced The IIIT-OXFORD PET dataset, the first computer vision dataset for the problem of pet breed discrimination, a fine grained object categorization problem. The dataset is large, both in the number of images and the range of breeds spanned, contains more than $7,000$ images distributed into 37 different breeds of cats and dogs. The dataset also includes detailed annotations for every image (breed, head bounding box, pixel-level segmentation of each pet). Three challenging tasks are defined: species, breed given species, and species and breed classification. This data, which will be made publicly available, is a great complement to the standard object recognition datasets.

Deformable Animal Detection In chapter 4, we introduced our DisPM detector for deformable animals detection. On VOC 2010 challenge, our method shows significant improvement over part based models and is indeed comparable to state-of-the-art. Our scheme makes use of only two feature types for detecting distinctive part followed by simple segmentation process, matches the performance of state-of-the-art methods using combination of features, pyramids and kernels.

Fine Grained Classification In chapter 5, a number of different models have been proposed and tested on the tasks of species and breed classification. These models were successful in not only identifying the species of the animal, but classifying it further according to its breed. Classification accuracy of 67% was observed on our dataset for automatically classifying pet into one of the 37 different breeds of cats and dogs.

Breaking Asirra Challenge In chapter 5, we tested performance of our models on challenging MSR Asirra test. Over 30% improvement in breaking the test was achieved over previously reported methods

in the literature, which is remarkable considering that this dataset was *designed* to be challenging for machines.

6.1 Future Work

To improve the DisPM performance further will require using more hints, cues and constraints in the segmentation model. For example: (i) Class based edge classification – learning which of the edges are due to the cat silhouette edges, and which arise from other sources (e.g. an occlusion boundary of a chair). Others have learnt edges for classes quite successfully [30, 83]. (ii) Class specific color restrictions – For example, cat coloring is uni or bi modal, e.g. only grey or black and white. (iii) Class specific shape restrictions – parts of the boundary should be smooth and curved. Although we have primarily investigated the DisPM detector for a subset of the animals of the PASCAL VOC challenge, there is no doubt that the distinctive part approach is applicable to many other animal classes.

On the dataset end, more and more subcategories need to be added to further enrich the dataset. Also, existing images need to be inspected by subject expert for sanity check.

Just like detection, classification also needs improvement of the automatic segmentations of the pets to saturate the performance obtained when the ground truth pet segmentations are used. Another direction to explore is the detection and segmentation of the pets, similar to the PASCAL VOC challenges, but focusing on these difficult categories. For subclassification tasks, learning more spatially localized discrimination of animal breeds (e.g. something like particular markings, or shape of nose etc). needs to be researched.

Related Publications

1. Omkar M. Parkhi, Andrea Vedaldi, C. V. Jawahar and Andrew Zisserman
   "The Truth About Cats and Dogs"
   in *proceedings of the* 13^{th} *International Conference on Computer Vision (ICCV) 2011*, Barcelona, Spain.

Bibliography

[1] American kennel club. http://www.akc.org/. 27, 28

[2] The cat fanciers association inc. http://www.cfa.org/Client/home.aspx. 27, 28

[3] Cats in sinks. http://catsinsinks.com/. 27

[4] Catster. http://www.catster.com/. 27, 28

[5] Dogster. http://www.dogster.com/. 27

[6] http://www.flickr.com/. 28

[7] The international cat association. http://www.tica.org/. 27

[8] My cat space. http://www.mycatspace.com/. 27

[9] My dog space. http://www.mydogspace.com/. 27

[10] Petfinder. http://www.petfinder.com/index.html. 27

[11] World canine organisation. http://www.fci.be/. 27

[12] S. Agarwal and D. Roth. Learning a sparse representation for object detection. In *Proc. ECCV*, pages 113–130, 2002. 1, 22

[13] N. Ahuja and S. Todorovic. Connected segmentation tree – a joint representation of region layout and hierarchy. In *Proc. CVPR*, 2008. 38

[14] S. Andrews, I. Tsochantaridis, and T. Hofmann. Support vector machines for multiple-instance learning. In *NIPS*, pages 561–568, 2002. 19

[15] A. animal captcha. http://research.microsoft.com/en-us/um/redmond/projects/asirra/, 2007. 2, 29

[16] F. Bach, G. Lanckriet, and M. Jordan. Multiple kernel learning, conic duality and the smo algorithm. In *Proc. ICML*, 2004. 16

[17] H. Bay, T. Tuytelaars, and L. Van Gool. SURF: Speeded up robust features. In *Proc. ECCV*, May 2006. 8

[18] T. Berg. Animals on the web dataset. http://www.tamaraberg.com/animalDataset/index.html. 1

[19] T. Berg, A. Berg, J. Edwards, M. Mair, R. White, Y. Teh, E. Learned-Miller, and D. Forsyth. Names and Faces in the News. 2004. 1

[20] E. Borenstein and S. Ullman. Class-specific, top-down segmentation. In *Proc. ECCV*, pages 109–124, 2002. 37

[21] E. Borenstein and S. Ullman. Learning to segment. In *Proc. ECCV*, volume 3, pages 315–328, 2004. 37

[22] A. Bosch, A. Zisserman, and X. Munoz. Image classification using random forests and ferns. 2007. 7

[23] A. Bosch, A. Zisserman, and X. Munoz. Representing shape with a spatial pyramid kernel. 2007. x, 8, 9, 11

[24] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *PAMI*, 23(11):1222–1239, 2001. 38, 41, 42, 43, 46

[25] S. Branso, C. Wah, B. Babenko, F. Schroff, P. Welinder, P. Perona, and B. S. Visual recognition with humans in the loop. In *European Conf. on Computer Vision*, 2010. 51

[26] G. Csurka, C. R. Dance, L. Dan, J. Willamowski, and C. Bray. Visual categorization with bags of keypoints. In *Proc. ECCV Workshop on Stat. Learn. in Comp. Vision*, 2004. 50, 51

[27] N. Dalal and B. Triggs. Histogram of Oriented Gradients for Human Detection. In *Proc. CVPR*, volume 2, pages 886–893, 2005. x, 1, 8, 9, 13, 18, 38, 39, 44

[28] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proc. CVPR*, 2005. 51

[29] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *Proc. CVPR*, 2009. 28

[30] P. Dollár, Z. Tu, and S. Belongie. Supervised learning of edges and object boundaries. In *CVPR*, 2006. 59

[31] R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis*. Wiley, 1973. 7

[32] C. Elkan. Using the triangle inequality to accelerate k-means. pages 147–153, 2003. 9

[33] J. Elson, J. Douceur, J. Howell, and J. Saul. Asirra: A CAPTCHA that exploits interest-aligned manual image categorization. In *Conf. on Computer and Communications Security (CCS)*, 2007. 2, 29, 55

[34] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2010 (VOC2010) Results. http://www.pascal-network.org/challenges/VOC/voc20010/workshop/index.html, 2010. 1, 3, 27, 36, 37, 38, 44, 46, 51

[35] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes (VOC) challenge. *IJCV*, 88(2):303–338, June 2010. 1, 23, 44

[36] L. Fei-Fei, R. Fergus, and P. Perona. A Bayesian approach to unsupervised one-shot learning of object categories. In *Proc. ICCV*, 2003. 27

[37] 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 *IEEE CVPR Workshop of Generative Model Based Vision*, 2004. 7

[38] L. Fei-Fei and P. Perona. A Bayesian hierarchical model for learning natural scene categories. June 2005. 8

[39] P. Felzenszwalb, R. Girshick, and D. McAllester. Cascade object detection with deformable part models. In *Proc. CVPR*, pages 2241–2248, 2010. 19

[40] P. Felzenszwalb and D. Huttenlocher. Pictorial structures for object recognition. *IJCV*, 61(1), 2005. 18, 37

[41] P. Felzenszwalb, D. Mcallester, and D. Ramanan. A discriminatively trained, multiscale, deformable part model. In *Proc. CVPR*, 2008. 9, 18, 19

[42] P. F. Felzenszwalb, R. B. Girshick, and D. McAllester. Discriminatively trained deformable part models, release 4. http://people.cs.uchicago.edu/ pff/latent-release4/. 19

[43] P. F. Felzenszwalb, R. B. Grishick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *PAMI*, 2009. vi, xi, 3, 18, 19, 21, 36, 37, 38, 39, 44, 46

[44] P. F. Felzenszwalb, R. B. Grishick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *PAMI*, 2009. 50, 51

[45] R. Fergus, L. Fei-Fei, P. Perona, and A. Zisserman. Learning object categories from Google's image search. In *Proc. ICCV*, 2005. 3, 38

[46] R. Fergus and P. Perona. Caltech object category datasets. http://www.vision.caltech.edu/html-files/archive.html, 2003. 22

[47] M. Fischler and R. Elschlager. The representation and matching of pictorial structures. *IEEE Transactions on Computer*, c-22(1):67–92, Jan. 1973. 18

[48] C. Fleuret and D. Geman. Stationary features and cat detection. *jmlr*, 2008. 36, 37, 50, 51

[49] J. Fritsch, S. Lang, M. Kleinehagenbrock, G. A. Fink, and G. Sagerer. Improving adaptive skin color segmentation by incorporating results from face detection. In *IEEE Workshop on Robot and Human Interactive Communication*, 2002. 37

[50] P. Golland. Discriminative direction for kernel classifiers. In *Proc. NIPS*, 2001. 55

[51] P. Golle. Machine learning attacks against the asirra captcha. In *15th ACM Conference on Computer and Communications Security (CCS)*, 2008. 50

[52] Google: Image search. http://www.google.com/images, 2003. 28

[53] S. Gould, R. Fulton, and D. Koller. Decomposing a scene into geometric and semantically consistent regions. In *iccv*, 2009. 37

[54] G. Griffin, A. Holub, and P. Perona. Caltech-256 object category dataset. Technical Report 7694, California Institute of Technology, 2007. 1, 22

[55] G. Griffin, A. Holub, and P. Perona. Caltech-256 object category dataset. Technical report, California Institute of Technology, 2007. 7, 27

[56] C. Gu, J. J. Lim, P. Arbeláez, and J. Malik. Recognition using regions. In *Proc. CVPR*, 2009. 38

[57] H. Harzallah, C. Schmid, F. Jurie, and A. Gaidon. Classification aided two stage localization. Presented at the ECCV VOC 2008 Workshop, 2008. 3, 38

[58] G. Heitz and D. Koller. Learning spatial context: Using stuff to find things. In *Proc. ECCV*, 2008. 37

[59] D. Hoeim, C. Rother, and J. Winn. 3D layout CRF for multi-view object class recognition and segmentation. In *cvpr*, 2008. 37

[60] M. P. Kumar, P. H. S. Torr, and A. Zisserman. OBJ CUT. volume 1, pages 18–25, 2005. 37

[61] L. Ladicky, P. Sturgess, K. Alahari, C. Russell, and P. Torr. Where, what and how many? combining object detectors and CRFs. In *eccv*, 2010. 37

[62] C. H. Lampert and M. B. Blaschko. Structured prediction by joint kernel support estimation. *Machine Learning*, 2009. 51

[63] C. H. Lampert, M. B. Blaschko, and T. Hofmann. Efficient subwindow search: A branch and bound framework for object localization. *PAMI*, 2009. 3, 38

[64] I. Laptev. Improvements of object detection using boosted histograms. In *Proc. ECCV*, 2006. 37, 45

[65] I. Laptev. Improvements of object detection using boosted histograms. In *Proc. BMVC*, pages 949–958. BMVC, 2006. 50, 51

[66] D. Larlus and F. Jurie. Combining appearance models and Markov random fields for category level object segmentation. In *cvpr*, 2008. 38

[67] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bag of features: Spatial pyramid matching for recognizing natural scene categories. In *Proc. CVPR*, 2006. 50, 52

[68] S. Lazebnik, C. Schmid, and J. Ponce. Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories. 2006. 3, 7, 38

[69] B. Leibe, A. Leonardis, and B. Schiele. Combined object categorization and segmentation with an implicit shape model. In *Workshop on Statistical Learning in Computer Vision, ECCV*, May 2004. 37

[70] A. Levin and Y. Weiss. Learning to combine bottom-up and top-down segmentation. In *Proc. ECCV*, volume 4, pages 581–594, 2006. 37

[71] D. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004. x, 8, 9, 12

[72] D. G. Lowe. Object recognition from local scale-invariant features. In *Proc. ICCV*, 1999. 51

[73] D. R. Martin, C. Fowlkes, and J. Malik. Learning to detect natural image boundaries using local brightness, color, and texture cues. *PAMI*, 1, 2004. 41, 52

[74] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. Van Gool. A comparison of affine region detectors. *IJCV*, 65(1/2):43–72, 2005. 8

[75] D. Mumford and B. Gidas. Stochastic models for generic images. *Quaterly of Applied Mathematics*, 59(1):85–111, 2001. 18

[76] M.-E. Nilsback and A. Zisserman. A visual vocabulary for flower classification. In *Proc. CVPR*, volume 2, pages 1447–1454, 2006. 1, 28, 51

[77] M.-E. Nilsback and A. Zisserman. Delving into the whorl of flower segmentation. In *Proc. BMVC*, volume 1, pages 570–579, 2007. 1, 28

[78] M.-E. Nilsback and A. Zisserman. Automated flower classification over a large number of classes. Dec 2008. 1, 28, 51

[79] T. Ojala, M. Pietikainen, and D. Harwood. A comparative study of texture measures with classification based on featured distributions. *Pattern Recognition*, 29(1):51–59, 1996. 44

[80] A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *IJCV*, 42(3), 2001. 8

[81] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *Proc. CVPR*, 2007. 1

[82] J. Ponce, T. L. Berg, M. Everingham, D. A. Forsyth, M. Hebert, S. Lazebnik, M. Marszałek, C. Schmid, B. C. Russell, A. Torralba, C. K. I. Williams, J. Zhang, and A. Zisserman. Dataset issues in object recognition. In J. Ponce, M. Hebert, C. Schmid, and A. Zisserman, editors, *Toward Category-Level Object Recognition*, volume 4170 of *LNCS*, pages 29–48. Springer, 2006. 7

[83] M. Prasad, A. Zisserman, A. W. Fitzgibbon, M. P. Kumar, and P. H. S. Torr. Learning class-specific edges for object detection and segmentation. Dec 2006. 59

[84] A. Rabinovich, A. Vedaldi, C. Galleguillos, E. Wiewiora, and S. Belongie. Objects in context. In *Proc. ICCV*, 2007. 37

[85] C. Rother, V. Kolmogorov, and A. Blake. Grabcut: interactive foreground extraction using iterated graph cuts. *Proc. SIGGRAPH*, 23(3):309–314, 2004. xi, 3, 17, 18, 38, 42, 52

[86] B. C. Russel, A. A. Efros, J. Sivic, W. T. Freeman, and A. Zisserman. Using multiple segmentations to discover objects and their extent in image collections. In *Proc. CVPR*, 2006. 37

[87] B. Schölkopf and A. J. Smola. *Learning with Kernels*. MIT Press, 2002. 56

[88] J. Shotton, J. Winn, C. Rother, and A. Criminisi. *TextonBoost*: Joint appearance, shape and context modeling for multi-class object recognition and segmentation. In *Proc. ECCV*, 2006. 37

[89] J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *Proc. ICCV*, volume 2, pages 1470–1477, Oct. 2003. 8

[90] J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *Proc. ICCV*, 2003. 50, 51

[91] E. Tola, V. Lepetit, and P. Fua. A fast local descriptor for dense matching. In *Proc. CVPR*, 2008. 7

[92] M. Varma and D. Ray. Learning the discriminative power-invariance trade-off. In *Proc. ICCV*, Rio de Janeiro, Brazil, Oct. 2007. 7, 16, 51

[93] A. Vedaldi and B. Fulkerson. VLFeat: An open and portable library of computer vision algorithms. `http://www.vlfeat.org/`, 2008. x, 9, 10, 12

[94] A. Vedaldi, V. Gulshan, M. Varma, and A. Zisserman. Multiple kernels for object detection. In *Proc. ICCV*, 2009. 1, 3, 16, 38

[95] A. Vedaldi and A. Zisserman. Structured output regression for detection with partial occulsion. In *Proc. NIPS*, 2009. 50

[96] P. Welinder, S. Branson, T. Mita, W. T., and T. Schroff. Caltech-ucsd birds 200. Technical report, UCSD, 2010. 1, 51

[97] K. Yu, T. Zhang, and Y. Gong. Nonlinear learning using local coordinate coding. In *Proc. NIPS*, 2009. 46

[98] J. Zhang, M. Marszałek, S. Lazebnik, and C. Schmid. Local features and kernels for classifcation of texture and object categories: An in-depth study. Technical Report RR-5737, INRIA Rhône-Alpes, Nov 2005. 7

[99] J. Zhang, M. Marszalek, S. Lazebnik, and C. Schmid. Local features and kernels for classification of texture and object categories: A comprehensive study. *IJCV*, 2007. 50, 52

[100] W. Zhang, J. Sun, and X. Tang. Cat head detection - how to effectively exploit shape and texture features. In *European Conf. on Computer Vision*, 2008. 36, 37, 44, 50, 51