KU LEUVEN

ARENBERG DOCTORAL SCHOOL
Faculty of Engineering Science

Object Classification with Latent Parameters

Hakan Bilen

Supervisory Committee:
Prof. dr. ir. H. Van Brussel, chair
Prof. dr. ir. L. Van Gool, supervisor
Prof. dr. ir. L. Van Eycken
Prof. dr. ir. T. Tuytelaars
Prof. dr. ir. E. Duval
Prof. dr. ir. B. Leibe
(RWTH Aachen University)

Dissertation presented in partial fulfillment of the requirements for the degree of Doctor in Engineering

November 2013

Object Classification with Latent Parameters

Hakan BILEN

Supervisory Committee:
Prof. dr. ir. H. Van Brussel, chair
Prof. dr. ir. L. Van Gool, supervisor
Prof. dr. ir. L. Van Eycken
Prof. dr. ir. T. Tuytelaars
Prof. dr. ir. E. Duval
Prof. dr. ir. B. Leibe
(RWTH Aachen University)

Dissertation presented in partial fulfillment of the requirements for the degree of Doctor in Engineering

November 2013

ISBN 978-94-6018-754-4

D/2013/7515/139

Preface

I would like to express my sincere gratitude to my supervisor Professor Luc Van Gool. Thank you for giving me the opportunity to be a member of the VISICS group and the freedom to find my own research direction and for your support. I would also thank Professor Tinne Tuytelaars for many helpful discussions and insights.

I thank my remaining members of my examination committee, Prof. Bastian Leibe, Prof. Erik Duval, Prof. Luc Van Eycken and the chairman Prof. Van Brussel. Their careful reading of my thesis and expertise helped improve its presentation greatly.

I am indebted to Vinay Namboodiri for guiding and mentoring me during these years. It has been a great pleasure working with you. It wouldn't be an overstatement to say that without your help, this thesis would never have been written.

I also thank all my friends and colleagues at the VISICS lab. They were great coworkers and wonderful friends and I have learned a lot from all of them. My special thanks go to Marco Pedersoli for his collaboration and invaluable insights during the last two years of my study. I must thank two programming and math gurus Markus Moll and Jan Hendrik for sharing their knowledge, coding and bug-hunting with me. I particularly thank my officemates Honza (the officemate of August 2013) and Markus (the officemate of September 2013) for not depriving me of their *awesomeness*. Thanks to Bert for blessing me with the computational power of our cluster to make the world even warmer.

I am grateful to many friends in Leuven. There are too many to name but I must thank Korcan, Onur, Egemen, Burak, Mali, Deniz, Zeynep, Kerem and Ercan for being great friends and constant companions during all these years. I especially want to thank Tina for her love, support and understanding of the endless deadlines during these years.

Finally, I want to thank my parents and sister for their unconditional love and support over all these years when I have been away from home. This thesis is dedicated to them.

Abstract

The main challenge of generic object classification - *i.e.* determining that an instance of one or the other object class is present - is that objects of such classes largely vary in appearance. Such differences can possibly be small compared to such differences with other categories. A successful classifier requires being invariant to the intra-class variability while being discriminant to the inter-class differences. The thesis contributes to making the relevant differences count.

This thesis proposes to improve generic object classification by introducing more flexible and richer representations to model certain types of variations. In particular, it learns and models the variations in spatial location, size and appearance of objects, and also interactions with other object categories and their surroundings. This way, we can learn to distinguish the intra-class and inter-class variations better when only given class labels, and that helps to improve visual object classification.

In the first part of the thesis, we address the variability in spatial location and size of objects by introducing a novel object representation that adds spatial information to the standard bag of words representation. We formulate our method in a general setting as inferring additional unobserved or 'latent' dependent parameters. In particular, we focus on two such types of parameters: The first type specifies a cropping operation. This determines a bounding box in the image. This box serves to eliminate non-representative object parts and background. The second type specifies a splitting operation. It corresponds to a non-uniform image decomposition into 4 quadrants, *i.e.* as a generalization of pyramidal bag-of-words.

In addition to variability in their spatial configuration, objects in the same category can differ in their parts and background. In the second part, we propose an object classification method that better handles the complexity of real world images by jointly learning and localizing not only the object, but also a crude layout of its constituent parts as well as the background. We

consider the object of interest as a composition of parts that can be placed together to better model its visual appearance. Furthermore, once the object (or foreground) is localized we also model the background as a composition of constituent parts. In order to enforce coherence in the models and better cope with appearance noise, we also learn pairwise relationships between adjacent parts. This permits us to avoid unlikely part configurations and therefore avoid false positive responses.

In the third part, we focus on learning the inter-class differences between visually similar object categories. We show that jointly learning and localizing pairwise relations between visually similar classes improves such classification: when having to tell whether or not a specific target class is present, sharing knowledge about other, auxiliary classes supports this decision. In particular, we propose a framework that combines target class-specific global and local information with information learnt for pairs of the target class and each of a number of auxiliary classes. Adding such pairwise information helps to learn the common part and context for a class pair and discriminate against other classes.

We evaluate all proposed methods on realistic datasets and compare them against previous, related methods. Extensive experimental evaluations show that modeling and learning the variations in spatial location, appearance and interactions with other object categories and their surroundings improve visual object classification.

Beknopte samenvatting

De belangrijkste uitdaging van generische object classificatie - d.w.z. het vaststellen dat een instantie van een bepaalde objectklasse aanwezig is - is dat objecten van deze klassen vaak sterk verschillen van elkaar in uiterlijk. Dergelijke verschillen binnen een objectcategorie kunnen klein zijn in vergelijking met verschillen overheen verschillende categorieën. Een succesvolle classificeerder vereist invariantie voor de intra-klasse variabiliteit en tegelijkertijd gevoeligheid aan de inter-klasse verschillen.

Dit proefschrift probeert te helpen om de relevante verschillen te laten tellen. Het stelt voor om generische object classificatie te verbeteren door de invoering van flexibelere en rijkere representaties om bepaalde types variaties te modelleren. In het bijzonder leert en modelleert het de variaties in spatiale locatie, grootte en uiterlijk van objecten, alsook interacties met andere objectcategorieën en hun omgevingen. Op deze manier kunnen we leren om beter het onderscheid te maken tussen intra-klasse en inter-klasse variaties wanneer enkel klasse labels gegeven zijn, en dat helpt om visuele object classificatie te verbeteren.

In het eerste deel van het proefschrift richten we ons op de variabiliteit in de ruimtelijke locatie en de grootte van objecten. Dat doen we door de invoering van een nieuwe object representatie die ruimtelijke informatie toevoegt aan de standaard bag of words representatie. We formuleren onze methode in een algemene setting als het afleiden van extra ongeobserveerde of 'latente' afhankelijke parameters. In het bijzonder richten we ons op twee dergelijke soorten parameters. Het eerste type specificeert een trim operatie. Dit bepaalt een selectiekader in het beeld. Dit kader werkt de niet-representatieve object onderdelen en de achtergrond weg. Het tweede type is een opsplitsings-operatie. Het komt overeen met een niet-uniforme beeldontleding in 4 kwadranten, of als een veralgemening van piramidale bag-of-words.

Naast de variatie in de ruimtelijke configuratie kunnen objecten van dezelfde categorie verschillen in hun onderdelen en achtergrond. In het tweede deel

stellen we een object classificatie methode voor die beter omgaat met de complexiteit van natuurlijke beelden door niet alleen het object, maar ook een ruwe layout van zijn bestanddelen en de achtergrond gezamenlijk te leren en te lokaliseren. We beschouwen het betreffende object als een samenstelling van onderdelen die kunnen worden samengevoegd om beter zijn visuele uiterlijk te modelleren. Verder modelleren we het achtergrond model, zodra het object (of de voorgrond) gelokaliseerd is, ook als een samenstelling van onderdelen. Om samenhang in de modellen af te dwingen en beter om te gaan met beeldruis, leren we ook paarsgewijze relaties tussen aangrenzende delen. Dit laat ons toe om onwaarschijnlijke configuraties tussen aanliggende delen te voorkomen en daardoor valse positieven te voorkomen.

In het derde deel richten we ons op het leren van de inter-klasse verschillen tussen visueel vergelijkbare object categorieën. We tonen aan dat gezamenlijk leren en lokaliseren van paarsgewijze relaties tussen visueel gelijkaardige klassen dergelijke classificatie verbetert: wanneer er beslist moet worden of een specifieke klasse aanwezig is, is het voordelig om kennis te delen over andere hulpklassen. Specifiek stellen wij een framework voor dat klasse-specifieke globale en lokale informatie combineert met informatie geleerd voor paren van de doelklasse en elk van de hulpklassen. Het toevoegen van zulke paarsgewijze informatie helpt om het gemeenschappelijke deel en de context te leren voor een klassenpaar en helpt om te differentiëren met andere klassen.

We evalueren alle voorgestelde methoden op realistische datasets en vergelijken ze met eerdere, verwante methoden. Uitgebreide experimentele evaluaties tonen ons dat het modelleren en leren van de variaties in de spatiale ligging, het uiterlijk en de interacties met andere object categorieën en hun omgeving de visuele object classificatie verbeteren.

List of Symbols

$\Delta(.,.)$	loss function
$\boldsymbol{A}$	matrix in uppercase bold letters
$\boldsymbol{a}$	column vector in lowercase bold letters
$\boldsymbol{A}^{-1}$	inverse of matrix $\boldsymbol{A}$
$\boldsymbol{A}^{\mathrm{T}}$	transpose of matrix $\boldsymbol{A}$
$\boldsymbol{h}_i$	latent parameter vector of $\boldsymbol{x}_i$
$\boldsymbol{w}$	weight vector
$\boldsymbol{x}_i$	i-th image
$\mathcal{A}$	sets in uppercase calligraphic letters
$\mathcal{H}$	set of all latent parameters
$\mathcal{X}$	set of all images
$\mathcal{Y}$	set of all image labels
$\psi(.), \phi(.)$	feature functions
a	scalars in normal letters
y_i	label of $\boldsymbol{x}_i$

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Designing intelligent machines is the ultimate goal of the artificial intelligence. Though there is no consensus on the definition of intelligence, a favorable definition by Nilsson in [Nilsson, 2010] is "quality that enables an entity to function appropriately and with foresight in its environment". Critical to interaction with the environment, the field of computer vision aims to enable machines to "see" by interpreting two-dimensional images that are obtained from a three-dimensional world. Achieving this goal has a huge potential to benefit society in many challenging applications such as autonomous driving, surveillance and personal assistance for disabled and elderly people.

Visual object recognition constitutes a key component in such applications that interprets images and establishes a visual association between the content of the image and previously observed data. The observed data can be compactly represented by a label which denotes a specific instance of an object category, e.g. my bicycle, or a generic object category, e.g. bicycle. The label can also be a part of a hierarchy such as vehicle – 2 wheeled vehicle – bicycle. In this manuscript we focus on classification of generic object categories.

It is essential to define what an object category is, before we further detail object recognition. The classical view, that is mostly accepted by the recognition community, defines categories as a set of entities that are shared by all their members. While the commonalities can be chosen in terms of functionality or appearance, the second option is typically favored in the computer vision community. However, forming an ultimate category concept that models all instances of an object category on this planet is almost impossible due to the diversity in appearance of objects. In this manuscript, we limit ourselves to the object categories that come from closed, well-defined sets. In other words, we

use a training set of images which is given by a dataset to learn a category and assume that we can find visual similarity between the given training images and previously unseen (or test) images.

1.1 Recognition and Localization

Generic object recognition tasks can be categorized into four main groups in terms of involved localization level:

Classification: This task involves a decision to determine that an instance of an object class is present in the test image. While the problem of deciding whether one class is present or not, is called *binary classification* in the literature, the problem of deciding whether one or other class is present is called *multi-class classification*. Thus, the classification problem considers only labeling of images but no localization.

Detection: Differently from classification, this task involves the question of "where are the instances of a particular object class (if any)?" [Everingham et al., 2010]. The localization in detection is typically position, scale and aspect ratio of the objects, information can be summarized by a bounding box.

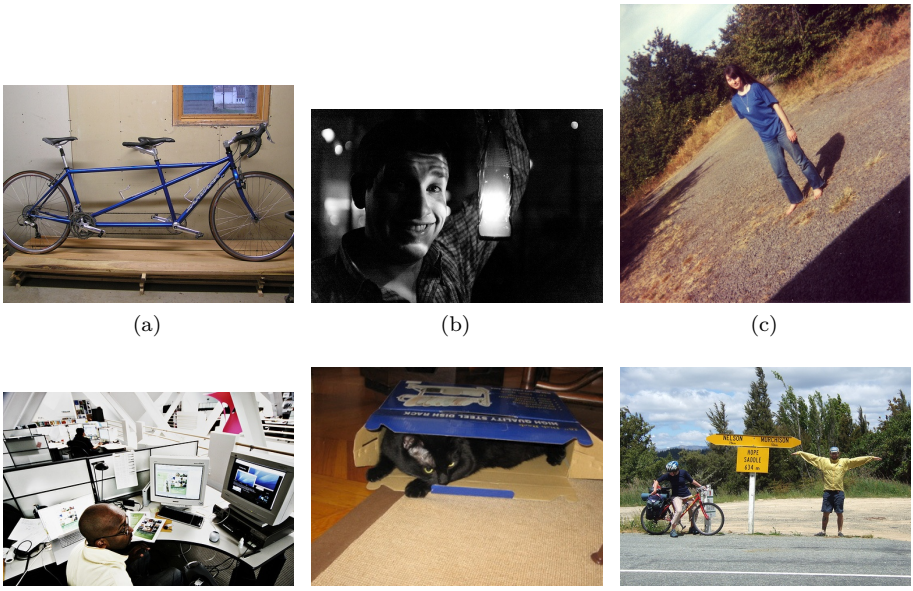
Segmentation: Segmentation can be seen as a refinement of the bounding box, requiring more precise localization and shape information. Differently from detection, it involves the prediction of object class at each pixel in an image.

Pose Estimation: This task involves prediction of part locations and/or their configuration. While pose estimation can also benefit from detection to localize the parts, it requires the poses of these parts as well.

The focus of this manuscript is the first task, object classification problem, i.e. determining class presence in images. In order to prevent confusion, we will use the term "object recognition" to refer to collection of these four problems.

1.2 Challenges

The main challenge of generic object classification is that the images of object instances largely vary in appearance. The main sources of variation are:



(a) (b) (c)

(d) (e) (f)

Figure 1.1: Challenges in generic object classification: (a) intra-class variability, (b) illumination changes, (c) viewpoint, (d) background clutter, (e) occlusion, (f) articulation.

Intra-class variability: Generic object recognition tasks are required to cope with more variation in the appearance within the same object category differently than specific object recognition. The variability can take various forms: color, texture, shape, number of parts, etc. Fig.1.1.(a) illustrates such intra-class variability in the bicycle category. The illustrated example contains more parts (additional seat, pedals, etc.) than a typical two-wheeled bicycle.

Illumination: A good recognition algorithm needs to deal with changes in the illumination. Illumination can mainly have two types of effect: a variation in illumination can change the amount of its radiance on the object surface and this can lead to a variation in its pixel intensities. Secondly, a change in the position of the lighting can cause the formation of shadows on the object surface. A challenging example that is affected by both types of illumination change is illustrated in Fig.1.1.(b).

Viewpoint: An important cause of the variability in appearance is due to geometric transformations during image acquisition. The transformations of the three dimensional world to two dimensional images leads to variations in translation, rotation, skew and scale of objects. An exemplar person image is illustrated in Fig.1.1.(c). Some views can occur more frequently than others and this can be learned as prior knowledge for classification.

Background clutter: Many real world images contain cluttered background and small objects. While visually similar backgrounds can help object classification by providing contextual clues (*e.g.* boats float on water), strong variations in background can make classification tasks more challenging. (See Fig.1.1.(d)).

Occlusion: Some parts of objects can be obscured by another object or be partially visible due to the viewpoint. Partial visibility at image borders is also known as truncation. Moreover, part of an object can occlude other parts which is referred to as self-occlusion. (See Fig.1.1.(e)).

Deformation and Articulation: In addition to rigid transformations, some objects can undergo deformations. The transformation can be in the form of soft tissue deformation such as cells in the presence of an applied force. As well, the transformation can be explained by articulation of rigid structures such as animal and human skeletons. Object classification datasets typically contain images of animals and people in sitting, lying, running configurations that exhibit such transformations. (See Fig.1.1.(f)).

1.3 Objectives

There has been substantial work that addressed some of the aforementioned challenges in object classification. This significant research effort can be broadly grouped into two fundamental approaches. The first approach aims to eliminate the irrelevant factors in images by developing invariant representations. For instance, an autonomous car aims to detect pedestrians on a road regardless of the illumination and clothing styles of those pedestrians. The recognition algorithm can use illumination and color invariant features such as histogram of oriented gradients (HOG). The second approach parameterizes its object representation to model the variations of appearance. For example, the same pedestrian recognition system can use different appearance models to deal with side-views and frontal views.

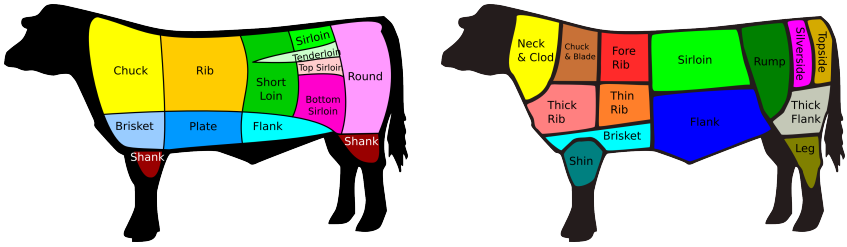


Figure 1.2: Definition of parts is not necessarily universal. The metaphorical diagram shows the US cuts of beef (left) and the British cuts (right). Courtesy of Shimon Edelman [Edelman, 2009].

While using invariant representations enables to represent objects with less parameters and thus leads to more efficient learning methods, it can reduce the discriminative power of our classifiers in certain applications. For example, in the case of digit recognition, a fully rotationally invariant representation cannot discriminate between 6 and 9. On the other hand, parameterization of models by considering various aspects of appearance can enhance the power of representation and its distinctiveness. A richer representation typically comes with more parameters to learn and thus usually requires more supervision. For instance, in order to deal with the view point, one can use annotated locations (e.g. bounding box or segmentation) of training images to learn how to localize objects. However, considering the diversity in object classes and substantial number of images for each class, the annotation can be time-consuming and costly. Moreover, annotating certain aspects of variability can be not well-defined or not optimal for learning. For example, it is not clear how to group the appearance of pedestrians in terms of their clothing with the guarantee that it will lead to the most discriminative groups for classification. By the same token, there is no universal rule to tell us how and to what level to split objects into their parts (see Figure 1.2).

In this manuscript we address learning richer and more flexible representations of object categories with as only supervision class labels (i.e. whether the object class is present in the image). Our method models certain types of variations as unobserved or *latent* variables and learns them in a discriminative setting. In particular we address variability in intra-class appearance, viewpoint, background clutter and occlusion by modeling and learning spatial location, size and appearance of objects, and also interactions with other object categories and with their surroundings, in a weakly supervised framework. This way, we can learn to distinguish the intra-class and inter-class variations better when only given class labels, and we show experimentally that it helps to improve

visual object classification.

1.4 Contributions and Organization

- **Localization:** In Chapter 3, we address the variability in spatial location and size of objects by introducing a novel object representation with more flexible spatial information than the standard spatial pyramid representation [Lazebnik et al., 2006]. We formulate our method in a general setting as inferring additional unobserved dependent parameters. In particular, we focus on two such types of parameters: The first type specifies a cropping operation. This determines a bounding box in the image. This box serves to eliminate non-representative object parts and background. The second type specifies a splitting operation. It corresponds to a non-uniform image decomposition into 4 quadrants. This work is published and was awarded the Best Paper Award at the British Machine Vision Conference 2011 [Bilen et al., 2011] and its extended version [Bilen et al., 2013b] is published in the International Journal of Computer Vision.

- **Multi-modal Appearance and Context:** In addition to variability in their spatial configuration, objects within the same category can differ in their parts and background. In Chapter 4, we propose an object classification method that better handles the complexity of real world images by jointly learning and localizing not only the object, but also a crude layout of its constituent parts as well as the background. We consider the object of interest as a composition of parts that can be composed to better model its visual appearance. Furthermore, once the object (or foreground) is localized we also model the background as a composition of constituent parts. In order to enforce coherence in the models and better cope with the appearance noise, we also learn pairwise relationships between adjacent parts. This permits us to avoid unlikely part configurations and therefore avoid false positive responses. This work is submitted to the Conference of Computer Vision and Pattern Recognition 2014 and is currently under review.

- **Shared Localized Features:** In Chapter 5, we focus on learning the inter-class differences between visually similar object categories. We show that jointly learning and localizing pairwise relations between visually similar classes improves such classification: when having to tell whether or not a specific target class is present, sharing knowledge about other, auxiliary classes supports this decision. In particular, we propose a framework that combines target class-specific global and local information

with information learned for pairs of the target class and each of a number of auxiliary classes. Adding such pairwise information helps to learn the common part and context for a class pair and discriminate against other classes. Parts of this chapter are published in the DAGM 2012 [Bilen et al., 2012] and in the Fine-Grained Visual Classification Workshop in the Conference of Computer Vision and Pattern Recognition 2013 [Bilen et al., 2013a].

Before presenting our contributions of the manuscript in Chapter 3, 4, 5, we review popular image representations and machine learning tools for image classification in Chapter 2. In Chapter 6, we finally summarize our contributions and suggest directions for future work.

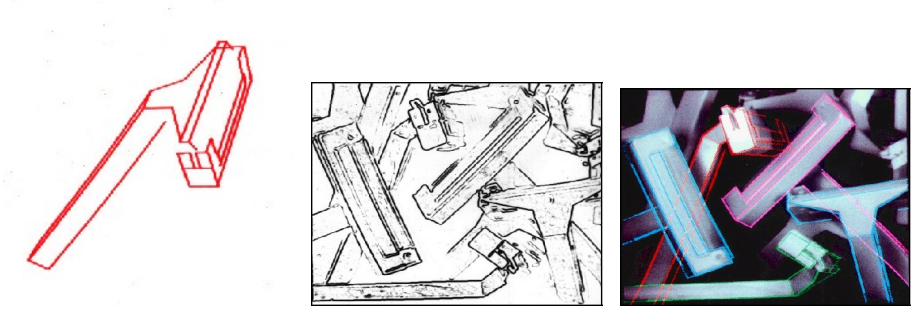
Chapter 2

Visual Object Classification: an overview

This chapter aims to familiarize the reader with category-level object classification. First, we give a short history of the object recognition work over the past four decades and discuss the limitations of the current work, then we position this manuscript along this line. We review the existing work in the field of object classification and focus on the fundamental tools of image representation and machine learning.

2.1 A Historical Perspective

The first attempts towards visual object classification go back to the 1960's. In the first thirty years, the recognition community mainly focused on instance specific object recognition which simplified the visual task by avoiding intra-class variability. The challenges of the early work were typically dealing with variation in viewpoint and illumination, and background clutter. In order to avoid some of the difficulties, many early approaches [Binford, 1971, Agin and Binford, 1973, Ponce and Chelberg, 1988] used triangulation-based range data (depth information) rather than 2-D intensity images and directly obtained a 3D shape of the scene. The field later advanced to make use of 2D intensity images. A typical recognition system of the period stored a 3D model of the target object, which was hand-designed or carefully constructed, and tried to recognize specific instances of this model from 3D range data or a 2D image.



(a) (b) (c)

Figure 2.1: Lowe's SCERPO system [Lowe, 1984]: (a) Three dimensional wire-frame model of the razor, (b) extracted edges of the image, (c) successful matches between sets of image segments and particular viewpoints of the model. (images courtesy of David Lowe)

A representative work example is Lowe's SCERPO system [Lowe, 1984] which extracts lines from a 2D image and groups them by exploiting co-linearity and parallelism. The method computes the unknown viewpoint by aligning projections of a 3D object model with the grouped lines. Figure 2.1 illustrates an example output of the system.

The main strength of the above type of systems was their ability to recognize objects under different viewpoints. They relied on the assumption that contours of objects from images could be robustly extracted. However the systems were not able to recognize objects having significant surface texture in the presence of illumination differences or background clutter. Thus, most of the papers in this period reported experiments on very clean images that were captured in a controlled environment, *e.g.* against a uniform background and under even illumination. Another shortcoming was that the 3D models only allowed for the representation of individual object exemplars rather than a whole object category.

In the 1990's, improvements in CPU technology led to a different paradigm, decline of 3-D shape models [Dickinson, 2009] in object recognition. Faster computers did now allow for an object search with multiple appearance templates that densely sampled the appearances of a 3-D model. The templates were typically obtained from images of a target object that were captured on a turn-table for different viewpoints. For the first time, object recognition systems focused on pixel-based appearance and this helped to avoid segmentation based problems.

A prominent example of the period is the 3-D object recognition system of Murase and Nayar [Murase and Nayar, 1995]. The system firstly collected images of objects for different views and then computed a template for each view by projecting it onto the coefficient space. At test time, a query image was projected to the coefficient space and assigned to the nearest neighbor template (or view). The use of principal component analysis (PCA) significantly decreased the time to compute the similarity between the templates and query. Similarly, Kirby and Sirovic [Kirby and Sirovich, 1990] and Turk and Pentland [Turk and Pentland, 1991] used PCA to tackle the face recognition problem.

In contrast to the previous ones, these methods did not require any explicit 3-D model and also enabled recognition of arbitrarily complex objects by densely modeling different appearances of the 3D object. On the other hand, the templates were computed over complete images and thus the methods had difficulties to cope with background clutter, as well as variations in scale and position. Some of the problems of template-based recognition were overcome in recent years in [Viola and Jones, 2004, Dalal and Triggs, 2005, Felzenszwalb et al., 2010].

In the last decade (2000's), the recognition trend moved from global to local representations. While the first generation of recognition systems also employed local features such as lines, cylinders, etc., modern systems benefited from more robust local features and representations [Schmid and Mohr, 1997, Tuytelaars and Mikolajczyk, 2008, Lowe, 1999a, Bay et al., 2008] which are invariant to certain kinds of geometric and photometric transformations. Furthermore, in contrast to very rigid geometric structure in the early systems, the new ones achieved a better modeling of object categories with more flexible spatial configurations [Sivic and Zisserman, 2003, Csurka et al., 2004, Lazebnik et al., 2006, Leibe et al., 2004, Fergus et al., 2003]. Csurka *et al.* [Csurka et al., 2004] proposed unstructured 'bag of key-points' or 'bag of words' (BoW) by ignoring the spatial information of local features. [Lazebnik et al., 2006] improved the BoW by grouping local features in terms of their image coordinates by using coarse regular grids. Leibe *et al.* [Leibe et al., 2004] model the feature geometry by a star model that learn relative positioning of parts to some reference point. Fergus *et al.* [Fergus et al., 2003] proposed a constellation model that uses joint Gaussian relationships between parts. Felzenszwalb *et al.* [Felzenszwalb and Huttenlocher, 2000] consider more complex geometric relationships between the parts by articulated spring models.

While the bag-of-words approaches [Sivic and Zisserman, 2003, Csurka et al., 2004, Lazebnik et al., 2006] give a predicted label and a rough estimate of object location and only require image labels (whether the target object is present or not) in training data, the later approaches [Leibe et al., 2004, Fergus et al., 2003, Felzenszwalb et al., 2010] provide more information about

the locations of parts and objects and usually require the segmentation of target objects or a uniform background. The problem of the label prediction is referred as to object classification and the ones that also gives location information is referred as to object detection (or localization) in the computer vision community. This manuscript focuses on the first one.

An illustration of the evolution of object recognition over the past decades is depicted in Figure 2.2 in terms of abstraction level between categorical model and input image. As discussed in the previous paragraphs, the common approach in the 1970s idealized images as projections of 3D models by removing object surface markings, background clutter and controlled the lighting conditions. The next generation of object recognition algorithms moved from 3D models to 2D appearance models and brought the object representation closer to 2D. The current object recognition systems benefit from a move from global to local representations which are invariant to scale, translation, rotation, illumination, *etc.* and are able to support object recognition in more realistic images than their predecessors. However, there is still a significant abstraction gap between object models and the invariant features and this limits the success of the current methods to deal with high intra-class variance. In this respect, our manuscript can be seen as an attempt to narrow this gap by developing more expressive object models.

The following sections give some further background on object classification, in particular, the building blocks of the BoW based approaches. We first detail the image representations and then describe related popular learning methods.

2.2 Image Representation for Classification

Many of the state-of-the-art approaches in object classification are based on the popular Bag-of-Visual-Words (BoW) (or Bag-of-Words, Bag-of-Key-points). The success of the BoW methods are also proven in the object classification competitions, the PASCAL VOC Challenges [Everingham et al., b] 2007-2011.

The origins of the BoW can be found in two fields, texture recognition [Julesz, 1981] and document retrieval [Salton and McGill, 1986]. The BoW provided an intuitive and powerful representation for texture recognition applications since texture is often characterized by repetition of basic elements or distinctive image patterns (textons [Julesz, 1981]) and usually not their spatial arrangement. Another early use of BoW can be found in document retrieval [Salton and McGill, 1986] that represents documents as frequencies of words from a dictionary by ignoring the order of words in sentences.

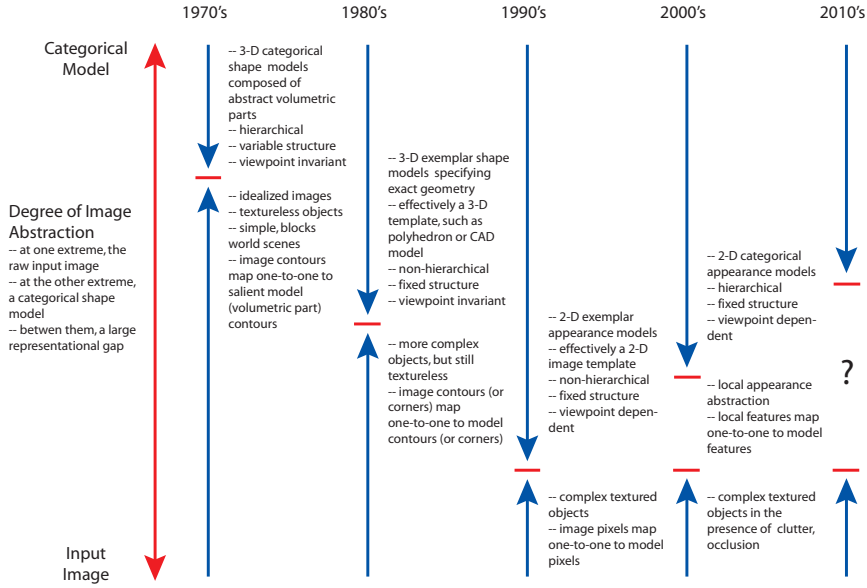


Figure 2.2: Evolution of object recognition over the past four decades [Dickinson, 2009] see the discussion in the text. (courtesy of Sven Dickinson).

Inspired by the previous use of the BoW in texture recognition and document retrieval, the first BoW based applications were applied to image retrieval [Sivic and Zisserman, 2003] and then to object classification tasks [Csurka et al., 2004]. While the documents are composed of words and can be represented as count of words in a discrete way, images are typically described in the continuous domain by intensity values or more sophisticated features like SIFT [Lowe, 2004], SURF [Bay et al., 2008], *etc.* and therefore images do not have any direct equivalents to words. Thus adapting the BoW to the image domain requires the construction of a dictionary or codebook that contains "visual words". Based on the dictionary, samples from images can be assigned to the most similar "word(s)" and this way an image can be represented by a fixed dimensional vector or a histogram that contains the frequency of each visual word occurrence. Those histograms can be used to represent images and then to learn object classifiers (see Figure 2.3).

Many of the classification systems that are based on the BoW representation follow a similar pipeline, see Figure 2.4. First, the most relevant regions are detected in each image and these support regions are then represented by local feature descriptors. These descriptors are quantized into visual words by using a codebook and the encoded vectors are pooled together to form the image

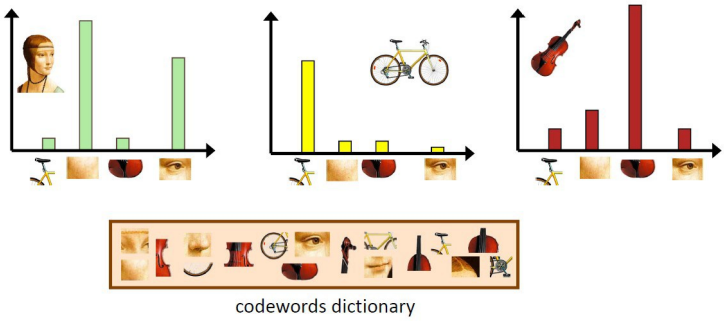


Figure 2.3: Illustration of bag-of-words: Images are first divided into smaller regions and a selection of those regions form a codewords dictionary (bottom row). Each object is then in this oversimplified example represented by an orderless list of four visual words from the dictionary (top row). (illustration courtesy of Li Fei Fei)

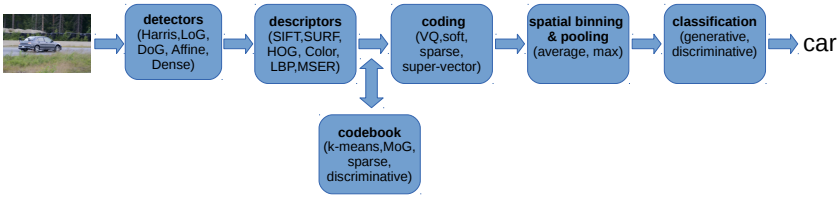


Figure 2.4: The popular BoW pipeline, see text for details.

level representation. Finally, these representations are fed into machine learning tools to predict the label of unseen images. The following parts explain each block of the pipeline in more detail.

Feature Detectors: The first step of the pipeline provides a number of support regions in each image for the subsequent descriptor computation and thus has a direct impact on the final image representation. In the literature, there have been two different approaches to sample or detect local regions.

The first approach is based on detecting regions that are covariant with a class of transformations such as viewpoint, scale, illumination changes. These detectors have initially been used for wide baseline matching for stereo pairs [Pritchett and Zisserman, 1998, Tuytelaars and Van Gool, 2000] that matches regions which are projections of the same 3D patch for different viewpoints. They detect interest points in scale-space and then compute an elliptical

region around those points. The interest points are usually found with Harris detector [Harris and Stephens, 1988] or based on the Hessian matrix. The scale selections can be *e.g.* obtained from the second moment of intensity gradient [Baumberg, 2000, Lindeberg, 1998]. The popular operators that are used in scale-space invariant methods are the Laplacian [Lindeberg, 1998], the Harris-Laplace [Mikolajczyk and Schmid, 2001], difference of Gaussian [Lowe, 1999b] and saliency [Kadir and Brady, 2001]. The scale-space is extended to affine invariant detectors in [Tuytelaars and Van Gool, 2000, Mikolajczyk and Schmid, 2002]. An extensive evaluation of the affine invariant region detectors can be found in [Tuytelaars and Mikolajczyk, 2008].

The second approach advocates a dense sampling of points on a regular grid over different scales. Differently from the previous approach dense extraction uses regions even if they are not distinctive for the content and ensures that information loss to reconstruct the image is low. After Nowak *et al.* [Nowak et al., 2006] showed that a dense feature extraction strategy yields better classification performance than using interest point detectors in commonly used datasets, dense sampling has become the de-facto strategy for classification.

Descriptors: Each sampled region can now be represented by a local descriptor. There is a large choice of possible descriptors that use different image properties such as pixel intensities, colors, edges. Some popular descriptors are:

- **Scale Invariant Feature Transform (SIFT)** [Lowe, 2004] is one of the most popular descriptors that is used in object classification and retrieval. The descriptor computes a 3D histogram of gradient orientation and location around a pixel at a selected scale. The spatial information is quantized into a 4×4 grid and gradient orientation is encoded in 8 bins. Due to the use of gradient values and l_2 normalization, the descriptor is invariant against additive and multiplicative intensity changes. Moreover, the spatial binning and local averaging of gradients brings a certain robustness to some level of geometric transformation.
- **Color SIFT** is an extension of the SIFT to different color spaces such as rg, HSV, etc. A comparative study of invariance and distinctiveness of SIFT in different color spaces can be found in [van de Sande et al., 2010].
- **Speeded Up Robust Features (SURF)** [Bay et al., 2008] computes the gradient orientations in two dimensions (x and y) and the use of integral images and box filters provides a computational advantage over the SIFT descriptors.
- **Histogram of Oriented Gradients (HOG)** [Dalal and Triggs, 2005] is similar to the SIFT descriptor, however, it uses a different normalization.

- **GIST** [Oliva and Torralba, 2001] is a global descriptor which is originally developed to represent scene images in a low-dimensional space. The descriptor encodes perceptual features such as naturalness, openness, roughness, expansion, ruggedness of an image and estimates them by using spectral information.
- **Local Self Similarities (LSS)** [Shechtman and Irani, 2007] describe images by the similarity of a pixel to its neighbors and captures the geometric layout of local regions.

Visual dictionary: The visual dictionary is a collection of visual words that are used to represent images. Visual words are obtained by partitioning the local descriptor space into local regions. The internal structure of these regions is usually disregarded and the regions are represented by a single member which is called visual word. A good visual dictionary should provide informative words to enable discriminative image representations. The visual dictionary is typically obtained by clustering which is an unsupervised grouping of similar features.

The K-means clustering [MacQueen et al., 1967] is probably the most common clustering method for dictionary learning. Given N descriptors where each of them is represented by a D-dimensional vector, $\boldsymbol{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N] \in R^{D \times N}$, K-means searches a visual dictionary with K vectors (or cluster centers) $\boldsymbol{V} = [\boldsymbol{v_1}, \ldots, \boldsymbol{v_K}] \in R^{D \times K}$ and a data-to-cluster assignment $\boldsymbol{U} = [\boldsymbol{u_1}, \ldots, \boldsymbol{u_N}] \in R^{K \times N}$ such that it minimizes the following constrained reconstruction problem:

$$\min_{\boldsymbol{V}} \sum_{i=1}^{N} \min_{\boldsymbol{u}_i} ||\boldsymbol{x_i} - \boldsymbol{V}\boldsymbol{u_i}||^2 \text{ s.t. } ||\boldsymbol{u_i}||_{l_0} = 1, ||\boldsymbol{u_i}||_{l_1} = 1, \boldsymbol{u_i} \succeq 0, \forall i. \tag{2.1}$$

The cardinality constraint $||\boldsymbol{u_i}||_{l_0} = 1$ means that there will be only one non-zero element in the corresponding coding $\boldsymbol{u_i}$ of image $\boldsymbol{x_i}$. $||\boldsymbol{u_i}||_{l_1} = 1, \boldsymbol{u_i} \succeq 0$ requires this single non-zero element to be 1.

As the cluster centers as well as the assignments are unknown and depend on each other, this problem is not convex and expectation-maximization is thus typically used in this optimization. Although visual dictionary computation is required only once at training time, it can still be computationally infeasible in case of large number of clusters ($> 10,000$). Thus one can also use more efficient algorithms to compute the visual dictionary such as hierarchical k-means [Nister and Stewenius, 2006] or approximate nearest neighbor [Muja and Lowe, 2009].

Other common clustering methods are: uniform discretization of feature space using a regular lattice [Tuytelaars and Schmid, 2007], mean-shift clustering that learns a non-uniform distribution [Jurie and Triggs, 2005] differently from K-means, mixture of Gaussians (MoG) that describes each cluster in terms of Gaussian density and assigns a probability to each point ($\boldsymbol{x}_i$) for each cluster cluster ($\boldsymbol{v}_i$) [Farquhar et al., 2005], or sparse coding methods that minimize the reconstruction error by assigning each point ($\boldsymbol{x}_i$) to a linear combination of cluster centers [Yang et al., 2009, Mairal et al., 2009].

The visual dictionaries can also be built discriminatively by exploiting class labels of images. There is a rich body of work [Maree et al., 2005, Moosmann et al., 2006, Shotton et al., 2008] that trains random forests [Breiman, 2001] on image patches in order to use them as discriminative codebooks. Moosmann *et al.* [Moosmann et al., 2006] use a randomized decision forest that recursively divides training images and codes them as visual descriptors. The descriptors are the transformed into a set of leaf node indices and votes for each index are accumulated into a global histogram. Semantic texton forests [Shotton et al., 2008] extend [Moosmann et al., 2006] by using the decision forest as a classifier and branch nodes in addition to the leaf nodes.

Encoding of Local Features: We can now encode the extracted descriptors using the learned visual dictionary such that the resulting coding has the same dimensionality for each image. Many successful encoding methods that have different focus on invariance and computation requirements have been proposed in the literature. A simple and commonly used coding approach is vector quantization (VQ) or hard assignment. As in Eq.(2.1) each local descriptor is assigned to its nearest neighbor cluster center and the statistics of the assignments are stored in a histogram. However, representing each descriptor by a single cluster center has limited expressiveness and causes significant information loss. Boiman *et al.* [Boiman et al., 2008] showed that discriminative descriptors are rare and frequently occurring descriptors which are less informative have lower quantization error and are more likely to be chosen by K-means optimization in Eq.(2.1). In other words, hard quantization of descriptors reduces the amount of discriminative information.

To ameliorate the quantization loss in VQ, many successful encoding methods have been proposed in recent years. Examples of commonly used coding schemes are:

- **Kernel Codebook** encoding [Philbin et al., 2008, van Gemert et al., 2008] relaxes the cardinality constraints $||\boldsymbol{u_i}||_{l_0} = 1$ and $||\boldsymbol{u_i}||_{l_1} = 1$ in Eq.(2.1)

by a *soft assignment* to cluster centers:

$$\boldsymbol{u_i}[k] = \frac{K(\boldsymbol{x_i}, \boldsymbol{v_k})}{\sum_{j=1}^{K} K(\boldsymbol{x_j}, \boldsymbol{v_k})} \tag{2.2}$$

where $\boldsymbol{u_i}[k]$ denotes the k-th element of vector u_i, $K(\boldsymbol{x}, \boldsymbol{v}) = \exp(\frac{\gamma}{2}||\boldsymbol{x} - \boldsymbol{v}||^2)$ and γ is a constant that defines the width of the distribution.

- **Improved Fisher** encoding [Perronnin et al., 2010] uses a MoG model to obtain the cluster centers and then computes the first ($\boldsymbol{u'_k}$) and second order differences ($\boldsymbol{u''_k}$) between the image descriptor and cluster centers:

$$\boldsymbol{u'_k} = \frac{1}{N\sqrt{\pi_k}} \sum_{i=1}^{N} \boldsymbol{u_i}[k] \boldsymbol{\Sigma}_k^{-\frac{1}{2}} (\boldsymbol{x_i} - \boldsymbol{v_k}) \tag{2.3}$$

$$\boldsymbol{u''_k} = \frac{1}{N\sqrt{2\pi_k}} \sum_{i=1}^{N} \boldsymbol{u_i}[k][(\boldsymbol{\Sigma}_k^{-\frac{1}{2}} (\boldsymbol{x_i} - \boldsymbol{v_k})) \odot (\boldsymbol{\Sigma}_k^{-\frac{1}{2}} (\boldsymbol{x_i} - \boldsymbol{v_k})) - \mathbf{1}_D] \tag{2.4}$$

where π, $\boldsymbol{\Sigma} \in R^{D \times D}$, $\boldsymbol{u_i}[k]$ and $\mathbf{1}_D$ denote the prior probabilities, positive semi-definite covariance matrix, soft assignment of $\boldsymbol{x_i}$ to cluster center v_k and a D dimensional 1 column vector respectively. $\odot$ is an element-wise multiplication operator. The Fisher encoding of an image for a given set of descriptors can be written as a vector of size $2DK$:

$$\boldsymbol{F}(\boldsymbol{X}) = [\boldsymbol{u_1'}^{\mathrm{T}}, \boldsymbol{u_1''}^{\mathrm{T}}, \ldots, \boldsymbol{u_K'}^{\mathrm{T}}, \boldsymbol{u_K''}^{\mathrm{T}}]^{\mathrm{T}}. \tag{2.5}$$

- **Super Vector** encoding [Zhou et al., 2010a] is similar to the Fisher encoding. However, it considers only the first order differences between features and clusters. Additionally it includes the weighted mean of each cluster and uses a posterior normalization.

- **Locality-constrained linear (LLC)** encoding [Wang et al., 2010] also relaxes the cardinality restriction on $\boldsymbol{u_i}$ and generates a locally smooth sparse representation by incorporating the locality constraint:

$$\min_{\boldsymbol{U},\boldsymbol{V}} \sum_{i=1}^{N} ||\boldsymbol{x_i} - \boldsymbol{V}\boldsymbol{u_i}||^2 + \lambda ||\boldsymbol{d_i} \odot \boldsymbol{u_i}||^2 \text{ s.t. } ||\boldsymbol{u_i}||_{l_1} = 1, \forall i. \tag{2.6}$$

where $\odot$ and λ denote an element-wise multiplication and a small regularization coefficient respectively. $\boldsymbol{d_i}$ is a vector of Euclidean distances between $\boldsymbol{x_i}$ and $(\boldsymbol{v_1}, \ldots, \boldsymbol{v_K})$ and the locality constraint ($||\boldsymbol{d_i} \odot \boldsymbol{u_i}||^2$) thus leads to a locally smooth and sparse representation by penalizing the assignments from feature $\boldsymbol{x_i}$ to dissimilar cluster centers $\boldsymbol{v_k}$.

A comprehensive evaluation of these encoding methods can be found in [Chatfield et al., 2011].

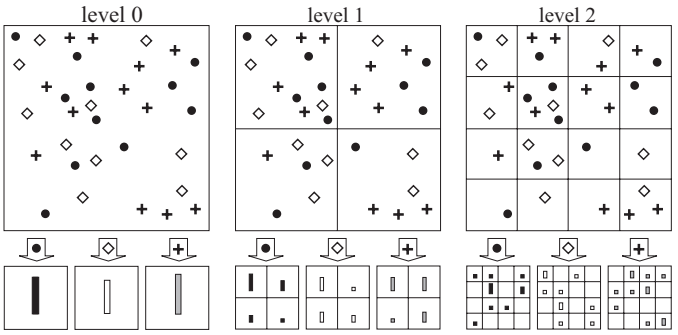


Figure 2.5: Illustration of 3-level spatial pyramid (SP) for a toy example: An image with three visual words that are shown with circle, cross, diamond shapes are divided into smaller cells by 1×1, 2×2 and 4×4 grids at three resolution levels. Each spatial cell is represented by an individual histogram. (image courtesy of Svetlana Lazebnik)

Spatial Binning: The standard BoW method represents images as orderless lists by ignoring spatial layout of images. While this brings a certain level of invariance against viewpoint changes, some informative geometric relations are lost in this setting. Lazebnik *et al.* [Lazebnik et al., 2006] have addressed this shortcoming and have shown that incorporating weak geometry leads to a richer BoW representation and improves classification. In this representation, images are divided into regular regions at different resolutions and it is thus named Spatial Pyramid (SP) representation. Features from each of these regions are encoded into a histogram and finally concatenated to form the final representation of an image. Images are typically divided by a $2^l \times 2^l$ grid for 2 pyramid levels $l = \{0, 1, 2\}$ and this leads to a higher dimensional representation (*e.g.* 3 level SP contains $1 + 2 \times 2 + 4 \times 4 = 21$ histograms). An illustration of the 3 level SP is depicted in Figure 2.5.

While SP is an improvement over the standard BoW representation, it imposes a rigid geometric structure on images and does not really learn the optimal layout of images. There has been limited work [Sharma and Jurie, 2011, Krapac et al., 2011] that have addressed this shortcoming. Sharma and Jurie [Sharma and Jurie, 2011] have proposed a discriminative approach that learns a layout by successively splitting the image into spatial cells. Krapac *et al.* have [Krapac et al., 2011] modeled spatial layout by the mean and variance of each visual word occurrence with the spatial Fisher Vector models. Both methods report an improvement in classification accuracy over the standard SP.

Pooling: In order to compute the final representation, the encoded descriptors on each spatial cell can be *pooled* or combined in two ways: average or max pooling. In the case of average pooling, the encoded features inside each spatial cell are pooled together by averaging visual word counts into a histogram and applying a normalization. In the case of max-pooling, each bin of the histogram is assigned to the maximum value of feature encodings. Yang *et al.* [Yang et al., 2009] report superior classification performance on several classification datasets with max-pooling when using linear classifiers.

2.3 Machine Learning Methods for Classification

After we have reviewed the popular methods along the BoW pipeline, we will now focus on machine learning tools for classification. In particular, we have a discrete class label y (such as the image contains a car or not) that we wish to predict based on a set of features x (such as the BoW representation). We are given a *training* set that consists of pairs with on the one hand the features $X = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ and on the other the corresponding labels $Y = \{y_1, \ldots, y_n\}$. Based on the given training data, we design a model and learn the parameters of this model, which will be used to predict the outcome for an unseen image.

From a probabilistic point of view, we can formulate the learning problem as finding the conditional distribution $p(y|\boldsymbol{x})$. In this case, we build our model to represent the conditional distribution and determine its parameters using the training set. This is known as discriminative model because we use the conditional distribution to discriminate directly between different labels y. The alternative approach is to find the joint distribution $p(\boldsymbol{x}, y)$ that uses the Bayes rule –*i.e.* $p(\boldsymbol{x}, y) = p(\boldsymbol{x}|y)p(y) = p(y|\boldsymbol{x})p(\boldsymbol{x})$ – to calculate $p(y|\boldsymbol{x})$ in order to make predictions of y for new samples $\boldsymbol{x}$. This is known as generative approach, since one can generate new synthetic data $\boldsymbol{x}$ by sampling from the joint distribution. In practice, the generalization performance of generative models is found to be poorer than discriminative ones due to the mismatch between the model and distribution of data. The classification models in this manuscript are also built on a discriminative learning framework.

Discriminative classification methods can broadly be grouped in two groups in terms of their parameterization: non-parametric and parametric methods. The k-Nearest Neighbor (k-NN) [Fix and Hodges, 1951, Cover and Hart, 1967] that requires no model is a simple but powerful non-parametric method used in object classification. Given a new example, it finds the closest k training samples $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k)$ in distance (commonly Euclidean) and then classifies according to majority vote among the k nearest neighbors. k-NN is a memory-based

classification method that uses training samples during testing and does not require any training. While this allows for easily adding new samples to the training set without any additional training, it requires both computing distances to all training samples and storing the entire training set. To ameliorate the computational and storage load, more efficient k-NN algorithms are proposed in [Leibe et al., 2006, Silpa-Anan and Hartley, 2008, Muja and Lowe, 2009].

The simplest parametric classification is based on linear models. In spite of their simplicity, linear models are often quite competitive to more general non-linear models in terms of performance. In addition to their competitive performance, they can usually be optimized by more efficient techniques. In the following part, our focus will be on linear methods for classification. We firstly give a general description of linear classifiers, and then focus on linear discriminant analysis (LDA), logistic regression and finally we discuss support vector machines (SVM) and their extensions to structured output classification with unobserved variables.

Our goal in classification is to take an input vector $\boldsymbol{x} \in \mathcal{R}^D$ and assign it to one of K discrete classes C_k where $k = 1, \ldots, K$. Most commonly, the classes are exclusive, *i.e.* each input is assigned to only one class. In such cases, we can always divide the input space into K *decision regions* such that each represents one of the class labels. The boundaries that separate the decision regions are called *decision boundaries*. For linear models, these boundaries are linear functions of the input vector $\boldsymbol{x}$. In order to assign the class label to the input, we use a *discriminant function* that measures the matching quality between the input $\boldsymbol{x}$ and the class labels. For a binary classification problem (*e.g.* is there a car in the image?), the linear discriminant function is written as

$$f_{\boldsymbol{w}}(\boldsymbol{x}) = \boldsymbol{w}^{\mathsf{T}}\boldsymbol{x}, \tag{2.7}$$

where $\boldsymbol{w} \in R^D$ denotes a hyperplane that defines the decision boundary. The prediction rule is given as

$$g_{\boldsymbol{w}}(\boldsymbol{x}) = \text{sign}(f_{\boldsymbol{w}}(\boldsymbol{x})) \tag{2.8}$$

which predicts that the object class is present (*i.e.* $y = 1$), if $g_{\boldsymbol{w}}(\boldsymbol{x}) > 0$. One can also include an additional bias value b such that $\boldsymbol{w}^{\mathsf{T}}\boldsymbol{x} + b$. For clarity reason, we redefine $\boldsymbol{x}$ and $\boldsymbol{w}$ as the original $\boldsymbol{x}$ and $\boldsymbol{w}$ are extended with the additional element one *i.e.* $[\boldsymbol{w}\ b]$ and $[\boldsymbol{x}\ 1]$.

2.3.1 Linear Discriminant Analysis (LDA)

Linear classification can be considered as a dimensionality reduction technique such that we take a D dimensional input x and then project it to one dimension

by using the discriminant function as in (2.7). However, classes that are well separated in D dimensional space can overlap in one dimensional space. The goal of LDA is to find a projection and thus a weight vector $\boldsymbol{w}$ that minimizes such an overlap and maximizes the class separation. Maximizing class separation requires to find a projection that minimizes the overlap between the intervals defined by the class means and variance within each class.

For a two class problem with N_1 points of class C_1 and N_2 points of class C_2, the class mean $\boldsymbol{m}_k$ is given by

$$\boldsymbol{m}_k = \frac{1}{N_k} \sum_{i \in C_k} \boldsymbol{x}_i \text{ for } k \in \{1, 2\}. \tag{2.9}$$

The class mean (m_k) and within class variance (s_k^2) of the transformed data for class C_k are written as

$$m_k = \boldsymbol{w}^\mathsf{T} \boldsymbol{m_k}, \;\; s_k^2 = \sum_{i \in C_k} (\boldsymbol{w}^\mathsf{T} \boldsymbol{x_i} - m_k)^2. \tag{2.10}$$

The Fisher criterion that defines the ratio between the between class variance $(m_2 - m_1)^2$ and the total within class variance $s_1^2 + s_2^2$ is given as

$$J = \frac{(m_2 - m_1)^2}{s_1^2 + s_2^2}. \tag{2.11}$$

The dependence on $\boldsymbol{w}$ can be made explicit as

$$J(\boldsymbol{w}) = \frac{\boldsymbol{w}^\mathsf{T} \boldsymbol{S_B} \boldsymbol{w}}{\boldsymbol{w}^\mathsf{T} \boldsymbol{S_W} \boldsymbol{w}} \tag{2.12}$$

where $\boldsymbol{S_B}$ and $\boldsymbol{S_W}$ denote the between class and within class covariance matrices respectively and are given by:

$$\boldsymbol{S_B} = (\boldsymbol{m_2} - \boldsymbol{m_1})(\boldsymbol{m_2} - \boldsymbol{m_1})^\mathsf{T} \tag{2.13}$$

$$\boldsymbol{S_W} = \sum_{k \in \{1,2\}} \sum_{i \in C_k} (\boldsymbol{x_i} - \boldsymbol{m_k})(\boldsymbol{x_i} - \boldsymbol{m_k})^\mathsf{T}. \tag{2.14}$$

The ratio can be maximized by differentiating $J(\boldsymbol{w})$ with respect to $\boldsymbol{w}$ and setting it to 0.

$$(\boldsymbol{w}^\mathsf{T} \boldsymbol{S_B} \boldsymbol{w}) \boldsymbol{S_W} \boldsymbol{w} = (\boldsymbol{w}^\mathsf{T} \boldsymbol{S_W} \boldsymbol{w}) \boldsymbol{S_B} \boldsymbol{w}. \tag{2.15}$$

We see that $\boldsymbol{S_B} \boldsymbol{w}$ is always in the direction of $(\boldsymbol{m_2} - \boldsymbol{m_1})$. Moreover, $(\boldsymbol{w}^\mathsf{T} \boldsymbol{S_B} \boldsymbol{w})$ and $(\boldsymbol{w}^\mathsf{T} \boldsymbol{S_W} \boldsymbol{w})$ are scalar values, we can drop them and this yields the following relationship:

$$\boldsymbol{w} \propto {\boldsymbol{S_W}}^{-1} (\boldsymbol{m_2} - \boldsymbol{m_1}). \tag{2.16}$$

We refer to [Hastie et al., 2001, Bishop et al., 2006] for further details.

One of the earliest uses of LDA in computer vision was for face recognition [Belhumeur et al., 1997]. It is shown that LDA outperforms the principal component analysis (PCA) in face recognition applications. This can be explained with the fact that PCA projects the data onto the directions with most variations and ignores discriminativity of the directions. Recently, Hariharan *et al.* [Hariharan et al., 2012] have shown that LDA classifiers can be used as an efficient alternative to support vector machines (SVM).

2.3.2 Logistic Regression

Logistic regression is a probabilistic discriminative method that models the posterior probabilities of K classes by linear functions of the input $\boldsymbol{x}$ and ensures that their sum equals to one. The posterior of the k-th and the K-th models can be written as:

$$\begin{aligned} p(C_k|\boldsymbol{x}) = & \quad \frac{\exp(\boldsymbol{w}_k^\mathsf{T}\boldsymbol{x})}{1+\sum_{k=1}^{K-1}\exp(\boldsymbol{w}_k^\mathsf{T}\boldsymbol{x})} \\ & \quad \vdots \\ p(C_K|\boldsymbol{x}) = & \quad \frac{1}{1+\sum_{k=1}^{K-1}\exp(\boldsymbol{w}_k^\mathsf{T}\boldsymbol{x})}. \end{aligned} \tag{2.17}$$

When $K=2$, the model can simply be expressed by a single set of parameters (*i.e.* $\boldsymbol{w}=\boldsymbol{w}_1$) and the posterior probability of class C_2 can be written in terms of class C_1 such that $p(C_2|\boldsymbol{x}) = 1 - p(C_1|\boldsymbol{x})$.

The parameters of logistic regression are usually determined by the use of maximum likelihood and the conditional probabilities in Eq.(2.17). Given N pairs of samples and their labels $(\boldsymbol{x}_i, y_i)$ ($y_i = 1$, if $\boldsymbol{x}_i$ belongs to C_1, $y_i = 0$, else.), the log likelihood for $\boldsymbol{w}$ is

$$\begin{aligned} l(\boldsymbol{w}) &= \sum_{i=1}^{N}\{y_i \log p(\boldsymbol{x}_i;\boldsymbol{w}) + (1-y_i)(1-\log p(\boldsymbol{x}_i;\boldsymbol{w}))\} \\ &= \sum_{i=1}^{N}\{y_i \boldsymbol{w}^\mathsf{T}\boldsymbol{x}_i - \log(1+\exp(\boldsymbol{w}^\mathsf{T}\boldsymbol{x}_i))\}. \end{aligned} \tag{2.18}$$

2.3.3 Support Vector Machines (SVM)

SVMs [Vapnik, 1999] are among the most popular learning methods in many classification problems including object classification. SVMs do not provide posterior probability as in logistic regression but find the smallest generalization error for the training set in terms of a *margin*. The margin is the smallest distance between the decision boundary (or hyperplane) and any of the samples. The goal is to choose the decision boundary which maximizes the margin.

Using the discriminant function in Eq.(2.7), the perpendicular distance from the hyperplane $\boldsymbol{w}$ to the sample $\boldsymbol{x_i}$ can be written as $|\boldsymbol{w}^\mathsf{T}\boldsymbol{x}|/||\boldsymbol{w}||$ where $|.|$ and $||.||$ denote absolute value and the euclidean norm respectively. The optimization problem that aims to find the biggest margin between the hyperplane and the training samples is found by solving

$$\max_{\boldsymbol{w}}\{\frac{1}{||\boldsymbol{w}||}\min_i y_i\boldsymbol{w}^\mathsf{T}\boldsymbol{x_i}\}. \tag{2.19}$$

It should be noted that we are only interested in the solutions where all samples are correctly classified $y_i\boldsymbol{w}^\mathsf{T}\boldsymbol{x_i} > 0$. The direct optimization of the given formulation can be very complex, the problem should thus be converted to a simpler equivalent problem. To do so, we can use the observation that scaling $\boldsymbol{w}$ with a parameter κ does not change the distance to the hyperplane - *i.e.* $(\kappa y_i\boldsymbol{w}^\mathsf{T}\boldsymbol{x_i})/(\kappa||\boldsymbol{w}||)$. Therefore we can set

$$y_i\boldsymbol{w}^\mathsf{T}\boldsymbol{x_i} = 1 \tag{2.20}$$

for the closest point to the hyperplane. In this case, all samples satisfy the constraint $y_i\boldsymbol{w}^\mathsf{T}\boldsymbol{x_i} \geq 1$. Now we can rewrite the optimization problem in Eq.(2.19) as a standard constrained quadratic optimization problem:

$$\arg\min_{\boldsymbol{w}} \frac{1}{2}||\boldsymbol{w}||^2 \text{ subject to } y_i\boldsymbol{w}^\mathsf{T}\boldsymbol{x_i} \geq 1. \tag{2.21}$$

When positive and negative samples are perfectly separable by a hyperplane, a solution that satisfies these constraints can be found. However, the classes in feature space can also overlap. In this case, we allow some samples to be on the wrong side of the margin but we penalize these violations by adding some cost in the optimization.

$$\arg\min_{\boldsymbol{w}} J(\boldsymbol{w}) = \arg\min_{\boldsymbol{w}} \{\frac{1}{2}||\boldsymbol{w}||^2 + C\sum_i \max(0, 1 - y_i\boldsymbol{w}^\mathsf{T}\boldsymbol{x_i})\} \tag{2.22}$$

where $J(\boldsymbol{w})$ is the *objective function* and C is a trade-off parameter that penalizes the margin violations. The sub-gradient of Eq.(2.22) with respect to $\boldsymbol{w}$ is

$$\frac{\partial J(\boldsymbol{w})}{\partial \boldsymbol{w}} = \boldsymbol{w} - C\sum_i y_i[1 - y_i\boldsymbol{w}^\mathsf{T}\boldsymbol{x_i} > 0]\boldsymbol{x_i} \tag{2.23}$$

where [.] is the Iverson bracket that is 1, if $1 - y_i \boldsymbol{w}^\mathsf{T} \boldsymbol{x_i} > 0$ holds, 0 otherwise.

An observation that provides an intuition for Eq.(2.23) is that the samples inside their class boundary (*i.e.* $y_i \boldsymbol{w}^\mathsf{T} \boldsymbol{x_i} \geq 1$) do not play any explicit role in shaping the decision boundaries. This property leads to a certain robustness against outliers. It should be noted that all training samples are considered to determine the decision boundary in LDA and logistic regression.

2.3.4 Structured (Output) SVM (SSVM)

So far, we have considered SVMs only for binary decisions – *i.e.* $y \in \{-1, 1\}$. Structured SVMs [Tsochantaridis et al., 2004, Taskar et al., 2005] extend the SVMs to more general structured output spaces. In this case, the output can be a combination of multiple binary decisions such as a background/foreground decision for each pixel in an image or more structured outputs such as a parsing tree or a bounding box. Multiple binary outputs can naively be represented by multiple independent SVMs by ignoring the dependency between the output labels or by an exponential number of SVMs that explicitly models the interdependence between the labels. The core contribution of the SSVMs is to model those dependencies without using an exponential number of parameters.

Differently from the binary SVMs, the input $\boldsymbol{x}$ is replaced by the joint feature representation $\boldsymbol{\psi}(\boldsymbol{x}, \boldsymbol{y})$ in the SSVMs. $\boldsymbol{\psi}(\boldsymbol{x}, \boldsymbol{y})$ is not only characterized by the input $\boldsymbol{x}$ but also by the structured output $\boldsymbol{y}$. The compatibility between an input $\boldsymbol{x}$ and output $\boldsymbol{y}$ can be written as an energy function:

$$E(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{w}) = \boldsymbol{w}^\mathsf{T} \boldsymbol{\psi}(\boldsymbol{x}, \boldsymbol{y}), \tag{2.24}$$

where $\boldsymbol{w}$ is the parameter model that we want to learn. Given an input $\boldsymbol{x}$, the probability of an output $\boldsymbol{y}$ is given by:

$$p(\boldsymbol{y}|\boldsymbol{x}) = \frac{1}{Z(\boldsymbol{x})} \exp(E(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{w})), \tag{2.25}$$

where $Z(\boldsymbol{x})$ is the normalizing term or the *partition function*:

$$Z(\boldsymbol{x}) = \sum_{\hat{\boldsymbol{y}} \in \mathcal{Y}} (\exp(E(\boldsymbol{x}, \hat{\boldsymbol{y}}, \boldsymbol{w}))). \tag{2.26}$$

In order to learn the parameter vector $\boldsymbol{w}$, one can maximize the likelihood of the ground truth output $\boldsymbol{y}_i$ for input $\boldsymbol{x}_i$:

$$\arg\max_{\boldsymbol{w}} \log p(\boldsymbol{y}_i|\boldsymbol{x}_i, \boldsymbol{w}). \tag{2.27}$$

For training an SVM in structured output spaces, we use the margin-rescaling structured SVM formulation [Tsochantaridis et al., 2004, Taskar et al., 2005] which requires minimization of the objective function or *regularized risk* $J(\boldsymbol{w})$ on the training set in form of:

$$\min_{\boldsymbol{w}} J(\boldsymbol{w}) := \min_{\boldsymbol{w}} \frac{1}{2}||\boldsymbol{w}||^2 + C\sum_i \Delta(\boldsymbol{y}_i, \hat{\boldsymbol{y}}_i) \tag{2.28}$$

where C is the trade-off parameter and $\Delta(y, \hat{y})$ denotes the *loss function* that penalizes mismatches between the ground truth output label $\boldsymbol{y}$ and predicted label $\hat{\boldsymbol{y}}$. A common loss measure for scalar output values, the zero-one loss is given by $\Delta(y, y) = 0$ and $\Delta(y, \hat{y}) = 1$, when $\hat{y} \neq y$. However, Δ is typically not convex nor continuous and optimizing it can be computationally expensive. In order to overcome the problem the SSVM replaces the loss function with a piecewise linear convex upper bound:

$$\min_{\boldsymbol{w}} \frac{1}{2}||\boldsymbol{w}||^2 + C\sum_i \{(\max_{\hat{\boldsymbol{y}}_i \in \mathcal{Y}} \boldsymbol{w}^\mathsf{T}\boldsymbol{\psi}(\boldsymbol{x_i}, \hat{\boldsymbol{y}}_i) + \Delta(\boldsymbol{y}_i, \hat{\boldsymbol{y}}_i)) - \boldsymbol{w}^\mathsf{T}\boldsymbol{\psi}(\boldsymbol{x_i}, \boldsymbol{y}_i)\}. \tag{2.29}$$

The proceeding optimization algorithm finds the *(potentially) most violating constraints* [Tsochantaridis et al., 2004], involving some output values $\hat{\boldsymbol{y}}_i$ and uses them to learn the parameter vector $\boldsymbol{w}$.

After learning the SSVM model $\boldsymbol{w}$, we can predict the output of an input $\boldsymbol{x}$ by using the rule:

$$g_{\boldsymbol{w}}(\boldsymbol{x}) = \arg\max_{\boldsymbol{y} \in \mathcal{Y}} \boldsymbol{w}^\mathsf{T}\boldsymbol{\psi}(\boldsymbol{x}, \boldsymbol{y}). \tag{2.30}$$

In the next chapters of this manuscript, we will formulate our learning problems in the binary or multi-class setting rather than the complex structured output spaces. However, the presented SSVM formulation also provides a principled way for the multi-class classification experiments in this manuscript by jointly learning the class specific parameter vectors $\boldsymbol{w_y}$:

$$E(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{w}) = \boldsymbol{w}_{\boldsymbol{y}}^\mathsf{T}\boldsymbol{\phi}(\boldsymbol{x}), \tag{2.31}$$

where $\boldsymbol{w} = [\boldsymbol{w}_1^\mathrm{T}, \ldots, \boldsymbol{w}_K^\mathrm{T}]^\mathrm{T}$ is a stack of vectors, $\boldsymbol{w}_K$ being a parameter vector associated with the K-th class. $\boldsymbol{\phi}(\boldsymbol{x}) \in \mathcal{R}^D$ denotes an arbitrary input representation.

2.3.5 Latent SSVM (LSSVM)

In many applications, the input-output relationship cannot be explained by only the $(\boldsymbol{x}, \boldsymbol{y})$ pairs in the training set but also depends on a set of unobserved

latent variables $\boldsymbol{h} \in \mathcal{H}$. For instance, in machine translation, the translation $\boldsymbol{y}$ of a sentence $\boldsymbol{x}$ depends on the linguistic structure $\boldsymbol{h}$ of the sentence (*e.g.* parse trees, word alignments). Similarly, the label $\boldsymbol{y}$ of an object in an image $\boldsymbol{x}$ can be characterized by not only its appearance but also the location of the object $\boldsymbol{h}$.

In the sequel, we closely follow the notation proposed by Yu and Joachims [Yu and Joachims, 2009]. The prediction rule in Eq.(2.30) is rewritten for the LSSVM as

$$g_{\boldsymbol{w}}(\boldsymbol{x}) = \underset{\boldsymbol{y}\in\mathcal{Y},\boldsymbol{h}\in\mathcal{H}}{\arg\max}\, \boldsymbol{w}^{\mathsf{T}}\boldsymbol{\psi}(\boldsymbol{x},\boldsymbol{y},\boldsymbol{h}) \tag{2.32}$$

where $\boldsymbol{\psi}(\boldsymbol{x},\boldsymbol{y},\boldsymbol{h})$ is the joint feature representation with additional latent parameter vector $\boldsymbol{h}$.

Training the LSSVM model requires to solve the optimization problem in (2.31) with additional latent variables:

$$\begin{aligned}\min_{\boldsymbol{w}} \frac{1}{2}||\boldsymbol{w}||^2 + C\sum_i \{(\max_{\substack{\hat{\boldsymbol{y}}_i\in\mathcal{Y}\\ \hat{\boldsymbol{h}}_i\in\mathcal{H}}} \boldsymbol{w}^{\mathsf{T}}\boldsymbol{\psi}(\boldsymbol{x_i},\hat{\boldsymbol{y}}_i,\hat{\boldsymbol{h}}_i) + \Delta(\boldsymbol{y}_i,\hat{\boldsymbol{y}}_i,\hat{\boldsymbol{h}}_i)) \\ - \max_{\boldsymbol{h}_i^*} \boldsymbol{w}^{\mathsf{T}}\boldsymbol{\psi}(\boldsymbol{x_i},\boldsymbol{y}_i,\boldsymbol{h}_i^*)\}.\end{aligned} \tag{2.33}$$

In contrast to the SSVM, including latent variables $\boldsymbol{h}$ makes the LSVM formulation in (2.33) non-convex. However, Yu and Joachims [Yu and Joachims, 2009] show that the objective function (2.33) can be decomposed into a concave and convex function and efficiently be solved by the concave-convex procedure (CCCP) [Yuille and Rangarajan, 2003] with convergence guarantee. In the next chapter, we will give more detailed explanation for the LSSVM optimization.

Chapter 3

Learning Spatial Pyramids

In this chapter, we address the variability in spatial location and size of objects by introducing a novel object representation. In particular we propose a generic framework to incorporate unobserved auxiliary information for classifying objects and actions. This framework allows us to automatically select a bounding box and its quadrants from which best to extract features. Parts of this chapter are published in the British Machine Vision Conference 2011 [Bilen et al., 2011] and published in the International Journal of Computer Vision [Bilen et al., 2013b].

In object detection, which includes the localization of object classes, people have trained their systems by giving bounding boxes around exemplars of a given class label. Here we show that the classification of object classes, i.e. the flagging of their presence without their localization, also benefits from the estimation of bounding boxes, even when these are not supplied as part of the training. The approach can also be interpreted as exploiting non-uniform pyramidal schemes. As a matter of fact, we demonstrate that similar schemes are also helpful for action category classification.

In this chapter we address the *classification* of objects (e.g. person or car) and actions (e.g. hugging or eating) [Pinz, 2005] in the sense of PASCAL VOC [Everingham et al., b], i.e. indicating their presence but not their spatial/temporal localization (the latter is referred to as detection in VOC parlance). The more successful methods are based on a uniform pyramidal representation built on a visual word vocabulary [Lazebnik et al., 2006, Wang et al., 2010, Boureau et al., 2010]. The focus then is often on the best features to use. In this chapter, we augment the classification through an orthogonal idea, i.e. by adding more flexible spatial information. This will be formulated more generally as inferring additional unobserved or 'latent'

dependent parameters. In particular, we focus on two such types of parameters:

- The first type specifies a cropping operation. This determines a bounding box in the image. This box serves to eliminate non-representative object parts and background.
- The second type specifies a splitting operation. It corresponds to a *non-uniform* image decomposition into 4 quadrants or temporal decomposition of a spatio-temporal volume into 2 video sub-sequences.

Apart from using these operations separately, we also study the effect of applying and jointly learning both these types of latent parameters, resulting in a bounding box which is also split. In any case, uniform grid subdivisions are replaced by more flexible operations.

3.1 Related Work

At the time of our initial work [Bilen et al., 2011], there was earlier work using latent variables, but typically for object detection and not classification [Felzenszwalb et al., 2010, Vedaldi and Zisserman, 2009, Blaschko et al., 2010]. A notable exception is a contribution by Nguyen *et al.* [Nguyen et al., 2009]. They proposed a method for joint localization (only cropping) and classification. We believe that our learning approach is more principled however, and we go beyond cropping by also offering splits and crop + split combinations. This comes with improved results. Moreover, we propose iterative learning for these non-convex optimization problems, thereby more successfully avoiding local minima, as well as an objective function that can better deal with unbalanced data sets. In the meantime, the use of latent variables has gained traction in the area of classification [Bilen et al., 2012, Sharma et al., 2012, Shapovalova et al., 2012].

While it is possible to learn our latent variables by using a separate routine [Satkin and Hebert, 2010], we adopt a principled max-margin method that jointly infers latent variables and class label. This we solve using a latent structural support vector machine (LSSVM) [Yu and Joachims, 2009]. Self-paced learning has recently been proposed as a further extension for the improved learning of latent SVMs [Kumar et al., 2010], but was not used here. Instead, we explore an extension of the LSSVM by initially limiting the latent variable parameter space and iteratively growing it. Moreover, we design a new objective function in the LSSVM formulation to more effectively learn in the case of unbalanced data sets, e.g. when having a significantly higher number of negative

images than positive ones. Those measures were observed to improve the classification results.

Our work can be seen as complementary to several alternative refinements to the bag-of-words principle. As a matter of fact, it could be combined with such work. For instance, improvements have also been obtained by considering multiple kernels of different features [Vedaldi et al., 2009, Gehler and Nowozin, 2009]. Another refinement has been based on varying the pyramidal representation step by considering maximal pooling over sparse continuous features [Wang et al., 2010, Boureau et al., 2010].

At a meta-level, recent progress in object classification has mainly been driven by the selection of more (sophisticated) features [Perronnin et al., 2010, Zhou et al., 2010b]. This has brought a couple of percentage points in terms of performance [Chatfield et al., 2011]. Our improvements can actually be combined with those, and are shown here to bring similar improvements on their own. Yet, our approach does this at a lower computational cost.

As to action classification, this has mainly followed a bag of words approach as well. Early work towards classification of actions using space-time interest points (STIP) [Laptev and Lindeberg, 2003] was proposed by Schüldt *et al.* [Schüldt et al., 2004]. A detailed evaluation of various features has been carried out lately by Wang *et al.* [Wang et al., 2009].

In summary, the main contributions of this chapter are a) the introduction of latent variables for enhanced classification, b) a principled technique for estimating them in the case of object and action classification, c) adapted optimization to improve learning in the case of imbalanced data sets, and d) the avoidance of local optima through an iteratively widened parameter space.

The remainder of the chapter is structured as follows. Section 3.2 describes the latent parameter operations and how they are included in the overall classification framework. Section 3.3 explains the inference and learning procedures. Section 3.4 shows how the LSSVM framework is adapted for imbalanced data sets. Section 3.5 introduces an iterative learning approach for these latent variables. Section 3.6 describes the results on standard object and action classification benchmarks and analyzes the statistical significance of the improved results. Section 3.7 concludes the chapter.

3.2 Latent Operations

We explore how far information resulting from cropped or splitted regions can serve classification. In order to see what is meant by those crop and split

operations, one can turn to Fig. 3.1 and Fig. 3.2 for the cases of single images (object classification) and videos (action classification), resp. Representative classification examples from the Graz-02 data set are shown in Fig. 3.3-3.6. We now discuss the two basic operations represented by our latent variables, cropping and splitting, in turn.

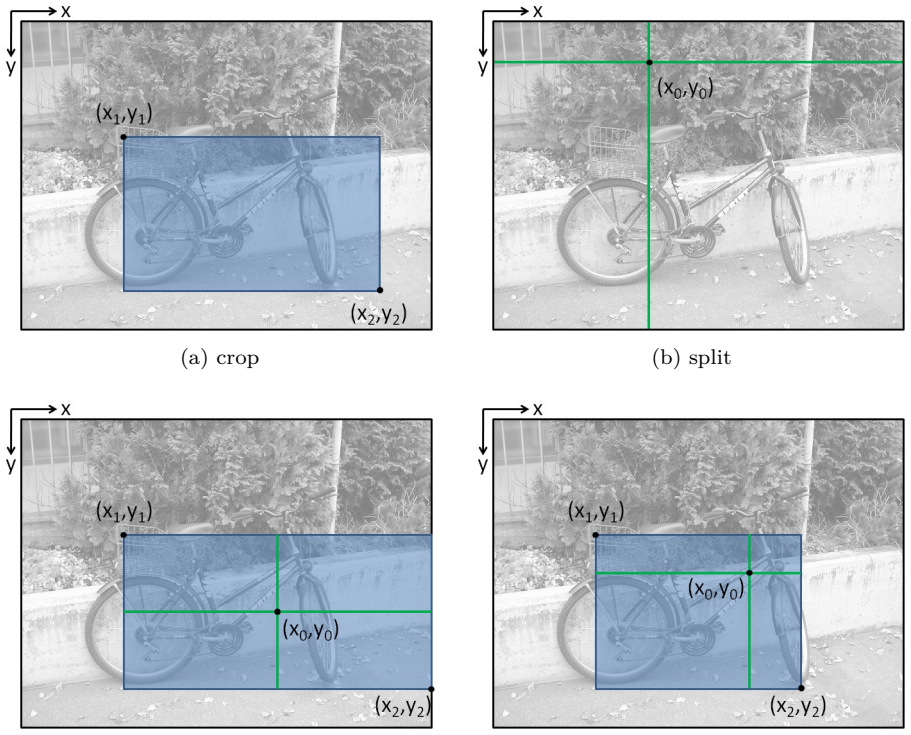
3.2.1 Crop

Our first latent operation builds on the motivation that including class related content and discarding irrelevant and confusing content should provide a better discriminant function for classification. For the sake of simplicity, we use a rectangular bounding box to separate the two parts. The bounding box is represented by two points for both spatial and temporal cropping. We denote the latent parameter set with $\boldsymbol{h}_{\text{crop}} = (x_1, y_1, x_2, y_2)$ and $\boldsymbol{h}_{\text{crop}} = (t_1, t_2)$ for images and video sequences respectively. Illustrations for cropping were shown in Fig. 3.1.(a) and Fig. 3.2.(a).

For the Graz-02 3-class person-car-bike examples in Fig. 3.3, we illustrate the derived cropping operations with blue drawn bounding boxes. Differently from object detection methods, our classification method is not required to localize objects accurately. Instead it can exploit bounding boxes to discard object parts that are not helpful for its particular classification task, while keeping the helpful ones in. The latter can very well include parts of the background (e.g. road for the car in Fig. 3.3.(c)-(d), building for the person in Fig. 3.3.(e)-(f)). On the other hand, parts with too much variation in their appearance or with a high uncertainty of being picked up by the selected features, can be left out of the box. Also a bounding box is allowed to include more than one object of the same class (Fig. 3.3.(b)).

3.2.2 Split

It is known that using pyramidal subdivisions of images or videos improves the classification of objects and actions [Lazebnik et al., 2006, Laptev et al., 2008]. Therefore, it stands to reason to also consider a pyramid-type subdivision, but with added flexibility. Rather than splitting an image uniformly into equal quadrants, we consider splitting operations that divide into unequal quadrants. In the same vein, we allow a video fragment to be temporally split into two sub-sequences, which are not halves. In contradistinction with cropping where all further analysis is confined to the selected bounding box, we will use all splitted portions as well as the entire image or video, i.e. a total of 5 portions for images and 3 for videos.



(a) crop

(b) split

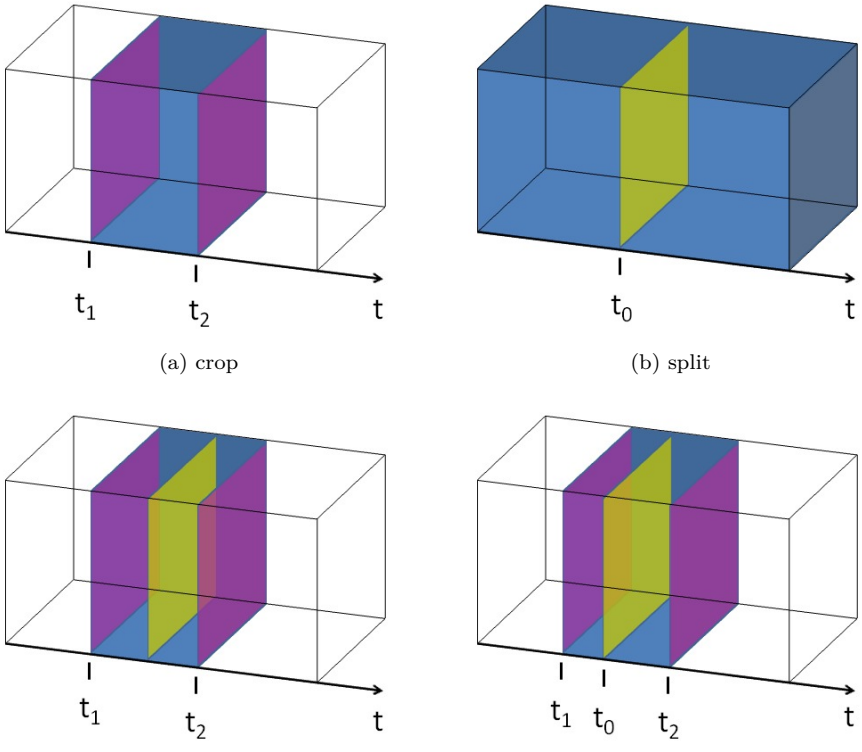
(c) crop-uniform split

(d) crop-split

Figure 3.1: Illustrative figure for latent operations, crop, split, crop-uniform split and crop-split on images. The crop-split operations have the most degree of freedom with six coordinates.

Note that in this chapter we only consider a single layer of subdivision of the pyramid, the extension to multi-layer pyramids is not covered yet. Hence, our splits are fully characterized by one point. We denote the latent variable set with $\boldsymbol{h}_{\text{split}} = (x_0, y_0)$ (Fig. 3.1.(b)) and $\boldsymbol{h}_{\text{split}} = (t_0)$ (Fig. 3.2.(b)) for images and videos, resp.

We show splitting samples for the bike, car and person classes with green crossing lines in Fig. 3.4. We observe that bikes are often located in the left and right bottom cells, while cars and people are usually splitted into four 'quadrants'.



(a) crop

(b) split

(c) crop-uniform split

(d) crop-split

Figure 3.2: Illustrative figure for latent operations, crop, split, crop-uniform split and crop-split on videos. Differently from spatial operations in images, the latent operations are performed only in the temporal domain.

3.2.3 Crop - Uniform Split

Our crop-uniform split operation learns a cropped region, which is then subdivided further into equal parts, in order to enrich the representation in pyramid-style. The latent parameter set is that of cropping. The combined operation is illustrated in Fig. 3.1.(c) and Fig. 3.2.(c). We illustrate crop-uniform splitting examples with blue cropping boxes and green uniform splits in Fig. 3.5. Fig. 3.5 heralds more effective model learning than through uniform splitting only. The richer representation of cropping and uniform splitting will in section 3.6 be seen to outperform pure cropping.

(a)

(b)

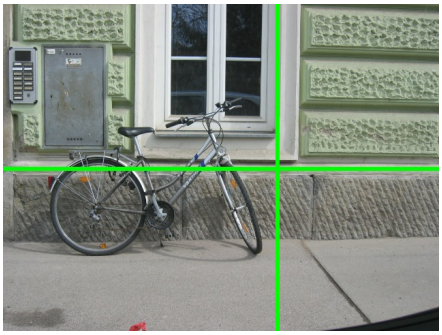
(c)

(d)

(e)

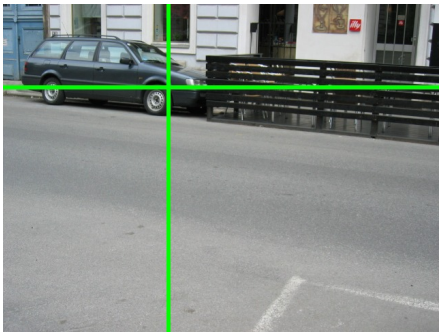
(f)

Figure 3.3: Crop examples for different object categories from the Graz-02 data set : (a) shows the eliminated non-representative object parts, (b) shows cropped region in the presence of multiple objects of the same class, (c)-(f) depict included background context in the bounding boxes. While the 'road' contains the context information for 'car', it is 'road' and 'building' for the 'person'.

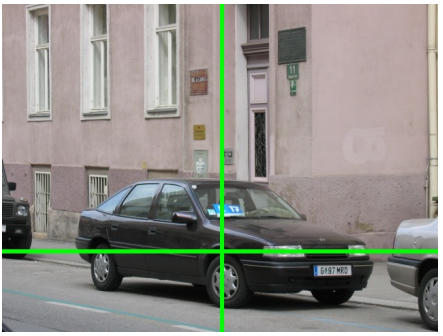


(a)

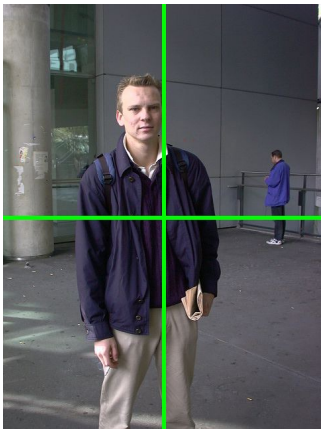
(b)



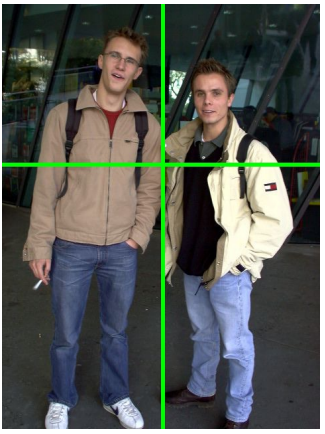
(c)



(d)

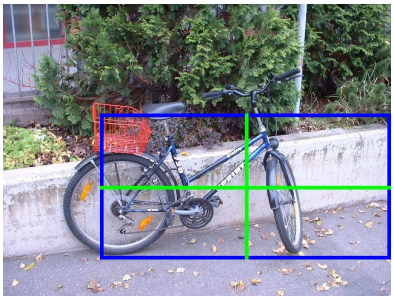


(e)



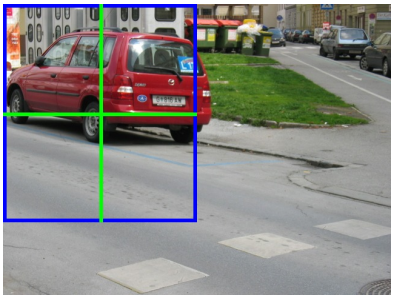
(f)

Figure 3.4: Representative split examples for the bike, car and person classes from the Graz-02 data set. The wheels of bikes in the shown images (a) and (b) are contained in the bottom left or right subdivisions. Splitting aligns the whole scene between the (c) and (d) examples. The upper quadrants contain buildings and windows of cars, while the lower ones contain road and wheels of cars. Since the split operation can only split the whole image into four divisions, it cannot exclude non-representative parts of images. In case of multiple objects, the splitting point can move to the visually dominant one (person) as in (e) or between two similar size objects (people) as in (f).

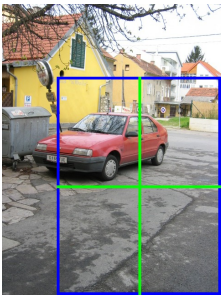


(a)

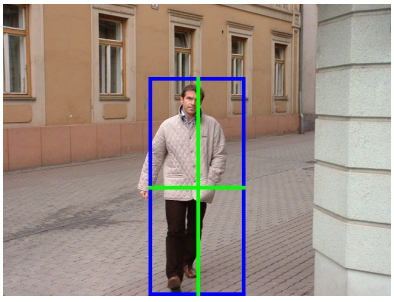
(b)



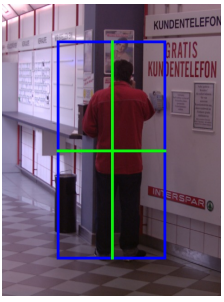
(c)



(d)

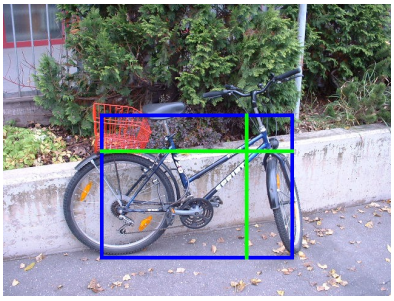


(e)

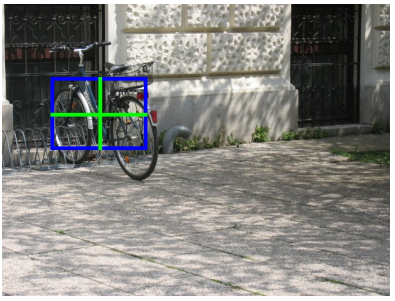


(f)

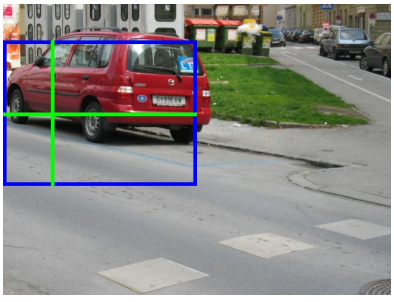
Figure 3.5: Representative crop-uniform split examples from the Graz-02 data set. (a) and (b) show coarse localizations of 'bikes' with uniform splitting. The (c) and (d) examples include 'cars' and 'road' in the upper and bottom subdivisions respectively. Differently from the strict bounding box concept in object detection tasks, the inferred image windows contain additional context information. Crop-uniform split achieves a coarse localization of 'person' in different (outdoor and indoor) environments in (e) and (f) respectively.



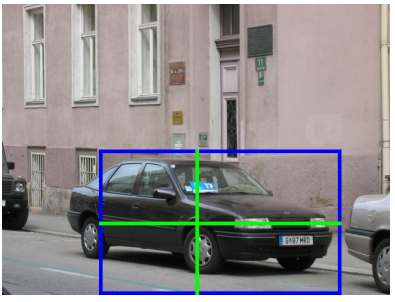
(a)



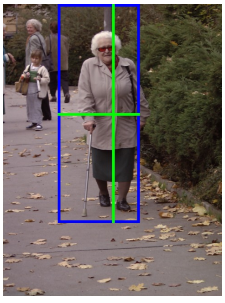
(b)



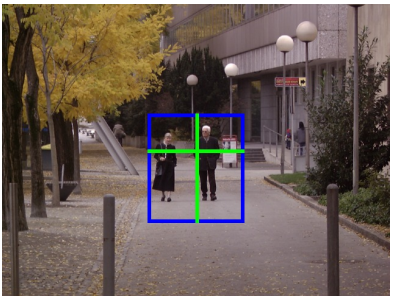
(c)



(d)



(e)



(f)

Figure 3.6: Representative crop-split examples from the Graz-02 data set. The crop-split is the most flexible operation and it can localize objects and align object parts better than the crop-uniform operation. The advantage of the crop-split over the crop-uni-split can be observed by comparing (a) to Fig. 3.5.(a). The crop-split achieves a better elimination of the background in the image (a). In the case of multiple objects, it picks the bigger person over the smaller ones in the background in (e). The image window in (f) contains two people that have similar sizes and are close to each other.

3.2.4 Crop-Split

The combined crop-split operation comes with the highest-dimensional latent parameter set of all four cases studied here. It learns both a cropping box and a non-uniform subdivision thereof. Its latent parameter set is a combination of the Cropping and Splitting operations, $\boldsymbol{h}_{\text{crop+split}} = (x_0, y_0, x_1, y_1, x_2, y_2)$ for images and $\boldsymbol{h}_{\text{crop+split}} = (t_0, t_1, t_2)$ for videos. The effect is illustrated in Fig. 3.1.(d) and Fig. 3.2.(d) resp. We illustrate crop-split examples with blue cropping boxes and green splits in Fig. 3.6. This figure already suggests that the crop-split model is able to roughly locate objects, although we do not use any ground truth bounding box locations during training.

3.3 Inference and Learning

We have introduced the notation for the latent SVMs (LSSVM) in Section 2.3.5. In the sequel, we follow the same notation and further detail inference and learning of the previously explained latent cropping/splitting operations for binary and multi-class classification tasks.

3.3.1 Inference

The inference problem corresponds to finding a prediction rule that infers a class label y and a set of latent parameters $\boldsymbol{h}$ for a previously unseen image. Formally speaking, the prediction rule $g_{\boldsymbol{w}}(\boldsymbol{x})$ maximizes a discriminant function $f_{\boldsymbol{w}}(\boldsymbol{x}, y, \boldsymbol{h})$ over y and $\boldsymbol{h}$ given the parameter vector $\boldsymbol{w}$ and the image $\boldsymbol{x}$, $f_{\boldsymbol{w}}(\boldsymbol{x}, y, \boldsymbol{h})$ measures the matching quality between input, output and latent parameters:

$$f_{\boldsymbol{w}}(\boldsymbol{x}, y, \boldsymbol{h}) = \boldsymbol{w}^{\mathsf{T}} \boldsymbol{\psi}(\boldsymbol{x}, y, \boldsymbol{h}) \tag{3.1}$$

where $\boldsymbol{\psi}(\boldsymbol{x}, y, \boldsymbol{h})$ is a joint feature vector. We use different $\boldsymbol{\psi}$ vectors for multi-class and binary classification tasks. The feature vector for multi-class setting is

$$\boldsymbol{\psi}_{\text{multi}}(\boldsymbol{x}, y, \boldsymbol{h}) = (\ \mathbf{0}^D \quad \dots \quad \mathbf{0}^D \quad \boldsymbol{\varphi}(\boldsymbol{x}, \boldsymbol{h}) \quad \mathbf{0}^D \quad \dots \quad \mathbf{0}^D\ )^{\mathsf{T}} \tag{3.2}$$

where $y \in \{1, \dots, k\}$ and $\boldsymbol{\varphi}(\boldsymbol{x}, \boldsymbol{h}) \in \mathcal{R}^D$ is a histogram of quantized features, given a latent parameter set, e.g. $\boldsymbol{h}_{\text{crop}}$ or $\boldsymbol{h}_{\text{split}}$. $\mathbf{0}^D$ denotes a D-dimensional zero row vector. $\boldsymbol{\varphi}(\boldsymbol{x}, \boldsymbol{h})$ is stacked into position $y \times D$.

The feature vector for binary-class setting is

$$\boldsymbol{\psi}_{\text{bin}}(\boldsymbol{x}, y, \boldsymbol{h}) = \begin{cases} \boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{h}) = (\ \boldsymbol{\varphi}(\boldsymbol{x}, \boldsymbol{h}) \quad \mathbf{0}^D\ )^T, \text{ if } y = 1 \\ -\boldsymbol{\phi}(\boldsymbol{x}) = (\ \mathbf{0}^D \quad -\boldsymbol{\varphi}(\boldsymbol{x})\ )^T, \text{ if } y = -1 \end{cases} \tag{3.3}$$

where $y \in \{-1, 1\}$ ($y = 1$ meaning the class is present in the image and $y = -1$ it is not) and $\boldsymbol{\varphi}(\boldsymbol{x})$ is the feature representation for the whole image. While $\boldsymbol{\psi}_{\text{multi}}$ is $K \times D$ dimensional (K denotes the number of classes), $\boldsymbol{\psi}_{\text{bin}}$ is $2 \times D$. Differently from the multi-class case, we learn to localize only in positive images and fix the image window to the whole image to represent negative images for the binary case. However, this is not the only possible representation, one can also localize in negative images similarly to positive images or set all the elements of feature vector of negative images to zero as in [Zhu et al., 2010].

The prediction rule $g_{\boldsymbol{w}}$ can be obtained by maximizing the discriminant function over label and latent space:

$$g_{\boldsymbol{w}}(x) = \underset{\hat{y} \in \mathcal{Y}, \hat{\boldsymbol{h}} \in \mathcal{H}}{\arg\max}\ f_{\boldsymbol{w}}(\boldsymbol{x}, \hat{y}, \hat{\boldsymbol{h}}). \tag{3.4}$$

3.3.2 Learning

Suppose we are given a set of training samples $X = \{\boldsymbol{x_1}, \ldots, \boldsymbol{x_n}\}$ and their labels $Y = \{y_1, \ldots, y_n\}$ and we want to learn a SVM model w to predict the class label of an unseen example. We also use latent parameters $H = \{\boldsymbol{h}_1, \ldots, \boldsymbol{h}_n\}$ to select the cropping and/or splitting operations that add spatial information to the classifier, as introduced in section 3.2. In cases where the set of spatial parameters $\boldsymbol{h}_i$ is also specified in the training set (as with training for detection), the standard structural SVM [Tsochantaridis et al., 2004] solves the following optimization problem:

$$\min_{\boldsymbol{w}} \left[\frac{1}{2} \|\boldsymbol{w}\|^2 + C \sum_{i=1}^{n} \left[\max_{\hat{y}_i, \hat{\boldsymbol{h}}_i} \left[\boldsymbol{w}^\mathsf{T} \boldsymbol{\psi}(\boldsymbol{x_i}, \hat{y}_i, \hat{\boldsymbol{h}}_i) + \Delta(y_i, \hat{y}_i, \boldsymbol{h}_i, \hat{\boldsymbol{h}}_i) \right] - \boldsymbol{w}^\mathsf{T} \boldsymbol{\psi}(\boldsymbol{x_i}, y_i, \boldsymbol{h}_i) \right] \right] \tag{3.5}$$

where C is the penalty parameter and $\Delta(y_i, \hat{y}_i, \boldsymbol{h}_i, \hat{\boldsymbol{h}}_i)$ is the loss function.

For the case of classification, the bounding boxes will typically not come with the training samples however, and need to be treated as latent parameters. To

solve the optimization problem in (3.5) without the labeled windows, we follow the latent SVM formulation of [Yu and Joachims, 2009]:

$$\min_{\boldsymbol{w}} \left[\frac{1}{2} \|\boldsymbol{w}\|^2 + C \sum_{i=1}^{n} \left[\max_{\hat{y}_i, \hat{\boldsymbol{h}}_i} \left[\boldsymbol{w}^\mathsf{T} \boldsymbol{\psi}(\boldsymbol{x_i}, \hat{y}_i, \hat{\boldsymbol{h}}_i) + \Delta(y_i, \hat{y}_i) \right] - \max_{\hat{\boldsymbol{h}}_i} \left[\boldsymbol{w}^\mathsf{T} \boldsymbol{\psi}(\boldsymbol{x_i}, y_i, \hat{\boldsymbol{h}}_i) \right] \right] \right] \tag{3.6}$$

Note that we remove $\boldsymbol{h}_i$ from Δ since it is not given. In the multi-class classification task, we use the 0-1 loss which is $\Delta(y_i, \hat{y}_i) = 1$ if $\hat{y}_i \neq y_i$, and else 0. We will explain the loss function that is designed for binary classification in section 3.4.

The latent SVM formulation can be rewritten as the difference of two convex functions. Note that these terms are convex in terms of $\boldsymbol{w}$.

$$\min_{\boldsymbol{w}} \left[\underbrace{\left[\frac{1}{2} \|\boldsymbol{w}\|^2 + C \sum_{i=1}^{n} \max_{\hat{y}_i, \hat{\boldsymbol{h}}_i} \boldsymbol{w}^\mathsf{T} \boldsymbol{\psi}(\boldsymbol{x_i}, \hat{y}_i, \hat{\boldsymbol{h}}_i) + \Delta(y_i, \hat{y}_i) \right]}_{p(\boldsymbol{w})} - \left[\underbrace{C \sum_{i=1}^{n} \max_{\hat{\boldsymbol{h}}_i} \left[\boldsymbol{w}^\mathsf{T} \boldsymbol{\psi}(\boldsymbol{x_i}, y_i, \hat{\boldsymbol{h}}_i) \right]}_{q(\boldsymbol{w})} \right] \right] \tag{3.7}$$

The difference of those two functions, $p(\boldsymbol{w}) - q(\boldsymbol{w})$ can be solved by using the Concave-Convex Procedure (CCCP) [Yuille and Rangarajan, 2003], where p and q are convex. The generic CCCP algorithm is guaranteed to decrease the objective function (3.7) at each iteration t and to converge to a local minimum or a saddle point. In section 3.5 we suggest an iterative method for avoiding an undesired local minimum and saddle point in the first iterations. The CCCP algorithm to minimize the difference of two convex functions works as described in the next section.

3.3.3 Algorithm

Initialize the algorithm by setting $t = 0$ and all elements of $\boldsymbol{w}_0$ to zero.

Iterate:

1. Compute hyperplane $\boldsymbol{v_t}$ such that $-q(\boldsymbol{w}) \leq -q(\boldsymbol{w_t}) + (\boldsymbol{w} - \boldsymbol{w_t})^\mathsf{T} \boldsymbol{v_t}$ for all $\boldsymbol{w}$.
2. Solve $\boldsymbol{w_{t+1}} = \arg\min_{\boldsymbol{w}} \left(p(\boldsymbol{w}) + \boldsymbol{w}^\mathsf{T} \boldsymbol{v}_t \right)$
3. Set $t = t + 1$

We iterate until the stopping condition $[p(\boldsymbol{w_t}) - q(\boldsymbol{w_t})] - [p(\boldsymbol{w_{t-1}}) - q(\boldsymbol{w_{t-1}})] < \epsilon$. Note that t is typically small (10-100). The first step involves the latent parameter inference problem

$$\boldsymbol{h}_i^* = \arg\max_{\hat{\boldsymbol{h}}_i \in \mathcal{H}} \boldsymbol{w_t}^\mathsf{T} \boldsymbol{\psi}(\boldsymbol{x_i}, y_i, \hat{\boldsymbol{h}}_i). \tag{3.8}$$

Computing the new $\boldsymbol{w}_{t+1}$ in the second line involves solving the standard Structural SVM problem [Tsochantaridis et al., 2004] with the inferred latent variables $\boldsymbol{h}_i^*$:

$$\begin{aligned} \arg\min_{\boldsymbol{w}} \quad & \tfrac{1}{2} \|\boldsymbol{w}\|^2 + C \textstyle\sum_{i=1}^n \max_{\hat{y}_i, \hat{\boldsymbol{h}}_i} \left(\boldsymbol{w}^\mathsf{T} \boldsymbol{\psi}(\boldsymbol{x_i}, \hat{y}_i, \hat{\boldsymbol{h}}_i) + \Delta(y_i, \hat{y}_i) \right) \\ & - C \textstyle\sum_{i=1}^n \left(\boldsymbol{w}^\mathsf{T} \boldsymbol{\psi}(\boldsymbol{x_i}, y_i, \boldsymbol{h}_i^*) \right) \end{aligned} \tag{3.9}$$

Solving the formula (3.9) requires to compute the constraint

$$\{y_i^*, \boldsymbol{h}_i^*\} = \arg\max_{\hat{y}_i, \hat{\boldsymbol{h}}_i} \left(\boldsymbol{w}^\mathsf{T} \boldsymbol{\psi}(\boldsymbol{x_i}, \hat{y}_i, \hat{\boldsymbol{h}}_i) + \Delta(y_i, \hat{y}_i) \right) \tag{3.10}$$

for each sample. In the literature, this term is called *most violated constraint* in [Tsochantaridis et al., 2004] or *loss augmented inference* in [Taskar et al., 2005]. It corresponds to the most confusing response from another than the actual class or another latent parameter than the inferred one.

The CCCP applied to the LSSVM leads to a very intuitive Expectation Maximization (EM) kind of algorithm. The procedure alternates between imputing the best latent variables $\boldsymbol{h}_i^*$ for the training pair $\boldsymbol{x}_i, y_i$ and solving the Structural SVM optimization problem by treating the latent variables as observed. However it minimizes the regularized loss against the single latent variable $\boldsymbol{h}_i^*$, while EM maximizes the expected likelihood under the distribution of the latent variables.

3.4 Optimizing AUC

Multi-class classification performances are typically measured in terms of accuracy, e.g. correctly classified images over total number of images. While this

evaluation criterion is informative in the multi-class setting, it can be misleading in binary classification, as the number of positive and negative images are unbalanced. This imbalance increases a lot more in the case of latent window parameters as we deal with more negative samples (all other bounding boxes in an image are considered negative). The area under the ROC curve (AUC), average precision (AP) and Precision at fixed recall give a more intuitive and sensitive evaluation in this case.

We evaluate our proposed classifiers in section 3.6 on various benchmarks including the PASCAL VOC 2007 data set [Everingham et al., b] which uses the AP to judge the classification performances. While it is possible to train our classifiers on the basis of accuracy loss and then report testing performance using the AP, Joachims [Joachims, 2005] shows that such difference may result in a suboptimal performance. To the best of our knowledge, there is no prior work which optimizes a structural SVM with latent parameters based on the exact AP measure. Although it has been shown that it is possible to optimize a classifier based on the approximated AP with the Structural SVM [Yue et al., 2007] or to factorize the optimization problem based on dual decomposition [Ranjbar et al., 2012], optimizing both the classifier and the latent parameters with a Structural SVM proved difficult. Therefore, we will train our classifiers using the AUC criterion, which optimizes for a ranking between positive and negative samples similar to the AP and helps to improve performance even when testing on AP. The proposed learning algorithm does not require any extra parameter to weight negative samples, does not worsen computational complexity compared to training on the basis of accuracy loss, and does improve the classification performance. We report our results on the PASCAL VOC 2007 data set and compare the AUC optimized classifiers to the accuracy based baselines in section 3.6.

The area under the ROC curve can be computed from the number of positive and negative pairs which are ranked in the wrong order, i.e.:

$$\text{AUC} = 1 - \frac{|\text{Swapped Pairs}|}{n^+ \cdot n^-} \tag{3.11}$$

where n^+ and n^- are the number of positive and negative samples respectively and $\text{Swapped Pairs} = \left\{(i,j) \;:\; y_i > y_j \;\wedge\; r(\boldsymbol{x}_i) < r(\boldsymbol{x}_j)\right\}$ with a ranking function $(r(x))$. We design the ranking function $(r(\boldsymbol{x}))$ based on the binary representation in (3.3) as the maximum response for $\boldsymbol{\psi}_{\text{bin}}(\boldsymbol{x}, 1, \boldsymbol{h}) - \boldsymbol{\psi}_{\text{bin}}(\boldsymbol{x}, -1, \boldsymbol{h})$:

$$r(\boldsymbol{x}) = \max_{\hat{\boldsymbol{h}}} \boldsymbol{w}^{\mathsf{T}}(\boldsymbol{\phi}(\boldsymbol{x}, \hat{h}) + \boldsymbol{\phi}(\boldsymbol{x})) \tag{3.12}$$

Using the ranking function (3.12), we can rewrite the swapped pairs that are used to compute the AUC as

$$\begin{aligned}\text{Swapped Pairs} = \Big\{(i,j):\ y_i = 1, y_j = -1 \text{ and} \\ \max_{\hat{\boldsymbol{h}}_{ij}} \boldsymbol{w}^\mathsf{T}[\boldsymbol{\phi}(\boldsymbol{x_i}, \hat{\boldsymbol{h}}_{ij}) + \boldsymbol{\phi}(\boldsymbol{x_i})] < \max_{\hat{\boldsymbol{h}}_{ji}} \boldsymbol{w}^\mathsf{T}[\boldsymbol{\phi}(\boldsymbol{x_j}, \hat{\boldsymbol{h}}_{ji}) + \boldsymbol{\phi}(\boldsymbol{x_j})]\Big\}.\end{aligned} \tag{3.13}$$

where $\hat{\boldsymbol{h}}_{ij}$ and $\hat{\boldsymbol{h}}_{ji}$ denote the best latent parameter for image $\boldsymbol{x_i}$ on the left hand side and for image $\boldsymbol{x_j}$ on the right hand side respectively.

In order to incorporate the ranking into the latent structural SVM problem, we design the feature vector $\boldsymbol{\psi}$ by substituting individual samples x with positive-negative pairs $\tilde{\boldsymbol{x}}$:

$$\boldsymbol{\psi}(\tilde{\boldsymbol{x}}_{ij}, \tilde{y}_{ij}, \tilde{\boldsymbol{h}}_{ij}) = \begin{cases} \boldsymbol{\phi}(\boldsymbol{x_i}, \tilde{\boldsymbol{h}}_{ij}) - \boldsymbol{\phi}(\boldsymbol{x_j}), \text{ if } \tilde{y}_{ij} = 1 \\ \boldsymbol{\phi}(\boldsymbol{x_j}, \tilde{\boldsymbol{h}}_{ij}) - \boldsymbol{\phi}(\boldsymbol{x_i}), \text{ if } \tilde{y}_{ij} = -1 \end{cases} \tag{3.14}$$

where $\tilde{\boldsymbol{x}}_{ij} = (\boldsymbol{x_i}, \boldsymbol{x_j})$ and $\tilde{y}_{ij} = \begin{cases} 1, \text{ if } y_i = 1, y_j = -1 \\ -1, \text{ if } y_i = -1, y_j = 1 \end{cases}$. Given the label pair $\tilde{y}_{ij}$, $\tilde{\boldsymbol{h}}_{ij}$ denotes a latent parameter for image $\boldsymbol{x_i}$ when $(\tilde{y}_{ij} = 1)$ or for image $\boldsymbol{x_j}$ when $(\tilde{y}_{ij} = -1)$ respectively. Please note that we discard positive-positive and negative-negative pairs in our training, since the AUC is only related to the ranking between positive and negative samples.

The error between the ground truth label set $\tilde{Y} = \{1, \cdots, 1\}$ and the prediction $\hat{\tilde{Y}} = \{\hat{\tilde{y}}_{ij}\}$ is proportional to (1 − AUC) of the original X and Y where $X = \{\boldsymbol{x_1}, \cdots, \boldsymbol{x_n}\}$ and $Y = \{y_1, \cdots, y_n\}$.

$$\Delta_{\text{AUC}}(\tilde{Y}, \hat{\tilde{Y}}) = \sum_{i=1}^{n^+} \sum_{j=1}^{n^-} \frac{1}{2}(1 - \hat{\tilde{y}}_{ij}) \tag{3.15}$$

Since the loss function in (3.15) decomposes linearly over the pairwise relationship (y_i, y_j), the most violated constraint $(\tilde{y}^*_{ij}, \tilde{\boldsymbol{h}}^*_{ij})$ can be computed for each pair individually:

$$\sum_{i=1}^{n^+} \sum_{j=1}^{n^-} \arg\max_{\hat{\tilde{y}}_{ij}, \hat{\boldsymbol{h}}_{ij}} \boldsymbol{w}^\mathsf{T} \boldsymbol{\psi}(\tilde{\boldsymbol{x}}_{ij}, \hat{\tilde{y}}_{ij} \hat{\tilde{\boldsymbol{h}}}_{ij}) + \frac{1}{2}(1 - \hat{\tilde{y}}_{ij}). \tag{3.16}$$

The most violated constraint computation for a given image pair $\tilde{\boldsymbol{x}}_{ij} = (\boldsymbol{x_i}, \boldsymbol{x_j})$ and corresponding label $y_{ij} = 1$ requires to check the inequality:

$$\max_{\hat{\boldsymbol{h}}_{ij}} \boldsymbol{w}^\mathsf{T} \Big[\boldsymbol{\phi}(\boldsymbol{x_i}, \hat{\boldsymbol{h}}_{ij}) + \boldsymbol{\phi}(\boldsymbol{x_i})\Big] < \max_{\hat{\boldsymbol{h}}_{ij}} \boldsymbol{w}^\mathsf{T} \Big[\boldsymbol{\phi}(\boldsymbol{x_j}, \hat{\boldsymbol{h}}_{ij}) + \boldsymbol{\phi}(\boldsymbol{x_j})\Big] + 1 \tag{3.17}$$

On the other hand, using the accuracy (0-1) loss and the feature representation in (3.3) leads to the following constraint computation which only considers responses from individual samples:

$$\begin{aligned} \max_{\hat{\boldsymbol{h}}_i} \boldsymbol{w}^\mathsf{T} \boldsymbol{\phi}(x_i, \hat{\boldsymbol{h}}_i) &< -\boldsymbol{w}^\mathsf{T} \boldsymbol{\phi}(x_i) + 1, \qquad \text{if } y_i = 1 \\ -\boldsymbol{w}^\mathsf{T} \boldsymbol{\phi}(x_i) &< \max_{\hat{\boldsymbol{h}}_i} \boldsymbol{w}^\mathsf{T} \boldsymbol{\phi}(x_i, \hat{\boldsymbol{h}}_i) + 1, \quad \text{if } y_i = -1. \end{aligned} \tag{3.18}$$

In practice, computing (3.17) for each pair does not add any significant computation load since $\max_{\hat{\boldsymbol{h}}_i} (\boldsymbol{w}^\mathsf{T} \boldsymbol{\phi}(\boldsymbol{x_i}, \hat{\boldsymbol{h}}_i))$ and $(\boldsymbol{w}^\mathsf{T} \boldsymbol{\phi}(\boldsymbol{x_i}))$ can be precomputed for each sample $(\boldsymbol{x_i})$ individually.

We can now write the latent SVM formulation in (3.7) for the AUC optimization. To do so, we define the convex functions $p(\boldsymbol{w})$ and $q(\boldsymbol{w})$ for brevity, and their difference can be used to compute the complete formulation. $p(\boldsymbol{w})$ is written as sum of a regularization term and (3.16):

$$p(\boldsymbol{w}) = \frac{1}{2} \|\boldsymbol{w}\|^2 + C \left[\sum_{i=1}^{n^+} \sum_{j=1}^{n^-} \max_{\hat{\tilde{y}}_{ij}, \hat{\boldsymbol{h}}_{ij}} \boldsymbol{w}^\mathsf{T} \boldsymbol{\psi}(\tilde{\boldsymbol{x}}_{ij}, \hat{\tilde{y}}_{ij}, \hat{\boldsymbol{h}}_{ij}) + \frac{1}{2}(1 - \hat{\tilde{y}}_{ij}) \right]. \tag{3.19}$$

In contrast to $p(\boldsymbol{w})$, the second convex function $q(\boldsymbol{w})$ can be computed linearly in terms of individual samples $(\boldsymbol{x})$ by using the feature representation (3.14):

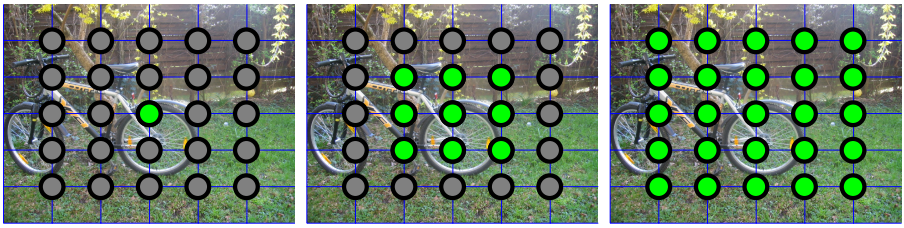
$$q(\boldsymbol{w}) = C \left[n^- \sum_{\substack{i, \\ y_i = 1}} \max_{\hat{\boldsymbol{h}}_i} \boldsymbol{w}^\mathsf{T} \boldsymbol{\phi}(\boldsymbol{x}_i, \hat{\boldsymbol{h}}_i) - n^+ \sum_{\substack{j, \\ y_j = -1}} \boldsymbol{w}^\mathsf{T} \boldsymbol{\phi}(\boldsymbol{x}_j) \right]. \tag{3.20}$$

So far, we have detailed the learning procedure that makes use of positive-negative image pairs $(\boldsymbol{x_i}, \boldsymbol{x_j})$ and penalizes ranking violations between those pairs. In parallel to the learning procedure, the prediction rule ranks images by using (3.12). The inference for an unseen image is rewritten as

$$g_{\text{AUC}}(\boldsymbol{x}) = \begin{cases} y^* = 1, \text{ if } \max_{\hat{\boldsymbol{h}}} \boldsymbol{w}^\mathsf{T} (\boldsymbol{\phi}(\boldsymbol{x}, \hat{\boldsymbol{h}}) + \boldsymbol{\phi}(\boldsymbol{x})) > 0 \\ y^* = -1, \text{ else.} \end{cases} \tag{3.21}$$

3.5 Iterative Learning of Latent Parameters

Learning the parameters of an LSSVM model often requires solving a non-convex optimization problem. Like every such problem, LSSVM is also prone



(a) `iter 0` (b) `iter 1` (c) `iter 2`

Figure 3.7: Illustration of the splitting operation in iterative learning. The green and gray nodes show the points where splitting is considered. At `iter 0` the image can only be splitted with horizontal and vertical lines through the image center, while at the next iteration `iter 1`, the image can be splitted with one of the 9 green nodes. At the last iteration `iter 2`, all splitting nodes are eligible.

to getting stuck in local minima. Recent work [Bengio et al., 2009] proposes an iterative approach to find better local minima within shorter convergence times for non-convex optimization problems. It suggests to first train the learning algorithm with easy examples and to then gradually feed in more complex examples. This procedure is called curriculum learning. The main challenge of curriculum learning is to find a good measure to quantify the difficulty of samples.

In this chapter, we take the size of the parameter space as an indication of the complexity of the learning problem. Initially, we run the learning algorithm with a limited latent subspace and then gradually increase the latent parameter space. Fig. 3.7 illustrates such iterative learning for the splitting operation. The circles centered in the nodes of the grid indicate the possible splitting points, i.e. the latent parameter set for the splitting operation. The green nodes indicate, from left to right, the growing number of splitting points that the algorithm can choose from during subsequent iterations.

3.6 Experiments

We evaluate our system on four publicly available computer vision benchmarks, the Graz-02 [Opelt et al., 2006a], the PASCAL VOC 2007 [Everingham et al., b] and the Caltech 101 [Fei-Fei et al., 2004] data sets for object classification, and the activities of daily living data set [Messing et al., 2009] for action classification.

For the object classification experiments, we extract dense SIFT features [Lowe, 1999a] with a spatial stride of 2 pixels at four scales ($4, 6, 8$ and 10 pixels) by using the `vl_phow` function from the VLFeat toolbox [Vedaldi and Fulkerson, 2008]. For the action classification experiments, we use the HoF descriptors [Laptev et al., 2008] to describe detected Harris3D interest points [Laptev and Lindeberg, 2003]. We apply K-means to the randomly sampled 200,000 descriptors from the training images/videos to form the visual codebook. The computed visual words are then used to encode the descriptors with the LLC method [Wang et al., 2010]. For the LLC encoding, we set the number of nearest neighbors and the regularization parameter to 5 and 10^{-4} respectively. The codebook sizes are 1024, 8192, 2048 and 1000 for the Graz-02, VOC-07, Caltech-101 and the Activities data sets respectively (often following the sizes used by others, in order to allow for a fair comparison in the subsequent experiments).

We compare the performance of the proposed latent operations, 'crop', 'split', 'crop-uni-split', 'crop-split' to the standard *bag-of-features* (BoW) and one level spatial pyramid (SP) [Lazebnik et al., 2006]. The BoW represents an image/video with a histogram of quantized local features and thus discards the spatial/temporal layout of the image/video structure. The SP is a more extensive representation which incorporates spatial information into the features by using a pyramidal representation. In our experiments, we use a one level SP (1×1 for the top layer and 2×2 for the base) for images, and a similar SP for videos, where the base is only temporally divided. Similarly, the feature dimensionality of the 'split', 'crop-uni-split' and 'crop-split' operations are equal with the SP. The performance criterion is the mean multi-class classification accuracy for the Graz-02, Caltech-101 and the Activities data sets and mean AP (mAP) for the VOC-07.

Our latent learning implementation builds on the publicly available code of Yu and Joachims [Yu and Joachims, 2009]. The regularizing parameter C of the LSSVM is tuned for each latent operation (crop, split, etc.) on each data set (Graz, VOC-07, etc.) by using cross-validation (the interval $[10^2, 10^7]$ is sampled logarithmically). The other free parameter ϵ, the stopping criterion for the CCCP algorithm, is set to 10^{-1} and 10^{-3} for the multi-class and binary classification experiments, respectively.

The running time of the LSSVM experiments is dominated by computing the 'most violated constraint' which was introduced in section 3.4. We need to compute the response of each classifier by scanning the latent parameter space (e.g. all possible boxes for the cropping operation), to find the violated constraints. It would therefore have been possible to improve the running time by using the branch and bound algorithm [Lampert et al., 2008]. For the cropping, splitting, crop-uniform-splitting, and crop-splitting operations the

		Graz-02	VOC-07	Caltech-101	Activities
Baseline	BoW	87.0 ± 1.4	49.9	61.3 ± 0.9	79.3
	SP	88.1 ± 1.4	54.7	72.7 ± 1.2	88.0
Ours	crop	88.4 ± 1.1	51.8	62.2 ± 1.0	72.0
	split	88.6 ± 1.3	55.3	73.3 ± 1.0	88.0
	crop-uni-split	90.4 ± 1.9	56.3	**75.3 ± 0.7**	**90.7**
	crop-split	**90.6 ± 1.8**	**57.1**	74.9 ± 0.9	88.7

Table 3.1: The classification results on the Graz-02, PASCAL VOC 2007, Caltech-101 and the activities of daily living data set. The performance criterion is multi-class accuracy for the Graz-02, Caltech-101 and the activities of daily living data set in percentage. It is mean average precision in percentage for the PASCAL VOC 2007. The performance of the crop, split, crop-uniform split and crop-split operations are compared to the baselines: BoW and SP. All the classifiers are learnt with the iterative LSSVM. We use the AUC based optimization to train the baseline and proposed classifiers for the VOC-07 data set.

training of each class-specific classifier in the VOC 2007 experiments took 1 hour, 5 minutes, 30 minutes and 3 hours on a 16 CPU machine, resp. Training for the other data sets went faster, and in the same relative orders of magnitude for the different operations.

3.6.1 Graz-02 Dataset

The Graz-02 data set contains 1096 natural real-world images with three object classes: bikes, cars and people. This database includes a considerable amount of intra-class variation, varying illumination, occlusion, and clutter. We form 10 training and testing sets by randomly sampling 150 images from each object class for training and use the rest for testing. We report the mean and standard deviation of the classification accuracy for the 10 corresponding experiments, each time also averaging over the 3 classes.

Table 3.1 shows the multi-class classification results. The crop operation improves the classification performance over the BoW and the SP representation by around 1.45 and 0.35 %, respectively. The non-uniform split operation also achieves better classification performance than the uniform split (SP). The crop-split operation has more degrees of freedom than the crop-uni-split model and outperforms the crop-uni-split: where the latter improves the baseline SP method by 2.4 %, the former improves it by 2.6 %. The crop-split operation thereby

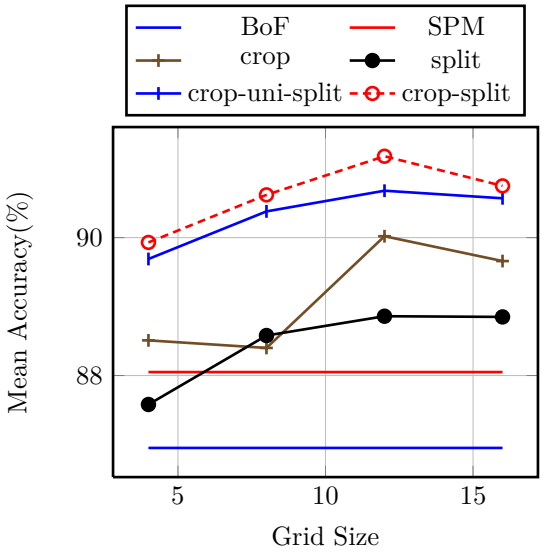


Figure 3.8: The mean classification accuracy on the Graz-02 data set with varying grid size. The grid size of 12 gives the best score for the crop, split, crop-uni-split and crop-split operations.

also gives the best result of all four operations. Adding splits systematically improved results over pure crops. This may not come as a surprise, as our implementation of splitting leads to substantially larger feature spaces (as SP does compared to BoW).

For cropping and splitting, we only consider points that lie on a regular grid. We now analyze the influence of the size of this grid on the classification accuracy. Fig. 3.8 plots the mean classification accuracy of the four proposed operations for the Graz-02 data set, and this for different grid sizes, i.e. 4×4, 8×8, 12×12, and 16×16. The results show that the performance of the classifiers increases with finer grids up to size 12, after which it slightly drops at 16. Hence, the optimal grid size on the Graz-02 data set is 12. Note that an increased grid size implies a significant, about quadratic, increase in computation time. We therefore report results for all other data sets with a grid size of 8.

	SP	crop-split
aeroplane	69.95	**72.76**
bicycle	59.62	**64.15**
bird	45.42	**46.10**
boat	64.39	**66.49**
bottle	**24.81**	24.22
bus	60.43	**65.57**
car	75.31	**78.64**
cat	57.45	**60.55**
chair	53.48	**55.02**
cow	42.87	**44.23**
dining-table	46.90	**48.70**
dog	**41.23**	41.01
horse	71.38	**73.33**
motorbike	62.70	**67.05**
person	82.44	**83.93**
pottedplant	**22.46**	21.38
sheep	43.54	**46.28**
sofa	49.58	**54.56**
train	70.92	**72.91**
tv	49.99	**54.06**
mean	54.74	**57.05**

Table 3.2: The classification results in terms of AP for each class of PASCAL VOC 2007. Both the SP and crop-split classifiers are trained with the iterative learning and AUC loss. The crop-split operation out-performs the SP in 17 out of 20 classes and the average improvement is 2.3% mAP.

3.6.2 PASCAL VOC 2007

The PASCAL VOC 2007 data set [Everingham et al., b] (VOC-07) contains 9,963 images which are split into training, validation and testing sets. The images are labeled with twenty classes, also allowing multiple classes to be present in the same image. We learn a one-vs-rest classifier for each class and report the mean Average Precision (mAP) which is the mean of AP values from each of the classifiers.

Table 3.1 depicts the classification results for the proposed operations. It should be noted that we use the AUC-loss based optimized classifiers for both the baseline and proposed latent operations to present a fair comparison. The 'crop'

operation yields an improvement of around 2% over the baseline BoW method to which it is similar in terms of feature space dimension. The 'split' operation improves the result over the SP method by 0.6%. The latent operations of 'crop-uni-split' and 'crop-split' provide further improvements over the SP and BoW baselines. Compared to SP, the 'crop-uni-split' operation yields an improvement of 1.5% and 'crop-split' one of 2.3%.

Table 3.2 shows the results for each object class individually for the crop-split operation. As can be observed from the results, we are able to improve the classification accuracy for 17 out 20 classes. In particular, the crop-split achieves substantial improvement for the 'bus' (5.1%), 'sofa' (5.0%), 'bicycle' (4.5%), 'motorbike' (4.3%) and 'tv monitor' (4%) categories. The method is not able to improve the accuracy for classes that are hard to localize because of their small size and cluttered background around them, such as 'bottle' and 'potted plant'.

So far, we have compared the proposed method to the SP which has the same feature dimensionality. We also show the classification results of the SP for different pyramid levels in Fig. 3.9. The subscript l of SP_l denotes the pyramid level such that it is composed of $2^0 \times 2^0 + 2^1 \times 2^1 + \ldots + 2^l \times 2^l$ D-dimensional histograms where D is 8192 for the experiments on the VOC-07. The plot shows that our baseline SP_1 gives the best result and using more pyramid levels does not improve the score in spite of the higher feature dimensionality. This can be explained with the fact that dividing an image with a fine grid produces more but smaller cells. Those small cells consist of few descriptors and thus they do not carry useful statistics.

3.6.3 Caltech-101 Dataset

The Caltech-101 data set [Fei-Fei et al., 2004] contains images of 101 object classes and an additional background class, i.e. 102 classes in total. The number of images per class varies from 31 to 800. We use 30 images for training from each class and use the rest of the images - as usual for this dataset with a maximum number of 50 - for testing. We run ten experiments on ten random divisions between training and testing images and report the mean accuracy and standard deviation for these runs.

Table 3.1 depicts the classification results for the Caltech 101 data set. The crop and split operations improve over the BoW and the SP baselines respectively as in the previous data sets. For this data set, where objects are always centered, the crop-uni-split operation achieves the highest performance among the proposed methods and improves the SP method by around 2.6%.

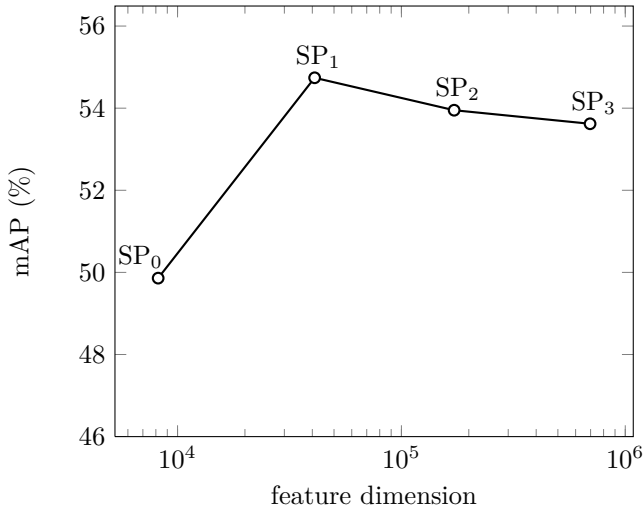


Figure 3.9: Classification results (mAP) with the AUC optimized SP for various pyramid levels on the VOC-07 versus the dimension of feature representation on the logarithmic scale. The subscript l of SP_l denotes pyramid level such that SP_l is composed of $2^0 \times 2^0 + 2^1 \times 2^1 + \ldots + 2^l \times 2^l$ histograms. The plot shows that SP_1 $(1 \times 1, 2 \times 2)$ gives the maximum score and increasing the pyramid level does not improve the classification performance.

	crop	split	crop-uni-split	crop-split
LSSVM	89.91 ± 1.69	**88.91 ± 1.37**	90.37 ± 1.21	90.32 ± 1.69
Iter. LSSVM	**90.02 ± 1.37**	88.86 ± 1.05	**90.68 ± 1.24**	**91.18 ± 1.38**

Table 3.3: Comparison of the LSSVM and Iterative LSSVM in terms of the multi-class classification accuracy for the proposed latent operations on the Graz-02 data set.

3.6.4 The Activities of Daily Living Dataset

The Activities data set [Messing et al., 2009] contains ten different types of complex actions like answering a phone, writing a phone number on a white-board and eating food with silverware. These activities are performed three times by five people with different heights, genders, and ethnicities. Videos are taken at high resolution (1280×720 pixels). A leave-one-out strategy is used for all subjects and the results are averaged as in [Messing et al., 2009].

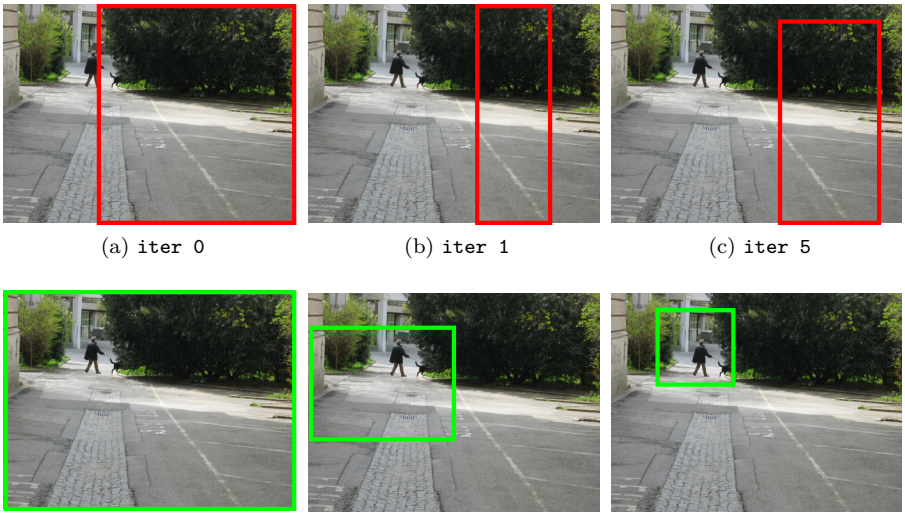
Table 3.1 shows the results for action classification on this data set. For this method, we obtain an improvement of 2.6% over SP method using the 'crop-uni-split' method. This is similar to the performance for classification of objects and indicates that the method is applicable to the classification of actions as well. The decrease in results for the 'crop' operation over the BoW method is mainly due to the fact that the HOF descriptors are not densely computed but only at the Harris3D interest points [Laptev and Lindeberg, 2003] and some temporal cells of the grid have very few descriptors. This problem may be overcome by densely computing spatio-temporal descriptors as done in the object classification experiments.

3.6.5 Results on Iterative Learning

We show results for the iterative learning of latent operations on the Graz-02, VOC-07 and Caltech-101 data sets. The grid size used for the Graz-02 data set is 12×12 and for the VOC-07 and Caltech-101 data sets it is 8×8. For the split operation we initially constrain the latent search space to the center of the images and expand it along the x and y directions by a fixed step size, a quarter of the number of rows and columns in the grid, *e.g.* 3×3 at $t = 1$, 6×6 at $t = 2$ on the 12×12 grid. For the crop, crop-uni-split, and crop-split operations, we initially fix the image window, *e.g.* $\{x_1, y_1, x_2, y_2\}$, as the full image. At each iteration, we relax the minimum width and height of the image window with a fixed step size, *i.e.* $0.5 \times$ grid size. Once the CCCP algorithm converges within the given latent space in an iteration, we expand the latent search space again at the start of the next. The algorithm terminates when the entire search space is covered.

Fig. 3.10 visualizes key iterations of the training for the cropping operation of a 'person' image for the LSSVM (see the first row in Fig. 3.10) and iterative LSSVM (see the second row in Fig. 3.10). In the iterative scheme, we initially fix the latent cropping box to be the full image size at the `iter 0` (Fig. 3.10.(d)). We then relax the constraint by allowing a smaller minimum size of the cropping box, i.e. half of the minimum size from the previous iteration. The ordinary LSSVM method does not have any such constraint on the latent parameter search. At the `iter 0`, the LSSVM misses the relatively small "person" in Fig. 3.10.(a) and converges to a wrong region and the error propagates to the next iterations (see Fig. 3.10.(b)-(c)). The LSSVM mis-classifies this training image as 'bike', while the iterative LSSVM gradually learns to localize the person better and correctly classifies the image (see Fig. 3.10.(f)).

Table 3.3 depicts the quantitative result of the iterative operations on the Graz-02 data set. The table indicates that the iterative method for LSSVM



(a) `iter 0` (b) `iter 1` (c) `iter 5`

(d) `iter 0` (e) `iter 1` (f) `iter 5`

Figure 3.10: Cropping operation on a 'person' labeled image for various iterations during training. The first and second rows show the result of the ordinary and iterative learning respectively. The first learning algorithm misses the 'person' in the first iteration and later converges to some part of background. The same local minimum is avoided in the second learning algorithm by restricting the possible image windows set to the full image in the first iteration and gradually relaxing the restriction.

generally improves the classification accuracy over the original formulation of the LSSVM. The crop-split benefits most from the iterative method, since it has more degrees of freedom and thus a stronger tendency to converge to a local minimum. The performance of iterative learning for the split operation worsens slightly.

Table 3.4 shows a quantitative comparison of original and iterative learning for the crop-split operation on the Graz-02, VOC-07 and Caltech-101 data sets. The iterative learning improves the classification performance for the Graz-02 and VOC-07 around 1%. However, we observe a slight drop in the classification accuracy on the Caltech-101. In the Caltech-101 data set objects are well centered, objects do not vary significantly in their sizes and the images are quite clean of clutter. Therefore, this data set does not benefit from the proposed learning method.

Fig. 3.11 plots the classification performance of the LSSVM and iter LSSVM

	Graz-02	VOC-07	Caltech101
LSSVM	90.32 ± 1.69	56.00	**75.04 ± 0.76**
iter. LSSVM	**91.18 ± 1.38**	**57.05**	74.93 ± 0.86

Table 3.4: Comparison of the LSSVM and iterative LSSVM on different data sets for the crop-split operation. Iterative LSSVM performs better in both the Graz-02 and VOC-07 data sets. The Caltech-101 data set does not benefit from the iterative method, since the images in this data set do not contain significant background clutter. Therefore, image windows are not less likely to converge to non-representative image parts in this data set.

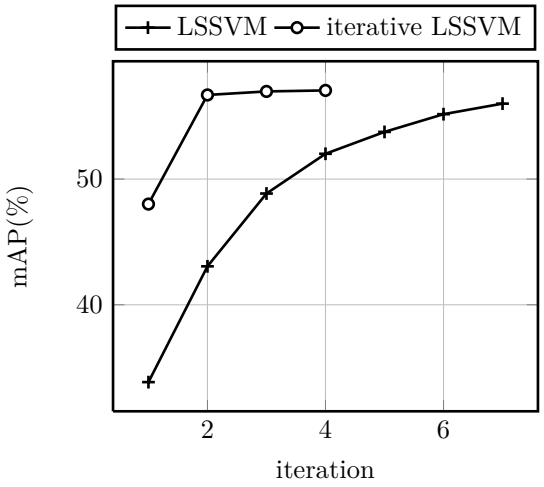


Figure 3.11: Classification results (mAP) with the AUC optimized crop-split on the VOC-07 over iterations for LSSVM and iter LSSVM algorithms. The minimum image windows size is limited to whole image size and half of it during the first and second iterations of the iterative learning respectively. The iterative learning starts with higher classification mAP on testing and takes fewer iterations to converge. The LSSVM and iter LSSVM converge to 56% and 57.05% mAP respectively.

for the crop-split operation on the VOC-07 data set over iterations. The CCCP algorithm, as described in section 3.3.3, at the beginning of each iteration, infers the latent variables. Having the latent parameters fixed, it minimizes the Eq. (3.9) during that iteration. We limit the minimum image window size for the iter LSSVM to whole and half image size during the first and second iterations respectively. We observe that the iter LSSVM already has 48% mAP at the end of the first iteration and converges fast to 57.05% mAP. However, the LSSVM takes 7 iterations to converge to 56% mAP.

3.6.6 AUC Optimization

In section 3.4, we described the use of an AUC based objective function to learn the classification with latent variables. This is useful in the case of binary classification, e.g. the VOC 2007 object classification task. For this task, we compare the proposed AUC loss against two baselines (ACC and N-ACC) in table 3.5. ACC denotes the 0-1 or accuracy loss. N-ACC is normalized accuracy loss for the number of positives and negatives, e.g. it penalizes false negatives more in the presence of more negative images. We evaluate their performances for the standard SP and latent crop-split operation. While the ACC loss performs worst in all three data sets, normalizing the loss (N-ACC) for positives and negatives with the number of positives and negatives respectively improves the mAP in both SP and crop-split. The AUC loss gives the best results and empirically shows that the AUC loss provides a better approximation of the AP on the VOC-07 data set than the ACC and N-ACC baselines.

Loss	SP (mAP)	crop-split (mAP)
ACC	53.46	54.37
N-ACC	54.18	56.98
AUC	**54.57**	**57.05**

Table 3.5: Comparison between the accuracy loss (ACC), normalized accuracy loss (N-ACC) and area under the roc curve loss (AUC) on the VOC-07 data set in mAP.

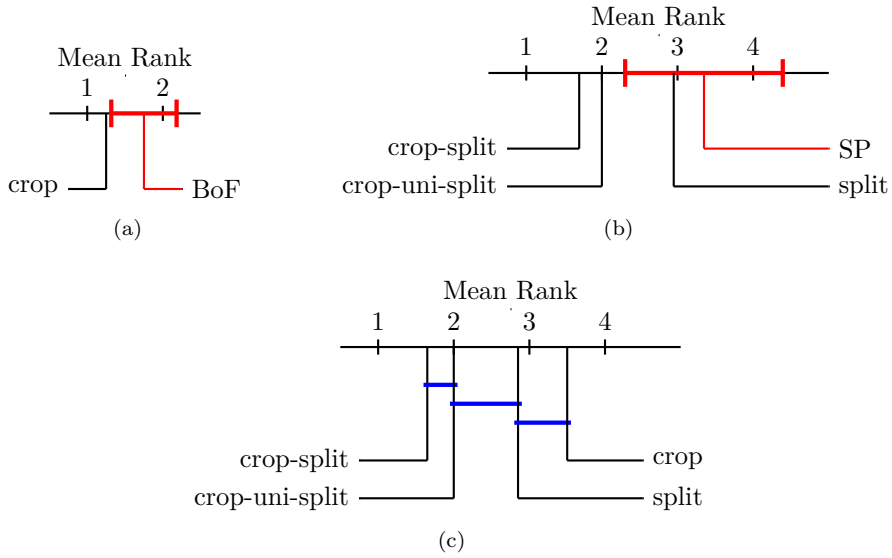


Figure 3.12: Significance analysis of the classification results on the VOC-07 data set. (a) shows a comparison of the BoW against the crop operation with the Bonferroni-Dunn test. The crop operation is outside the marked red interval and significantly different ($p < 0.05$) from the control classifier BoW. (b) shows comparison of the SPM against the split, crop-uni-split and crop-split operations with the Bonferroni-Dunn test. The crop-uni-split and crop-split operations are outside of the red marked range, therefore they are significantly better ($p < 0.05$) than SP. (c) shows comparison of all the proposed latent operations against each other with the Nemenyi test. Groups of classifiers that are not significantly different (at $p < 0.05$) are connected.

3.6.7 Statistical Significance of the Results

In this section, we further analyze whether the difference in performance between the proposed latent operations and the baselines is statistically significant. There is little work in the literature that handles the statistical evaluation of multiple classifiers on multiple data sets. We analyze our results by following two different evaluation tests as recommended by the authors of [Demšar, 2006].

In the first analysis, we group the methods in terms of their feature dimension to have a fair comparison. We explore whether the 'crop' operation produce statistically significant differences from the 'control' or baseline classifier BoW. We also compare the 'split', 'crop-uni-split' and 'crop-split' operations to the

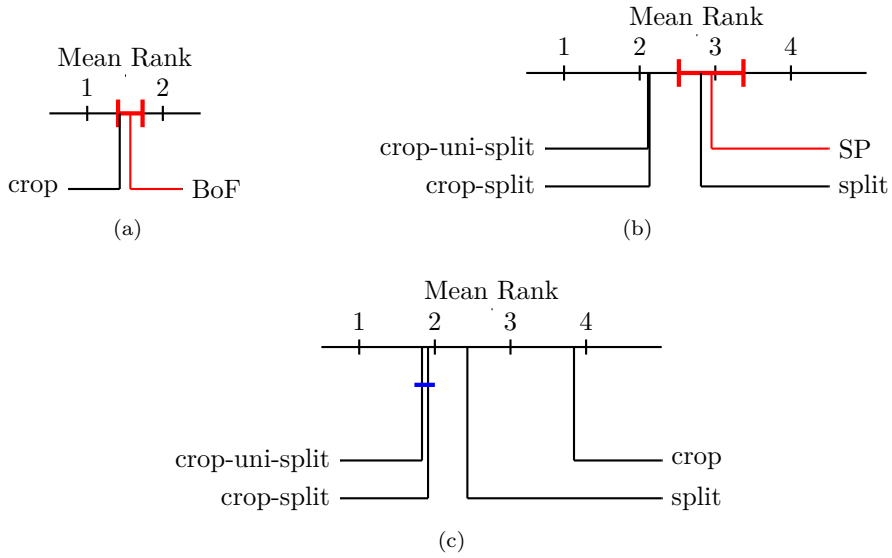


Figure 3.13: Significance analysis of the classification results on the Caltech-101 data set. (a) shows a comparison of the BoW to the crop operation with the Bonferroni-Dunn test. The crop operation is inside the red marked interval and is not significantly different ($p < 0.05$) from the control classifier BoW. (b) shows comparison of the SPM to the split, crop-uni-split and crop-split operations with the Bonferroni-Dunn test. The crop-uni-split and crop-split operations are outside of the red marked range, therefore they are significantly better ($p < 0.05$) than SP. (c) shows comparison of all the proposed latent operations to each other with the Nemenyi test. Groups of classifiers that are not significantly different (at $p < 0.05$) are connected.

SP. More specifically, we followed the two step approach of the Friedman test [Friedman, 1937] with the Bonferroni-Dunn *post-hoc* analysis [Dunn, 1961]. This approach ranks the classifiers in terms of their classification results (highest classification accuracy is ranked 1, 2nd one is ranked 2 *etc.*) and therefore it does not require any assumptions about the distribution of the accuracy or AP to be fulfilled. In our experiments, we consider each class as a separate test and rank each class among different methods. We test the hypothesis that it could be possible to improve on the control classifiers (BoW, SP) by using the latent operations. The null hypothesis which states that all the algorithms are equivalent is tested by the Friedman test. After the null hypothesis is rejected, we use the Bonferroni-Dunn test which gives a "critical difference" (CD) to measure the difference in the mean rank of the control and proposed classifiers.

Fig.3.12.(a)-(b) and Fig.3.13.(a)-(b) depict the results of the first analysis for the VOC-07 and Caltech-101 data sets respectively. This diagram is proposed by [Demšar, 2006]. The top line in the diagrams is the axis which indicates the mean ranks of methods in an ascending order from the lowest (best) to the highest (worst) rank. We mark the interval of CD to the left and right of the mean rank of the control algorithm (BoW and SP) in Fig.3.12.(a)-(b) and Fig.3.13.(a)-(b). The algorithms with the mean rank outside this range are significantly different from the control. Fig.3.12.(a)-(b) depict that the crop performs significantly better than the BoW; crop-uni-split and crop-split are significantly better than the SP on the VOC-07. Fig.3.13.(a)-(b) show that the crop is not significantly better than the BoW, the crop-uni-split and crop-split are still significantly better than the SP on the Caltech-101. While the VOC-07 data set images include cluttered background and small objects embedded in challenging backgrounds, the Caltech-101 images are cleaner. Therefore, only 'crop' operation cannot perform significantly better than BoW on the latter data set. The 'split' operation has enough degree of freedom to improve over the SP in neither of the data sets.

In the second analysis, we compare the performance of the latent operations to each other. We follow the same testing strategy as the authors of [Everingham et al., 2010] to analyze the significance of the results. We have used the Friedman test with a different post hoc test, known as Nemenyi test [Nemenyi, 1963]. Whereas the Bonferroni-Dunn test is more suitable to compare the proposed algorithms with a control classifier, the Nemenyi test is more powerful to compare all classifiers to each other. This test also computes a CD to check whether the difference in mean rank of two classifiers is bigger than this value. We show results of the second analysis for the VOC-07 and Caltech-101 data sets in Fig.3.12.(c) and Fig.3.13.(c) respectively. Fig.3.12.(c) shows that the 'crop' and 'split' are not significantly different from each other in terms of their classification performance, but their combination 'crop-split' is significantly better than both 'crop' and 'split'. This shows that these two operations are complementary to each other. In both Fig.3.12.(c) and Fig.3.13.(c) the 'crop-uni-split' and 'crop-split' are not significantly different from each other. This can be explained with the fact that the splitting operation cannot horizontally and vertically flip images. For example, in the case of a "horse" image and its horizontally flipped version, the splitting operation cannot align the parts of these two images in the same cells. We believe that an additional reflection parameter that horizontally swaps the features in the left cells with the right ones may improve the effectiveness of the splitting operation.

3.7 Discussions

We have developed a method for classifying objects and actions with latent window parameters. We have specifically shown that learning latent variables for flexible spatial operations like 'crop' and 'split' are useful for inferring the class label. We have adopted the latent SVM method to jointly learn the latent variables and the class label. The evaluation of our principled approach yielded consistently good results on several standard object and action classification data sets. We have further improved the latent SVM by iteratively growing the latent parameter space to avoid local optima. We also realized a better learning algorithm for unbalanced data by using an AUC based objective function. In the future, we are interested in extending the approach for weakly supervised object detection and improved large scale classification.

Chapter 4

Object Classification with Latent Regions

In the previous chapter, we have addressed the variation in object location and scale, by modeling them as latent variables and optimizing the learning algorithm over a set of possible spatial configurations. We have shown that this method represents objects in a more flexible way than the static spatial pyramid based methods and outperforms them. In this chapter, we propose an object classification method that better handles the complexity of real world images by jointly learning and localizing not only the object, but also a crude layout of its constituent parts as well as the background. This chapter adds three other contributions over the previous chapter and shows their impact on the final classification accuracy: (i) multiple exchangable local mixture representations for both background and foreground models, (ii) pairwise relationships between adjacent regions, (iii) novel initialization for these regions. Part of this chapter has been submitted to the Conference of Computer Vision and Pattern Recognition 2014 and currently under revision.

4.1 Introduction

In this chapter, we again classify objects (*e.g.* person or car) [Pinz, 2005] in the sense of PASCAL VOC [Everingham et al., b], *i.e.* indicating their presence in an image, but not their spatial localization (the latter is referred to as detection in VOC parlance). There is a broad palette of classification methods, most of them focusing on a better feature

representation [van de Sande et al., 2010, Ahonen et al., 2006] or better feature encoding [Wang et al., 2010, Zhou et al., 2010a] or a better mapping to a high-dimensional space [Duchenne et al., 2011]. However, very few methods have really improved the semantic representation of an object. In this chapter we propose an improved semantic representation of an object by considering its spatial location in the scene and the multi-modality of its appearance (*i.e* intra-class variation such that instances of the same object class can vary in their shape, color, *etc.*). Note that the problems of representing spatial location and multi-modality of appearance especially arise in the BoW representation that has been most successful for these classification problems.

At first sight, spatial location is accounted for by the BoW representation through discriminative visual words that are learned and should fire only on the object. The rest of the image is ideally associated with the non-discriminative visual words that should not contribute much to the final classification score. In the same way, multi-modal appearances should ideally be represented by the BoW representation using more visual words (by associating the discriminative words with different appearances in the same object class).

In practice, the first problem is that the BoW representation is quite sensitive to background visual noise such that visual words in the background that are not really correlated to the object, but due to a limited number of samples appear to be. The second problem is that the recognition model is prone to false positives in the presence of high intra-class variation. For instance, if we want to recognize only perfectly yellow and red birds assuming those colors as our visual features, a BoW representation would also recognize the ones with both yellow and red color and over-generalize the bird class because they are the composition of the two.

Incorporating the spatial information of an object and its multi-modal appearance model in a BoW representation is not straight-forward. Most of the state-of-the-art methods [Zhou et al., 2010a, Wang et al., 2010, Perronnin et al., 2010] still rely on a spatial pyramid (SP), which is a simple split of the image in a fixed grid of sub-regions, and for each they use a different BoW model. However, this is clearly sub-optimal because it represents the image as a *static and uniformly distributed* collection of appearance models.

We have addressed the localization problem in Chapter 3 by using the object location as a latent variable and optimizing the learning algorithm over a set of possible spatial configurations. This is a better representation than the SP, but it is still far from 'reality'. In real images the instances of a same object class can have multiple and quite different appearances that depend on the point of view, pose and specific object that are instantiated in the picture. Whereas, our previous method assumes that a single classification model can

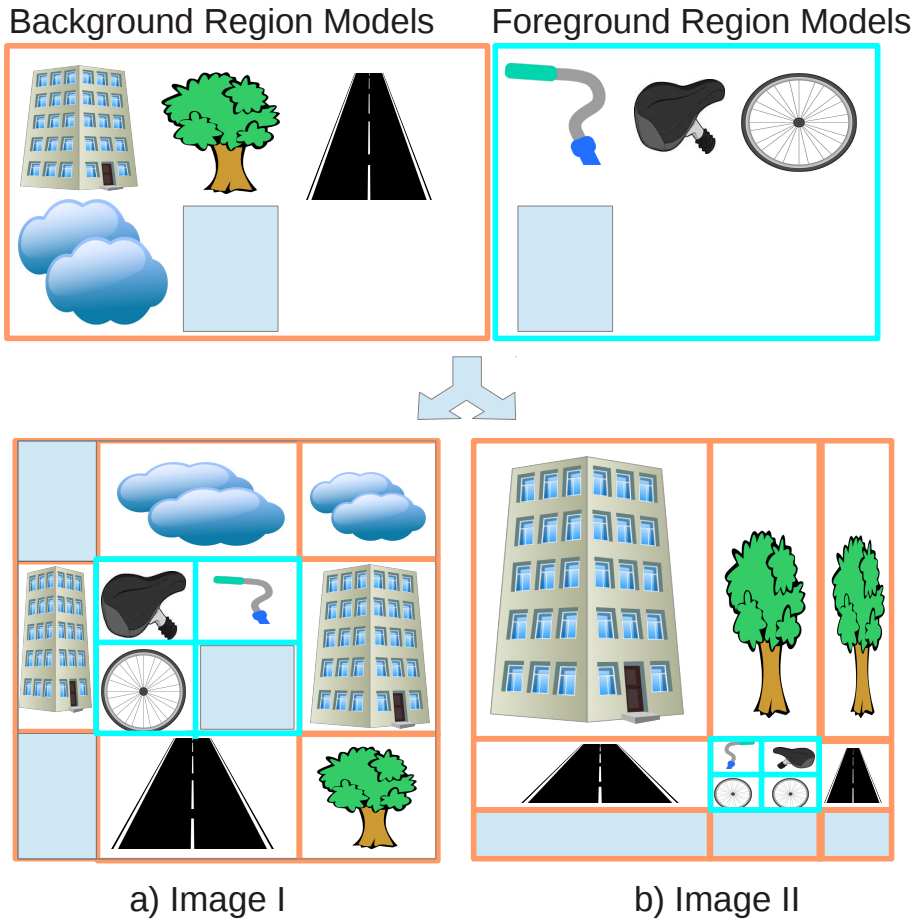


Figure 4.1: Overview of the proposed method. For our classification procedure we model an image as a composition of an object (foreground) enclosed in the cyan window box and the rest of the image (background). Both, foreground and background are represented by a pool of appearance models that are learned in a weakly supervised way. The empty blue rectangle models indicate occlusion models. Finally, for each appearance model and each location we learn unary and pairwise costs that reward likely configurations.

represent all the instances of a given object class. For instance, the crop-split operation uses a fixed model for each of the four splits and assumes that all instances of the same class have similar four parts. In this chapter, we relax this constraint, allow object instances in the same class to be represented by different part models and also learn the pairwise relation between these parts. Furthermore, the previous chapter does not consider any separation between the object and its background (everything that is not the object of interest) but it only focuses on inferring the most discriminative spatial configuration in an image. Although our previous method can still benefit from the background when it is discriminative (*e.g.* including the background 'road' in the crop window), it cannot explicitly model different background appearances and the pairwise relationship between foreground parts and different backgrounds.

In this sense, we propose an object classification method that improves the method in the previous chapter, better handles the complexity of real images by jointly learning and *localizing not only the object, but also its constituent parts as well as the background*. Similar to the 'Reconfigurable Bag of Words'(RBOW) [Parizi et al., 2012], where a scene is modeled as composition of multiple constituent parts, in this work we consider the object of interest as a composition of parts that can be placed together to better model its visual appearance. Furthermore, once the object (or foreground) is localized we also model the background as a composition of constituent parts. Finally, to enforce coherence in the models and better cope with appearance noise, we also learn *pairwise relationships between adjacent parts*. This permits us to avoid unlikely part configurations and therefore avoid false positives due to 'hallucinated' recognitions.

In spite of the seemingly high complexity of the model that needs to be learned from weak supervision, in this chapter we show that (i) we can formulate the problem as an instance of a latent structural SVM (LSSVM) [Yu and Joachims, 2009], (ii) we can, with careful initialization, learn meaningful models for the constituent parts of the object of interest as well as for the background (and their relationships) in a weakly supervised setting, where only the image label is given (no bounding box nor segmentation) and (iii) we empirically show through several experiments on the PASCAL VOC 2007 [Everingham et al., b] that the learned models improve the previous state-of-the-art and therefore may very well be a better representation of real images.

The remainder of the chapter is structured as follows. Section 4.2 relates our method to previous work. Section 4.3 formulates the inference of the latent variables and the learning procedure of the structural latent SVM model. Section 4.4 discusses different initialization strategies for the latent variables. Section 4.5 describes and discusses the results on the VOC 2007 dataset [Everingham et al., b] and finally Section 4.6 concludes the chapter.

4.2 Related Work

Many state-of-the art object classification methods focus on improving local features and encoding. Van de Sande *et al.* [van de Sande et al., 2010] study the invariance and distinctiveness of different color descriptors. Ahonen *et al.* [Ahonen et al., 2006] propose an enhanced descriptor by concatenating the local binary pattern (LBP) texture features. In parallel to the effort in local feature representation, many successful encoding methods have been proposed such as locality-constrained linear encoding (LLC) [Wang et al., 2010], the Improved Fisher encoding [Perronnin et al., 2010], super vector encoding [Zhou et al., 2010a], and kernel codebook encoding [van Gemert et al., 2008] to improve the histogram of quantized local features. A comprehensive evaluation of these encoding methods can be found in [Chatfield et al., 2011]. Most of these refinements are complementary to our work and they can be integrated with our method as well. As a matter of fact, we use the LLC encoding [Wang et al., 2010] in our work.

In the literature, numerous works [Lazebnik et al., 2006, Nguyen et al., 2009, Bilen et al., 2011, Pandey and Lazebnik, 2011, Russakovsky et al., 2012] have explored the idea of using spatial information for object classification. Spatial pyramids [Lazebnik et al., 2006] make use of the spatial information by dividing images into uniform regions and describe each region with a bag of words (BoW). We have shown in Chapter 3 that the choice of subregions in spatial pyramids can be further customized and optimized on image level to have better classification performance. Russakovsky *et al.* [Russakovsky et al., 2012] propose a complimentary object centric background model to boost the classification performance by using context information around the foreground. However, these approaches have limited power to deal with significant variability in appearances and views within the same object class.

Work closely related to ours is the recently proposed reconfigurable bag of words (RBOW) approach [Parizi et al., 2012] that models a scene as a composition of multiple constituent parts. Similarly, we also consider the object of interest and the background regions as a composition of parts that can be placed together to better model visual appearance. Whereas the RBOW method focused on scene classification, we tailor our method for object classification tasks. The fundamental difference between the two tasks is that in the latter one the foreground (object itself) usually has less variability in terms of appearance and includes more discriminative features than the background. Using background regions still helps to improve object classification performance but they need to be modeled separately from the foreground. We also validate this claim experimentally in Section 4.5. This issue does not arise in scene classification, since there is no clear distinction between regions as being foreground or

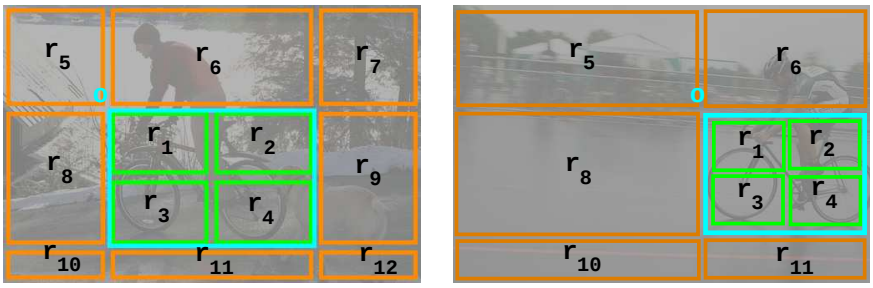
background. For our object classification task we propose an object centric approach. In this approach position of the foreground regions automatically defines the background regions around it. Moreover, we enforce coherence between the adjacent foreground-foreground, background-background and foreground-background regions by learning their pairwise relationship and show that we can better cope with appearance noise. Similar to the RBOW method, Yakhnenko *et al.* [Yakhnenko et al., 2011] represent images as a collection of regions. Yet they use only two region labels to represent foreground and background, and assume that all the parts of the objects can be represented by one foreground region model, while we solve the challenging task of capturing multiple foreground and background appearances.

Context has also been used in [Heitz and Koller, 2008, Choi et al., 2010, Rabinovich et al., 2007, Desai et al., 2009, Alexe et al., 2012] for object detection. In [Desai et al., 2009, Choi et al., 2010], the authors exploit the spatial interactions between object instances and in [Heitz and Koller, 2008, Rabinovich et al., 2007] the foreground-background relation is explored. Alexe *et al.* [Alexe et al., 2012] propose to use context information to reduce the number of candidate object windows. These methods require bounding box annotations of the training images however, while our approach asks only for image-level class labels.

4.3 Inference and Learning

We want to learn a binary classifier (class *vs.* non-class) that estimates for each image the location of the foreground as well as the constituent parts of the foreground and background. The training happens in a weakly supervised setting, as only the class-label of an image is given. Only a single object of the target class and therefore a single foreground window is supposed to be present in the image.

Fig. 4.2 illustrates the image representation that is used in this chapter. We represent each image $\boldsymbol{x}$ as a collection of foreground (drawn in green) and background regions (drawn in orange) (r_i). We uniformly split the region inside the given foreground window (o) (drawn in cyan) into four foreground regions $\{r_1, r_2, r_3, r_4\}$. Note that it corresponds to the "crop-uniform split" operation in the previous chapter. Although the "crop-split" gives better results, the difference in performance between the "crop-uniform split" and "crop-split" was found to be insignificant in the previous chapter, while the uniform one is computationally faster. The foreground window provides a natural split for the eight background regions $\{r_5, \ldots, r_{12}\}$ such as bottom-left, top-right,



(a) (b)

Figure 4.2: Diagram of possible spatial configurations. (a) depicts the full configuration with all the foreground $r_1, \ldots, r_4$ and the background regions $r_5, \ldots, r_{12}$. The foreground window o, drawn in cyan, separates the foreground and background regions. (b) depicts an example of a foreground window o being next to the image boundary. In this case, the regions r_7, r_9 and r_{12} are not visible. In the experiments, we set all the elements of the histograms on these regions to zero.

etc. of o. The spatial arrangement of all foreground and background regions can thus be parameterized through the specification of the single foreground window $o \in \mathcal{O}$, where $\mathcal{O}$ is the set of possible image windows in an image. Each region r_i is represented by a *region label* from a pool of learned appearances or models. We use l_i to specify the selected label for the region r_i. We have two two independent pools of region labels for foreground and background labels as illustrated in Fig. 4.1 and we learn a *region model* for each region label. In Fig. 4.1 the region labels of the toy classification problem are "building", "tree", "road", "cloud" and "occlusion" for the background and are "handle bar", "seat", "wheel" and "occlusion" for the foreground. Note that these labels are not known apriori and their annotations are not available but are automatically learned in our experiments.

We formulate the learning problem in two steps. The first step is inference, which finds the configuration of the foreground and background regions that maximizes a scoring function. The second is the learning step, which trains a model given a set of images and their class labels. We detail the two procedures in the following sections. Note that, as we are using a discriminative setting, learning will make use of the inference step.

4.3.1 Inference

The inference problem of our method is to find a prediction rule that infers a class label y for a previously unseen image $\boldsymbol{x}$ using a learned discriminatively trained model parameter $\boldsymbol{w}$:

$$y^* = \arg\max_y f_{\boldsymbol{w}}(\boldsymbol{x}, y), \tag{4.1}$$

where $f_{\boldsymbol{w}}(\boldsymbol{x}, y)$ is the discriminant function trained to give a high score if the image $\boldsymbol{x}$ belongs to class y. Moreover, we use an image window (o) to divide an image into foreground and background regions (r_i) (Fig. 4.2). We also assign a region label (l_i) to each region r_i. These parameters (location of o and region labels l_i) define the configuration of the image and are considered as latent variables, because the ground truth annotation of them is not available. As we use a linear model, the discriminant function $f_{\boldsymbol{w}}(\boldsymbol{x}, y)$ with the latent variables $\boldsymbol{h}$ is rewritten as:

$$f_{\boldsymbol{w}}(\boldsymbol{x}, y) = \max_{\boldsymbol{h}} \boldsymbol{w}^{\mathsf{T}} \boldsymbol{\psi}(\boldsymbol{x}, y, \boldsymbol{h}), \tag{4.2}$$

where $\boldsymbol{\psi}(\boldsymbol{x}, y, \boldsymbol{h})$ is the joint feature vector in the LSSVM formulation [Yu and Joachims, 2009] and $\boldsymbol{h} = (o, l_1, \ldots, l_M)$ contains the configuration of the latent variables. M denotes the total number of foreground and background regions in an image and is maximally 12 in our case. The window o and region labels l_i are obtained as the best configuration of foreground and background regions:

$$\begin{aligned} f_{\boldsymbol{w}}(\boldsymbol{x}, y) = & \max_o \sum_{i=1,\ldots,4} \max_{l_i} \left(\boldsymbol{A}^{\mathrm{fg}}_{r_i,l_i} + \boldsymbol{B}^{\mathrm{fg}\,\mathsf{T}}_{l_i} \boldsymbol{\phi}(\boldsymbol{x}, o, r_i) \right) \\ & + \sum_{i=5,\ldots,12} \max_{l_i} \left(\boldsymbol{A}^{\mathrm{bg}}_{r_i,l_i} + \boldsymbol{B}^{\mathrm{bg}\,\mathsf{T}}_{l_i} \boldsymbol{\phi}(\boldsymbol{x}, o, r_i) \right). \end{aligned} \tag{4.3}$$

where $\boldsymbol{A}^{\mathrm{fg}}_{r_i,l_i}$ and $\boldsymbol{A}^{\mathrm{bg}}_{r_i,l_i}$ are biases that tell us how compatible the region label l_i is with the region r_i for foreground and background respectively. In the same way, $\boldsymbol{B}^{\mathrm{fg}}_{l_i}$ and $\boldsymbol{B}^{\mathrm{bg}}_{l_i}$ are the appearance parameters associated with the feature map $\boldsymbol{\phi}(\boldsymbol{x}, o, r_i)$ for the region label l_i for foreground and background. As the best label can be selected for each region independently, the optimization is fast and it can be done for each window location o.

Now, we can introduce pairwise costs $\boldsymbol{C}_{r_i,r_j,l_i,l_j}$ that define the compatibility between the chosen labels l_i, l_j of adjacent regions r_i, r_j with $(i, j) \in \varepsilon$, the set

of connected region pairs. The new discriminant function is:

$$f_{\boldsymbol{w}}(\boldsymbol{x}, y) = \max_{o,l} \Big(\sum_{i=1,\ldots,4} \boldsymbol{A}^{\text{fg}}_{r_i,l_i} + \boldsymbol{B}^{\text{fg}\mathsf{T}}_{l_i} \boldsymbol{\phi}(\boldsymbol{x}, o, r_i) + \sum_{i=5,\ldots,12} \boldsymbol{A}^{\text{bg}}_{r_i,l_i} + \boldsymbol{B}^{\text{bg}\mathsf{T}}_{l_i} \boldsymbol{\phi}(\boldsymbol{x}, o, r_i) + \sum_{(i,j)\in\varepsilon} \boldsymbol{C}_{r_i,r_j,l_i,l_j} \Big). \tag{4.4}$$

In this case, for each possible window o the selection of a region label for a certain region depends also on its neighbors. Thus, whereas in Eq.(4.3) we could select the label for each region independently, now the scoring function needs a global optimization over $l = \{l_1, l_2, \ldots, l_{12}\}$. To do that, we use a conditional random field (CRF) optimization for each window location o based on re-weighted tree belief propagation [Kolmogorov, 2006]. As the number of regions and labels is relatively small this optimization is still quite fast.

The discriminant function (4.4) allows us to define the LSSVM parameter vector $\boldsymbol{w}$ and the joint feature map $\boldsymbol{\psi}(\boldsymbol{x}, y, h)$ in Eq.(4.2). The parameter vector is now a concatenation of the bias parameters $\boldsymbol{A}^{\text{fg}}_{r_i,l_i}$, $\boldsymbol{A}^{\text{bg}}_{r_i,l_i}$, the appearance parameters $\boldsymbol{B}^{\text{fg}}_{l_i}$, $\boldsymbol{B}^{\text{bg}}_{l_i}$ and the pairwise parameters $\boldsymbol{C}_{r_i,r_j,l_i,l_j}$. Having the parameter vector $\boldsymbol{w}$, we design the joint feature vector $\boldsymbol{\psi}(\boldsymbol{x}, \boldsymbol{h})$ for a given class y and configuration $\boldsymbol{h}$ as follows: When the class y is present ($y = 1$) in the image, $\boldsymbol{\phi}(\boldsymbol{x}, o, r_i)$ is positioned at the corresponding location of the label l_i for each region r_i. When the class is not present $y = -1$ in the image x, we set all elements of the feature map vector $\boldsymbol{\psi}(\boldsymbol{x}, y, h)$ zero. Note that this does not mean that the negative images are not used during the training. As shown in the next section, our learning procedure enforces the highest response from a negative image to be smaller than 0 and the one from a positive image bigger than 0 with a margin.

4.3.2 Learning

Given a set of training samples $X = \{\boldsymbol{x_1}, \ldots, \boldsymbol{x_n}\}$ and their labels $Y = \{y_1, \ldots, y_n\}$, where each $y_i \in \{-1, 1\}$ $(i = 1, \ldots, n)$, we learn a linear SVM model $\boldsymbol{w}$ to predict the class label of an unseen example. We also use the latent parameters $H = \{\boldsymbol{h}_1, \ldots, \boldsymbol{h}_n\}$ to select the image windows o that specify the spatial configuration, and labels l that explain the resulting foreground and background regions best. The region labels l correspond to those introduced in Section 4.3.1. To jointly learn the SVM model and latent parameters, we follow

the latent SVM formulation of [Yu and Joachims, 2009]:

$$\min_{\boldsymbol{w}} \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{n}\Big[\ \max_{\hat{y}_i,\hat{\boldsymbol{h}}_i}\left[\boldsymbol{w}^{\mathsf{T}}\boldsymbol{\psi}(\boldsymbol{x_i},\hat{y}_i,\hat{\boldsymbol{h}}_i)+\Delta(y_i,\hat{y}_i)\right] - \max_{\hat{\boldsymbol{h}}_i}\left[\boldsymbol{w}^{\mathsf{T}}\boldsymbol{\psi}(\boldsymbol{x_i},y_i,\hat{\boldsymbol{h}}_i)\right]\Big] \tag{4.5}$$

where C is the penalty parameter and $\Delta(y_i,\hat{y}_i)$ is the $0-1$ loss function $\Delta(y_i,\hat{y}_i)=1$ if $\hat{y}_i \neq y_i$, and else 0. We refer to Section 3.3 for more details.

4.4 Initialization of Latent Parameters

The success of our method relies on learning discriminative appearance models that can represent a wide range of variability in the appearance and spatial configuration of foreground and background regions. In other words, each foreground and background model should be distinctive and at the same time general enough to appear in a certain number of images. We use the LSSVM framework to train those models and relations in a discriminative way. However, the BoW representation with a linear SVM model has many degrees of freedom and is thus usually able to learn a classifier with small training error by over-fitting on the training set. Here, the initialization of parameters for the optimization plays an important role to learn discriminative background and foreground models. One can set the parameter vector $\boldsymbol{w}$ or the latent parameters $\boldsymbol{h}_i$ for each sample i to initialize the optimization. In our experiments, we prefer to initialize the latent variables, since setting $\boldsymbol{w}$ with an arbitrary norm (*i.e.* $\|\boldsymbol{w}\|$) may introduce stability problems or biases.

In practice we find that our optimization algorithm is more sensitive to different initialization strategies for background regions than for foreground ones. This can be explained by the fact that background regions carry more variation in appearance than foreground. Therefore, we focus on the initialization of background regions. For the foreground regions, we use a *fixed initialization* strategy by assigning a particular region label to each foreground region (*i.e.* $r_i \leftarrow l_i$ for $i \in \{1,2,3,4\}$). Moreover, we use the localization result of the crop-uni-split model (or LOC), which is introduced in Chapter 3, to set our initial window (o) for each image. The patches inside and outside of those windows are considered as foreground and background patches respectively.

A naive initialization strategy is to assign a particular region label to each background region depending on its location, as done for the foreground regions.

However, in our experiments, we found that using this fixed initialization prevents changes in the latent parameters during the following optimization iterations. One can also cluster regions that are collected from positive images. This method guarantees that regions are grouped in terms of similarity of their appearance. However, it does not ensure that those groups do not exist in negative images and are discriminative. Alternatively, one can train an exemplar classifier, such as exemplar SVM [Malisiewicz et al., 2011], for each positive background region against a collection of negative regions. Then, we can test the trained exemplar classifiers on the validation set and choose the most discriminating ones and use them to label background regions. However, training thousands of linear SVMs is computationally expensive. Therefore, we propose to use a simpler linear classification method, linear discriminant analysis (LDA) [Hastie et al., 2001]. LDA has a shorter training time and comparable performance, as shown in [Hariharan et al., 2012].

In order to initialize the latent parameters in the training set, we follow the procedure:

1. Training an LDA classifier requires the computation of the covariance matrix $\boldsymbol{S_W}$ and mean $\boldsymbol{m}_-$ of negative background regions (see Eq. 2.16). To compute them, we use a 8×8 grid and use all the possible windows in negative images.

2. We run the crop-uniform split method on the training set to initialize foreground windows (o). Setting window o in positives images automatically defines the locations of foreground and background regions.

3. We encode each background region (r_j) in each positive image ($\boldsymbol{x}_i$) with the LLC encoding (see Section 2.2) and denote it as ($\boldsymbol{p}_{ij}$) and learn a LDA classifier $\boldsymbol{\theta}_{ij}$ (as explained in Section 2.3.1):

$$\boldsymbol{\theta}_{ij} \propto {\boldsymbol{S_W}}^{-1}(\boldsymbol{p}_{ij} - \boldsymbol{m}_-). \tag{4.6}$$

4. In order to prevent very similar background regions to be chosen, we compute the cosine of the angle between the learned LDA classifier pairs and remove similar ones with a threshold of 0.4.

5. To pick the best N LDA classifiers, we use an SVM with a l_1 regularization which encourages sparsity among the learned weights. Briefly, we describe each background region with a K-dimensional vector that contains scores of the K learned LDA classifiers. We concatenate these vectors and obtain a $8 \times K$ descriptor for each image. We train an SVM on these features that chooses the most discriminative and independent LDA classifiers.

6. We take absolute value of the learned linear SVM model and use the corresponding LDA classifiers with the highest weights to initialize the background regions of positive images in the training set.

4.5 Experiments

Dataset. We evaluate our system on the challenging PASCAL VOC 2007 [Everingham et al., b]. It contains 9,963 images which are split into training, validation and testing sets. The images are labeled with twenty classes, also allowing multiple classes to be present in the same image. We learn a one-vs.-rest classifier for each class and report the average precision for each class as well as the mean AP (mAP) which is the mean of AP values from each of the classifiers.

Implementation Details. We extract dense SIFT features [Lowe, 1999a] by using the `vl_phow` function from the VLFeat toolbox [Vedaldi and Fulkerson, 2008]. We apply K-means to the randomly sampled 200,000 descriptors from the training images to form the visual codebook. The computed visual words are then used to build up the descriptors using the LLC coding and max pooling [Wang et al., 2010]. For the LLC coding, we set the number of nearest neighbors and the regularization parameter to 5 and 10^{-4} respectively. The codebook size is 8192. The encoded feature vectors for foreground and background are normalized to have l_2 norm 1 and 0.1 respectively. This normalization strategy forces the SVM model parameters to be regularized more strictly for the ones that correspond to background. This gives more importance to the foreground representation and it has a positive impact on the final classifier accuracy.

Spatial Pyramid. Our baseline is a BoW implementation with a $1 \times 1, 2 \times 2$ spatial pyramid and LLC coding. We have shown that using more levels of pyramid does not improve the performance in the previous chapter (See Fig. 3.9). BoW with a spatial pyramid is the basic configuration used by most of the state-of-the-art methods. In Table 4.1 this method is denoted by `(1)` and it obtains a mAP of 54.7%. Notice that the score is obtained by using a single feature and sparse coding. Using multiple features (*e.g.* LBP, HOG) and a better encoding (*e.g.* fisher kernels) should improve the baseline as well as any row of the table because our contributions are orthogonal to those.

Localization. This configuration corresponds to the "crop-uniform split" in Chapter 3. For each image, a latent window is used to localize the object of interest. We use a coarse 8×8 grid to spatially quantize the images as in the previous chapter and this produces 1296 unique configuration for the foreground window o. For the latent SVM we build our models on top of the publicly available code of Yu and Joachims [Yu and Joachims, 2009]. The

regularization parameter C is set to 10^6 for all experiments. As shown in Table 4.1 configuration `(2)`, the latent localization of the object of interest is a fruitful strategy, and improves over the baseline SP in most of the categories. This method increases the mAP over the SP by around 2 points.

Multiple Appearances. In Table 4.1, configurations `(3)` and `(4)` correspond to our multiple models for foreground and background respectively. With the introduction of multiple models both foreground and background result in a similar improvement of around 1 point each. For background we use the initialization based on LDA as explained in Section 4.4. In our preliminary experiments we have noticed that the best performance is obtained by using the same number of models as the number regions. Thus we learn 4 foreground models and 8 background models.

Pairwise Compatibility. On top of the previous configuration we add the pairwise costs defined in Section 4.3 and denote this setting as `(6)` in Table 4.1. These additional costs enforce coherency between adjacent regions and therefore help to produce a more consistent representation of the scene. The overall benefit of the pairwise costs is 1.1 percent and certain classes show a substantial improvement (*e.g.* bicycle +2.0, cat +3.0, cow +3.1, dog +4.2, motorbike +2.0, sheep +1.8).

We also evaluate the effect of modeling foreground and background separately and of localizing the foreground. A similar setting has been used in [Parizi et al., 2012] for a scene classification task where there is no distinction between foreground and background. In practice, for this configuration (denoted as `(5)` in Table 4.1) we use a fixed window o at the center of the images to divide the image into equal 9 regions. We let each of these regions (r_i) to choose best the region label (l_i) considering also the pairwise constraints. In this case the mAP is 55.8% and the increment with respect to the SP is only 1.4, whereas with localization the increment is 5.2 points. This indicates that the localization of the object of interest also helps to produce better appearance models and it is therefore crucial for a good object classification system. We also visualize the estimated latent variables for the full configuration in Fig. 4.4 and show that we can obtain *semantically* meaningful results.

Latent Initialization. We also compare the initialization strategy based on LDA, which is explained in Section 4.4, with the fixed initialization, where each region r_i depending on its index value (i) is assigned to a label l_i. In case of fixed initialization the number of models should be equal to the number of regions. Therefore, in order to provide a fair comparison, we also use 4 foreground and 8 background models (with the pairwise connections in both settings) for the LDA. We obtain 58.1% mAP for the fixed initialization, while the LDA based strategy achieves 59.3% (Table 4.1, configuration `(6)`) with a net improvement

of 1.2.

To further analyze our initialization strategy and latent learning, we set an additional baseline experiment by initializing the foreground windows with the ground truth bounding boxes. We adapt the annotated boxes to an 8×8 grid by quantizing their coordinates. In the case of multiple instances of the same object in an image, we pick the one with the bigger area. Initializing the foreground window with the ground truth ones achieves 59.5% average precision and improves 0.2% over the weakly supervised case. This shows that our classifiers achieve a comparably good performance and still learn well with latent localization.

Comparison to Similar Methods. We also compare our best configuration, `(6)` to the reported results from the work of [Chatfield et al., 2011] (LLC(25k)) and [Russakovsky et al., 2012] (OCP) that also use a single type of local feature, SIFT and DHOG respectively. The first column (LLC(25k)) of Table 4.2 shows the result obtained by using a SP, LLC encoding with 25,000 visual words and approximated chi square kernel and a $1 \times 1, 2 \times 2, 3 \times 1$ spatial pyramid. Even though this setting uses a bigger codebook and a non-linear kernel, our model is still better. The second column (OCP) depicts the results of [Russakovsky et al., 2012]. This method also localizes the object of interest and represents the background. However, it is still 2.1 points below our best configuration. This shows that using multiple models and pairwise costs really helps to boost our classification.

Weakly Supervised Detection. An interesting aspect of our method is that it outputs a coarse location of objects as well as a class label. The classifiers were optimized to improve classification by using a rough localization (*i.e.* a grid with 8×8 cells with a minimum 2×2 foreground size) that ensures good classification results while being computationally efficient. Of course, the use of such a coarse grid implies that the detection results are not very accurate. We have evaluated the detection accuracy of the baseline configuration (localization `(2)`) and our best version (localization+mixtures+CRF `(6)`) using the protocol (*i.e.* the percentage of training images in which an object instance is correctly localized by the highest-scoring detection in terms of the PASCAL criterion (50% inter-section over union) introduced in [Deselaers et al., 2010] for the weakly supervised case. While the baseline `(2)` gives a score of 20.9%, our improved configuration `(6)` obtains 24.7% over all classes of the VOC-07. Note that we don't use any class/aspect information. This shows that using multiple mixtures for the object parts also helps to arrive at a better localization. Further improvements of the method can probably be obtained by using a finer grid or a selection of windows [van de Sande et al., 2010].

Image Retrieval with Semantic Similarity. In addition to inferring a

	(1)	(2)	(3)	(4)	(5)	(6)
LOC		x	x	x		x
MFG			x	x	x	x
MBG				x	x	x
CRF					x	x
mean	54.7	56.5	57.2	58.2	55.8	**59.3**
aeroplane	70.0	70.1	72.5	**76.1**	75.0	74.2
bicycle	59.6	64.0	65.3	63.5	62.6	**65.5**
bird	45.4	45.9	46.2	49.1	42.6	**50.0**
boat	64.4	66.9	66.9	**67.7**	66.6	67.2
bottle	24.8	24.6	25.3	**27.7**	24.3	26.9
bus	60.4	64.0	63.7	63.6	59.4	**65.2**
car	75.3	77.0	76.9	79.0	75.7	**80.2**
cat	57.5	59.9	58.2	60.4	58.4	**63.4**
chair	53.5	55.1	**55.6**	54.7	50.2	53.9
cow	42.9	45.4	46.0	46.4	44.2	**49.5**
diningtable	46.9	46.9	47.6	51.3	48.8	**52.4**
dog	41.2	41.7	42.0	43.4	44.9	**47.6**
horse	71.4	74.7	74.6	76.6	75.3	**77.4**
motorbike	62.7	66.2	67.0	66.5	64.8	**68.5**
person	82.4	82.5	82.6	83.3	81.5	**83.7**
pottedplant	22.5	22.7	26.7	26.9	24.7	**27.2**
sheep	43.5	44.2	44.7	44.8	43.0	**46.6**
sofa	49.6	53.8	55.1	54.5	51.5	**55.6**
train	70.9	72.6	73.2	75.2	72.2	**75.6**
tv	50.0	53.1	53.8	53.8	49.5	**54.4**

Table 4.1: The classification results in terms of AP on PASCAL VOC 2007 for different configurations of our method. LOC, MFG, MBG and CRF denote localization, mixture of foreground models, mixture of background models and conditional random fields respectively.

	LLC (25k)	OCP	Our Method (6)
mean	57.7	57.2	**59.3**
aeroplane	72.4	**74.2**	**74.2**
bicycle	62.2	63.1	**65.5**
bird	47.3	45.1	**50.0**
boat	**68.9**	65.9	67.2
bottle	25.8	**29.5**	26.9
bus	64.0	64.7	**65.2**
car	77.3	79.2	**80.2**
cat	59.8	61.4	**63.4**
chair	54.3	51.0	**53.9**
cow	46.0	45.0	**49.5**
diningtable	51.1	**54.8**	52.4
dog	43.2	45.4	**47.6**
horse	76.7	76.3	**77.4**
motorbike	67.1	67.1	**68.5**
person	83.5	**84.4**	83.7
pottedplant	**27.7**	21.8	27.2
sheep	44.9	44.3	**46.6**
sofa	52.8	48.8	**55.6**
train	**76.0**	70.7	75.6
tv	52.5	51.7	**54.4**

Table 4.2: Comparison to the related published results on the PASCAL VOC 2007, the LLC (25k) [Chatfield et al., 2011] and OCP [Russakovsky et al., 2012]. Our method outperforms other related methods in most of the classes and also in mean average precision.

class label, our method also divides the image into regions and assigns a label to each image region. We claim that these labels provide a coarse semantic level representation of the image. In this part we test our method on retrieval of similar content for a given query image on VOC07. We show preliminary qualitative results in Fig. 4.3. We run our classifiers trained for the configuration (6) on the test images of VOC07 dataset and use the inferred foreground and background region labels to describe each image. We randomly pick a query image and compute the Hamming distance between the labels of the query and the test images and rank their distance to the query. As a baseline we evaluate a SP representation only on the images that are from class of the query. We use the cosine similarity between the two normalized histograms to rank related images. Fig. 4.3 shows that SP can retrieve similar images only when the spatial

Figure 4.3: Sample retrieval images obtained by using the image representation of configuration `(6)` (Ours) and spatial pyramid (SP). Our method makes use of the latent labels assigned at inference to retrieve semantically similar images.

layout of the entire test image is close to the query. In contrast our method can describe images with a more "semantic" representation which therefore induces a better retrieval.

Computational Cost. The running time of the LSVM experiments is dominated by inference of the best configuration for each image (*e.g.* scanning all possible windows and all possible background models). The training of each class-specific classifier for the full configuration (configuration `(6)`) in the VOC 2007 experiments took 4 hours on a 12 CPU machine. The typical inference time for an image is 5 seconds.

4.6 Conclusion

In this chapter we have introduced a new semantic representation of an image based on latent variables that can improve the object classification accuracy without requiring any additional annotation. With an incremental evaluation of each characteristic of our model on the challenging Pascal VOC 2007, we have shown that localizing the object of interest in the image is important as

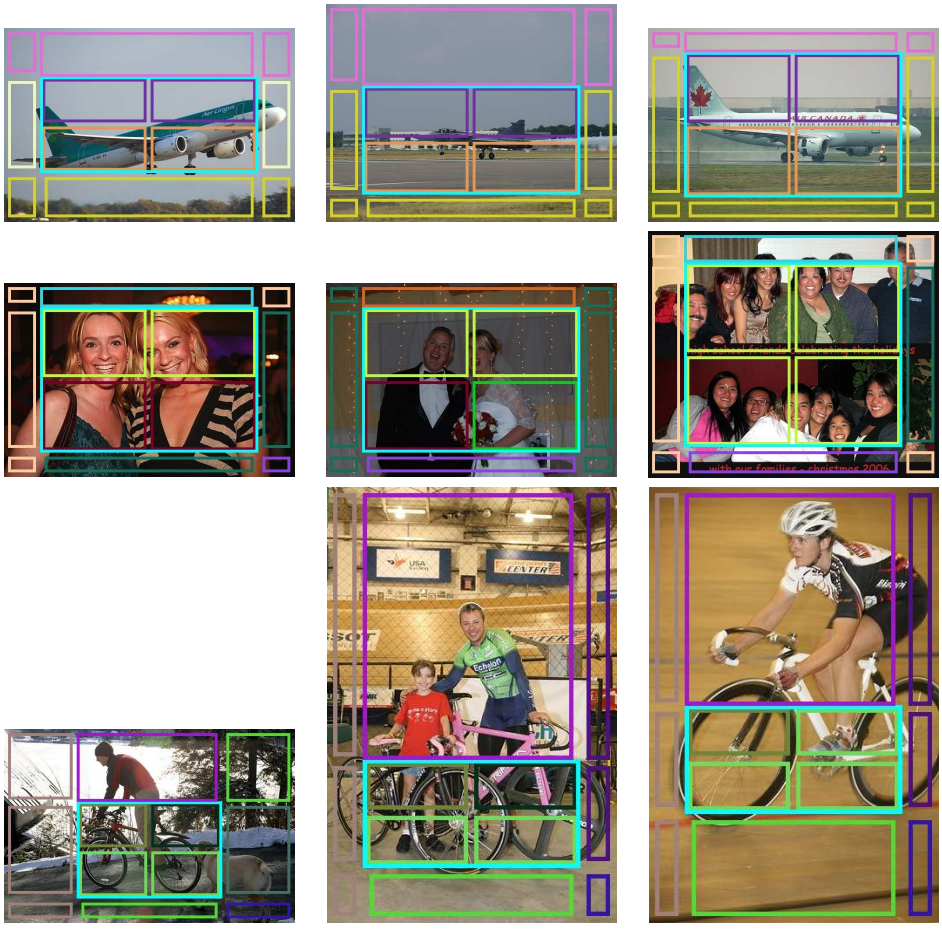


Figure 4.4: Examples of estimation of the latent variables on different classes of Pascal VOC 2007. The cyan bounding box represents the localized object of interest. The other bounding boxes represent the different regions of the image for foreground (inside the object of interest) and background (outside the object of interest). For a certain class, the color of the bounding box represents the inferred appearance model. Thus, same color means same appearance model. The examples in the first row show that 'sky' and 'ground' background regions are consistently labeled with a particular model. In the second row, faces and upper body of people are assigned to different foreground models. In the last row, as 'bicycle' is the class of interest, people in the images are assigned to a background region label (l_i).

well as properly representing the multi-modal appearance of the object and its background. Furthermore, additional accuracy can be obtained by learning and enforcing pairwise costs between the neighboring regions. Altogether our model is able to achieve a gain of 4.6 points over the standard spatial pyramid, without any additional images, low-level feature or annotations.

Chapter 5

Classification with Global, Local and Shared Features

In the previous two chapters, we have focused on developing classification methods by modeling intra-class (or within-class) variations with latent parameters. In this chapter, we again consider the classification problem of deciding whether one of a number of pre-specified object classes, *e.g.* bicycle, motorbike, or person, is present in an image. However, we focus on learning inter-class differences between visually similar object categories. In particular, we show that additional learning of pairwise relations between classes improves such classification: when having to tell whether or not a specific *target class* is present, sharing knowledge about other, *auxiliary classes* supports this decision. Parts of this chapter are published in the DAGM 2012 [Bilen et al., 2012] and in the Fine-Grained Visual Classification Workshop of the Conference of Computer Vision Pattern Recognition 2013 [Bilen et al., 2013a].

Our method stands in contrast to standard classification approaches that only exploit global as in [Lazebnik et al., 2006] or local information as in [Nguyen et al., 2009] and our Chapter 3 about the target class. In particular, we propose a framework that combines target class-specific global and local information with information learnt for pairs of the target class and each of a number of auxiliary classes. The advantage of adding such pairwise information is that it aids generalization. The common context for a class pair helps it being discriminated against other classes. For instance, similar classes like 'bicycle' and 'motorbike' share features that enable to discriminate both from other classes. The target class-specific parts of the models for 'bicycle' and 'motorbike' rather focus on specific nuances that are needed to discriminate between the

pair. Even if in this chapter we often formulate the approach in terms of object classification, the very same framework will be demonstrated for flower and action classification in the same fashion.

In summary, our target class model combines information about:

1. global image appearance, using a spatial pyramid over the image, thereby providing context information;
2. local appearance, based on a target class-specific window, loosely corresponding to a bounding box;
3. shared appearances, based on a series of windows, each jointly defined for the target class and one of the auxiliary classes with which there are visual commonalities.

We show that all components of this combined representation can be learnt jointly, with as only supervision the class label for the training images (i.e. which target class appears in the images without any information on its location).

We have evaluated our approach for object, flower and action classification tasks using standard benchmarks, after such joint learning of the global, local, and shared components. We have experimentally evaluated each of these components individually and jointly for solving these various problems. The results show that adding the shared component is beneficial in all cases.

5.1 Related Work

In Chapter 3, we have considered the use of local representations – in the form of a window – as a latent variable that is learnt jointly with other classification model parameters. In Chapter 4, we have shown that using a pool of foreground and background appearances effectively represents the variance within the same category and thus improves the classification performance. In this chapter, we advocate the localization of shared appearances between related class pairs in addition to the class specific localization as done in Chapters 3 and 4. In contrast to the previous two chapters which focus on learning the intra-class variations, this chapter aims to better learn the similarities and differences between different classes (or inter-class variation) in addition to the intra-class variations and finally to improve classification. Here, we do not consider the different foreground and background appearances as in the previous chapter for the sake of simplicity. However, the use of multiple appearances can also be

added to this method, since the methods in the current and previous chapters are complimentary.

The central contribution of this chapter is the use of appearance properties that are shared by pairs of classes. The issue of sharing has so far been explored more for object detection [Salakhutdinov et al., 2011, Fergus et al., 2010, Opelt et al., 2006b] than for object classification, where this is more intricate to implement. In the case of classification we cannot assume that training images come with the locations of objects. Sharing for classification is therefore more challenging. It has been considered in the large margin framework based on a pre-specified hierarchy [Dekel et al., 2004]. We do not rely on such restriction. Sharing has also been implemented by relying on auxiliary information such as text [Sadeghi and Farhadi, 2011] or by constructing hierarchies from WordNet [Marszałek and Schmid, 2007]. An interesting recent approach for sharing used other detector information as cues [Li-Jia Li and Fei-Fei, 2010]. As a matter of fact, there has also been work that uses the output of classifiers to learn sharing between classes [Torresani et al., 2010]. In contrast to that approach, we learn the sharing together with the classier itself. Moreover, we learn not only to share at the level of a class pair, but also adapt the sharing window to the individual instance of the target class (*i.e.* the window is not at a fixed relative position for the entire class).

The use of multiple visual contexts has been considered for recognizing scenes [Quattoni and Torralba, 2009]. However, that work relies on using different features to capture those different appearance contexts. Recent work by Pandey and Lazebnik [Pandey and Lazebnik, 2011], follows this line of thought and combines global GIST features with local HOG features. Our work is complementary to these ideas. We focus on obtaining different contexts from a single feature type. Yet, our framework is not restricted to a single feature.

There is also substantial work developed on multi-label learning that aims to capture the dependency among classes in which an object can be classified into more than one class. Ghamrawi and McCallum [Ghamrawi and McCallum, 2005] model the label co-occurances by defining a conditional random field over all the pairwise class combinations. Ueda and Saito [Ueda and Saito, 2002] propose a probabilistic generative method that parameterizes the dependency among classes with mixture models. In [Liu et al., 2006], a non-negative matrix factorization is used to learn the optimal class groupings. Ji *et al.* [Ji et al., 2008] learn a shared subspace among multiple labels to capture the class correlations. These methods are orthogonal to our approach and can also be incorporated to our work to learn better class groupings and better shared feature subspaces.

5.2 Model Definition

To build our classifiers, we again make use of the structural SVM formulation with latent parameters [Yu and Joachims, 2009]. In our model, input $\boldsymbol{x} \in \mathcal{X}$, output $y \in \mathcal{Y} = \{c_1, \ldots, c_k\}$ and latent parameters $\boldsymbol{h} \in \mathcal{H}$ correspond to the image, its label, and a set of bounding boxes, respectively. We use discriminant functions of the form $f_{\boldsymbol{w}} : \mathcal{X} \times \mathcal{Y} \times \mathcal{H} \to \mathcal{R}$ which scores triplets of $(\boldsymbol{x}, y, \boldsymbol{h})$ for a learnt vector $\boldsymbol{w}$ of the structural SVM model as

$$f_{\boldsymbol{w}}(\boldsymbol{x}, y, \boldsymbol{h}) = \boldsymbol{w}^{\mathsf{T}} \boldsymbol{\psi}(\boldsymbol{x}, y, \boldsymbol{h}) \tag{5.1}$$

where $\boldsymbol{\psi}(\boldsymbol{x}, y, \boldsymbol{h})$ is a joint feature vector that describes the relation among $\boldsymbol{x}$, y and $\boldsymbol{h}$. In our model, $\boldsymbol{\psi}(\boldsymbol{x}, y, \boldsymbol{h})$ concatenates histograms which are obtained from multiple rectangular windows with the bag of words (BoW) representation [Wang et al., 2010]. We use different windows to encode the 3 information channels, *i.e.* global, local, and shared. We can write our feature vector for class y as $\boldsymbol{\psi}(\boldsymbol{x}, y, \boldsymbol{h}) = \big(\mathbf{0}, \ldots, \boldsymbol{\psi}_y^{\mathrm{gl}}, \boldsymbol{\psi}_y^{\mathrm{loc}}, \boldsymbol{\psi}_{y,c_1,\ldots,c_k}^{\mathrm{sh}}, \ldots, \mathbf{0}\big)$, where the components – again exemplified for object classification – are:

Global Features: $\boldsymbol{\psi}_y^{\mathrm{gl}} = \boldsymbol{\phi}(\boldsymbol{x})$ is a histogram vector, *e.g.* a histogram of quantized densely sampled SIFT descriptors [Lowe, 1999a] (for the object classification experiments) over the whole image $\boldsymbol{x}$ by using the spatial pyramid (SP) representation [Wang et al., 2010]. We use the same codebook to build the global, local and shared features. However, we use different visual vocabulary sizes for each dataset and we refer to Section 5.5 for more details. For the global features, we use three levels $(1 \times 1, 2 \times 2, 4 \times 4)$ for the SP.

Local Features: $\boldsymbol{\psi}_y^{\mathrm{loc}} = \boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{h}_y^{\mathrm{loc}})$ is a histogram over an image part selected with window $\boldsymbol{h}_y^{\mathrm{loc}}$, which roughly corresponds to a bounding box $\boldsymbol{h}_y^{\mathrm{loc}}$ around the instance of the target class. We use a two-level SP $(1 \times 1, 2 \times 2)$ over quantized SIFT descriptors for the local feature vector $\boldsymbol{\phi}(\boldsymbol{x}, y, \boldsymbol{h}_y^{\mathrm{loc}})$.

Shared Features: $\boldsymbol{\psi}_{y,\hat{y}}^{\mathrm{sh}} = K_{\mathcal{S}}(y, \hat{y}) \boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{h}_{y,\hat{y}}^{\mathrm{sh}})$ is a histogram over a window $\boldsymbol{h}_{y,\hat{y}}^{\mathrm{sh}}$. It is a two-level SP $(1 \times 1, 2 \times 2)$ over quantized SIFT descriptors. Suppose $\mathcal{S}$ is the set of all class pairs of on the one hand the target class y and on the other hand each one of the auxiliary classes with which the target class is supposed to share information. $K_{\mathcal{S}}(y, \hat{y})$ is an indicator function that outputs 1, if the label pair $(y, \hat{y}) \in \mathcal{S}$, and else is 0. Note that $K_{\mathcal{S}}(y, \hat{y}) = K_{\mathcal{S}}(\hat{y}, y)$. We explain the procedure to obtain $\mathcal{S}$ in Section 5.4.

We can now rewrite the discriminant function (5.1) by including these feature vectors:

$$f_{\boldsymbol{w}}(x, y, \boldsymbol{h}) = \boldsymbol{w}_y^{\mathrm{gl}\mathsf{T}} \boldsymbol{\phi}(\boldsymbol{x}) + \boldsymbol{w}_y^{\mathrm{loc}\mathsf{T}} \boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{h}_y^{\mathrm{loc}}) + \sum_{\hat{y} \in \mathcal{Y}} K_{\mathcal{S}}(y, \hat{y})\, \boldsymbol{w}_{y,\hat{y}}^{\mathrm{sh}\mathsf{T}} \boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{h}_{y,\hat{y}}^{\mathrm{sh}}) \quad (5.2)$$

where $\boldsymbol{w}_y^{\mathrm{gl}}$,$\boldsymbol{w}_y^{\mathrm{loc}}$, $\boldsymbol{w}_{y,\hat{y}}^{\mathrm{sh}}$ denote the parts of $\boldsymbol{w}_y$ that correspond to the global, local, and shared parameter vectors respectively, *i.e.* we define $\boldsymbol{w}_y = \left(\boldsymbol{w}_y^{\mathrm{gl}}, \boldsymbol{w}_y^{\mathrm{loc}}, \boldsymbol{w}_{y,c_1}^{\mathrm{sh}}, \ldots, \boldsymbol{w}_{y,c_k}^{\mathrm{sh}}\right)$ and $\boldsymbol{w} = (\boldsymbol{w}_{c_1}, \ldots, \boldsymbol{w}_{c_k})$. The set of latent parameters can similarly be written as $\boldsymbol{h}_y = \left(\boldsymbol{h}_y^{\mathrm{loc}}, \boldsymbol{h}_{y,c_1}^{\mathrm{sh}}, \ldots, \boldsymbol{h}_{y,c_k}^{\mathrm{sh}}\right)$ and $\boldsymbol{h} = (\boldsymbol{h}_{c_1}, \ldots, \boldsymbol{h}_{c_k})$.

We use a common or *shared* parameter vector $\boldsymbol{w}_{y,\hat{y}}^{\mathrm{sh}}$ to encode the similarity between the labels y and $\hat{y}$. The equality $\boldsymbol{w}_{y,\hat{y}}^{\mathrm{sh}} = \boldsymbol{w}_{\hat{y},y}^{\mathrm{sh}}$ means that the classes y and $\hat{y}$ share a common parameter vector. Not adopting that equality renders the model heavier while experiments in Section 5.5.3 show a drop in performance. A graphical illustration of our model for a toy object classification task is shown in Fig.5.1. The images $\boldsymbol{x}_1, \boldsymbol{x}_2$ are labeled as c_1, c_2, *i.e* bicycle and motorbike, respectively. While there are separate class-specific parameter vectors for the global $\boldsymbol{w}_{c_1}^{\mathrm{gl}}, \boldsymbol{w}_{c_2}^{\mathrm{gl}}$ and local $\boldsymbol{w}_{c_1}^{\mathrm{loc}}, \boldsymbol{w}_{c_2}^{\mathrm{loc}}$ channels, an identical parameter vector $\boldsymbol{w}_{c_1,c_2}^{\mathrm{sh}}$ is shared between the labels c_1 and c_2. The latent parameters are used to learn instance specific shared, rectangular windows $\boldsymbol{h}_{c_1,c_2}^{\mathrm{sh}}$ and $\boldsymbol{h}_{c_2,c_1}^{sh}$ as well as the target class-specific rectangular windows $\boldsymbol{h}_{c_1}^{\mathrm{loc}}$ and $\boldsymbol{h}_{c_2}^{\mathrm{loc}}$.

5.3 Inference and Learning

5.3.1 Inference

The inference problem corresponds to finding a prediction rule that infers a class label and a set of latent parameters for an unseen image. Formally speaking, the prediction rule $g_{\boldsymbol{w}}(\boldsymbol{x})$ maximizes Eq. (5.1) over y and $\boldsymbol{h}$ given the parameter vector $\boldsymbol{w}$ and the image $\boldsymbol{x}$:

$$g_{\boldsymbol{w}}(\boldsymbol{x}) = \underset{y \in \mathcal{Y}, \boldsymbol{h} \in \mathcal{H}}{\arg\max}\, f_{\boldsymbol{w}}(\boldsymbol{x}, y, \boldsymbol{h}) \quad (5.3)$$

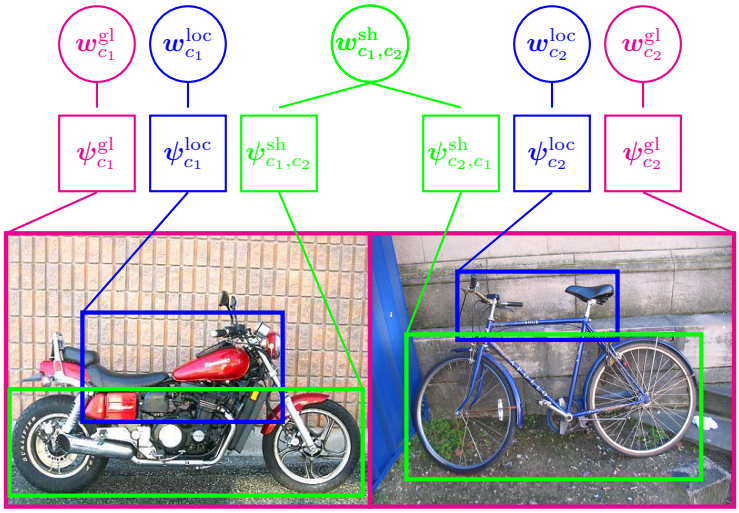


Figure 5.1: Graphical illustration of our model for two images that are labeled with class c_1 ('bicycle') and c_2 ('motorbike') respectively. In this illustration, we assume that 'bicycle' and 'motorbike' are visually similar classes and they share a common distribution of visual features. We represent each image as a sum of global features $\boldsymbol{\psi}^{\text{gl}}$ that include context information, local features $\boldsymbol{\psi}^{\text{loc}}$ that include class-specific discriminative information (*e.g.* seat and handle bar) and shared features $\boldsymbol{\psi}^{\text{sh}}$ (*e.g.* wheels) that carry common information between class-pairs. We use rectangular windows to localize the local and shared features. The windows corresponding to the global, local and shared features are drawn in magenta, blue and green respectively. The SVM models corresponding to those features are denoted with $\boldsymbol{w}$.

Since the windows corresponding to the global, local, and shared models do not depend on each other, the inference can be efficiently solved as follows:

$$g_{\boldsymbol{w}}(\boldsymbol{x}) = \arg\max_{y \in \mathcal{Y}} \Big[\boldsymbol{w}_y^{\text{gl}\mathsf{T}} \boldsymbol{\phi}(\boldsymbol{x}) + \arg\max_{\boldsymbol{h}_y^{\text{loc}} \in \mathcal{H}} \left[\boldsymbol{w}_y^{\text{loc}\mathsf{T}} \boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{h}_y^{\text{loc}}) \right] + \sum_{\hat{y} \in \mathcal{Y}, \hat{y} \neq y} \arg\max_{\boldsymbol{h}_{y,\hat{y}}^{\text{sh}} \in \mathcal{H}} \left[K_{\mathcal{S}}(y, \hat{y}) \, \boldsymbol{w}_{y,\hat{y}}^{\text{sh}\mathsf{T}} \boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{h}_{y,\hat{y}}^{\text{sh}}) \right] \Big] \tag{5.4}$$

5.3.2 Learning

Suppose we are given a set of training samples $\{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)\}$ and we want to learn a model $\boldsymbol{w}$ to predict the class label of an unseen example. Here we assume that each input $\boldsymbol{x}_i$ has only one label y_i. When the set of windows h are labeled for the training set, the problem can be solved by the standard SSSVM [Tsochantaridis et al., 2004]. Yet, as the window labels are actually not available for training the classification model, we treat them as latent parameters. Thus as before, we follow the LSSVM formulation of [Yu and Joachims, 2009] that is explained in Section 2.3.5:

$$\min_{\boldsymbol{w}} \frac{1}{2} \|\boldsymbol{w}\|^2 + C \sum_{i=1}^{n} \Bigg[\max_{\hat{y}_i, \hat{\boldsymbol{h}}_i} \left[\boldsymbol{w}^{\mathsf{T}} \boldsymbol{\psi}(\boldsymbol{x_i}, \hat{y}_i, \hat{\boldsymbol{h}}_i) + \Delta(y_i, \hat{y}_i) \right] - \max_{\hat{\boldsymbol{h}}_i} \left[\boldsymbol{w}^{\mathsf{T}} \boldsymbol{\psi}(\boldsymbol{x_i}, y_i, \hat{\boldsymbol{h}}_i) \right] \Bigg] \tag{5.5}$$

where C is the penalty parameter and $\Delta(y_i, y)$ is the loss function. The loss is taken to be $\Delta(y_i, y) = 1$ if $y_i = y$, 0 else. The above formulation yields a non-convex problem and we use the Concave-Convex Procedure (CCCP) [Yuille and Rangarajan, 2003] to solve it.

Our problem of learning the target class-specific $\boldsymbol{w}_y^{\mathrm{gl}}, \boldsymbol{w}_y^{\mathrm{loc}}$ and shared $\boldsymbol{w}_{y,\hat{y}}^{\mathrm{sh}}$ model parameters is compatible with the latent SVM formulation because the class labels and latent parameters can be optimized for each image individually.

5.4 Choosing Shared Label Pairs

We have introduced the indicator function $K_{\mathcal{S}}(y, \hat{y})$ to allow for sharing only between the class label pairs which are included in the set $\mathcal{S}$, *i.e.* $K_{\mathcal{S}}(y, \hat{y})$ is 1 if $(y, \hat{y}) \in \mathcal{S}$, else it is 0. $\mathcal{S}$ can be designed in various ways. One can include all class pairs in $\mathcal{S}$ and let the learning algorithm determine the weights $\boldsymbol{w}_{y,\hat{y}}^{\mathrm{sh}}$. However, this approach may lead to a non-optimal solution since sharing between visually very different classes can degrade the classification performance (see the baseline `(3)` in Section 5.5.3). Including all the class pairs also leads to a computational complexity that is quadratic in the number of classes. Alternatively, one can introduce additional binary latent variables to learn which class pairs should be included in $\mathcal{S}$. However, naively minimizing the loss in Eq. (5.5) with respect to those latent parameters will always result in including all the pairs.

In our experiments, we assume that the classes that are often confused with the target class in classification share enough visual similarities with the target to turn them into good candidates to build the class pairs. We thus only activate the pairwise features for such pairs. We learn a single threshold to obtain $\mathcal{S}$ from the confusion tables of the validation sets. The super-threshold class pairs extracted from the confusion table are symmetric but not necessarily transitive. For example, if the 'bicycle' class shares with 'motorbike' then also vice-versa. However, it may be that 'bicycle' shares with the class 'motorbike' and not 'bus', but 'motorbike' shares with both classes 'bicycle' and 'bus'. We provide more details about the chosen shared classes in Section 5.5.3.

5.5 Experiments

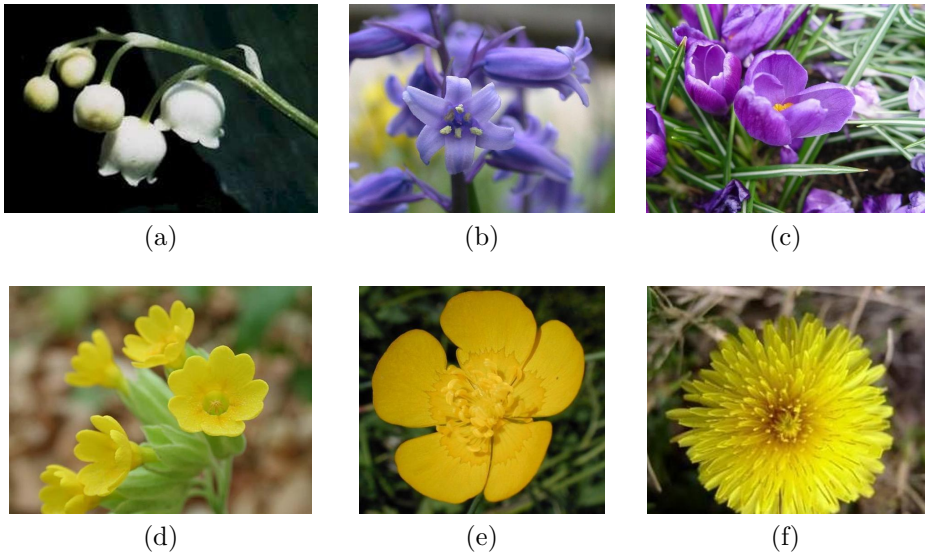
5.5.1 Datasets

We evaluate our method on the PASCAL VOC 2006 (VOC06) [Everingham et al., a], Oxford Flowers17 (Flowers17) [Nilsback and Zisserman, 2006] and TV Human Interactions (Interactions) [Patron et al., 2010] benchmarks:

VOC06: This dataset consists of 5,304 images with 10 object categories. We extract dense SIFT features [Lowe, 1999a] at every fourth pixel at a single scale and quantize them by using a 1024 words dictionary. We take the original training, validation and testing splits as in [Everingham et al., a], and remove the images with multiple class labels.

Flowers17: The dataset contains 17 flower categories and 80 images from each flower species. Figure 5.2 depicts sample images from this dataset. We compute densely sampled Lab color values and quantize them using an 800 words dictionary. The dataset has three predefined splits including 40/20/20 training-validation-testing images per class. The ground truth pixel-wise segmentation is also available for some images but it is not used in this chapter.

Interactions: This dataset contains video sequences containing four human interaction types: handshakes, high fives, hugs, kisses and an additional background class (See Figure 5.3). The videos are collected from over 20 different TV shows. We describe the videos by a set of HOF and HOG descriptors [Laptev et al., 2008] located at the detected Harris3D interest points [Laptev and Lindeberg, 2003] and quantize them using a 1024 words vocabulary.



(a) (b) (c)

(d) (e) (f)

Figure 5.2: Example images from the Flowers 17 [Nilsback and Zisserman, 2006]: This dataset contains only one object class 'flower' but 17 flower types or *sub-classes*. Here, we show samples from six sub-classes. While the flower sub-classes (a), (b), (c) in the first row are quite different than each other in terms of their color, the ones in the second row ((d),(e),(f)) are more similar and have subtler differences.

hand-shake high-five hugging kissing

Figure 5.3: Sample frames from the videos in the TV Human Interactions dataset [Patron et al., 2010]. This dataset consists of realistic human interactions from different TV shows. Classification of the interactions is challenging due to background clutter, a varying number of people in the scene, camera motion and changes of camera viewpoints.

We use the same training and testing sets as [Patron et al., 2010]. We randomly pick 40% of the original training set and use them to validate the selection of the best threshold for sharing and report the performance of our method on the original split.

5.5.2 Implementation Details

We use a sparse encoding of the BoW feature representation in [Wang et al., 2010] for all 3 of $\boldsymbol{\psi}_y^{\text{gl}}, \boldsymbol{\psi}_y^{\text{loc}}, \boldsymbol{\psi}_{y,\hat{y}}^{\text{sh}}$, with 5 nearest neighbors and the respective SPs of $(1 \times 1, 2 \times 2, 4 \times 4)$,$(1 \times 1, 2 \times 2)$ and $(1 \times 1, 2 \times 2)$ in these three cases for the images, and $(1, 2, 4)$, $(1, 2)$ and $(1, 2)$ combinations of frames for the videos. Moreover, we adopt a coarse discretization of the latent space $\mathcal{H}$ by forcing the corners to lie on an 8×8 spatial grid and at the boundaries of 32 equal temporal intervals in the case of videos. Our inference and learning algorithms scale linearly with the number of possible windows, thus this discretization significantly shortens the computation times. As our experiments have shown, defining $\mathcal{H}$ at pixel resolution did not substantially improve the classification performance.

5.5.3 Results

Baselines: In order to evaluate the contribution of the global (gl), local (loc) and shared (sh) features, we report the classification results for each of these feature types individually, and also for their combinations, *i.e.* global+local, global+shared, local+shared and global+local+shared. We refer to global and local as the baselines, corresponding to a three level SP with the LLC coding [Wang et al., 2010] and our "crop-uniform split" operation in Chapter 3, respectively. The results for the baselines and the proposed methods are depicted in Table 5.1. The table shows that the best configurations are always obtained with the shared components (*i.e.* global+shared or global+local+shared).

VOC06: We can observe from Table 5.1 that the baseline local performs better than the global one. This can be explained with the fact that the images from this dataset contain significant background noise and the objects are mostly not centered. Combining these two methods does not increase the average classification accuracy over the local one. We see that using the shared features is always useful. It improves the global, local and their combination (global+local) 3.7%, 0.4% and 2.8% respectively. We obtain the best classification accuracy with the configuration 'gl+sh'.

	VOC06	Flowers17	Interactions
global	53.8	65.6	34.4
local	54.8	63.1	35.2
global+local	54.5	68.7	37.2
global+shared	**58.2**	66.1	**40.0**
local+shared	55.2	65.2	37.6
global+local+shared	57.6	**71.1**	**40.0**

Table 5.1: Classification results with the global, local and shared features. The results are given as the classification accuracy averaged over the number of target classes, in percentages. The impact of adding each feature type (global, local and shared) are shown incrementally. The results show that including shared features always improves the classification performance.

Ours	(1)	(2)	(3)
58.2	57.6	49.6	58.1

Table 5.2: The results for three additional baselines on the VOC 2006 dataset. In (1), we do not use the shared features, however we employ multiple local models and windows. In (2) we do not localize the shared features but set the shared windows to the entire image instead. In (3) we share between all class pairs by skipping the selection procedure in Section 5.4.

We also compare our algorithm to three additional baselines ((1),(2),(3)) as shown in Table 5.2. In the configuration (1), we do not share between different class pairs, however we use multiple local windows. Although this model has a parameter vector with higher dimension, our symmetric sharing model still performs better. For the second baseline, we do not localize the shared features however we use the whole image for sharing by setting all h^{sh} to the entire image size. The result obtained from the second baseline shows that sharing information through smaller learnt windows is beneficial. For the third one, we use all the class pairs to share, *i.e.* $K_{\mathcal{S}}(y, \hat{y}) = 1$ for all $(y, \hat{y})$ pairs with $\hat{y} \neq y$. The result shows that sharing with all the label pairs lead to inferior performance. Thus it is important to find which class pairs are similar and informative for our learning.

Figure 5.4 and Figure 5.5 illustrate examples of class pairs at each row with their inferred local and shared windows from the VOC06 dataset. The local and shared windows are drawn in blue and green respectively. In Figure 5.4, we see

that the local windows contain most of the shared windows for the three class pairs. The shared windows includes visually similar parts that co-exist in both of the classes such as "wheels" for the bicycle-motorbike, "legs" and "grass" for the cow-sheep and "windows" for the bus-car pairs. In Figure 5.5, the shared windows include most of the local ones in contrast to the previous examples. Here the faces of "cats" and "dogs" are used to discriminate these two classes and are thus localized by the local windows, while their deformable bodies are localized by the shared windows. It should be noted that our localization is not very precise in these examples, as we define the local and shared windows on a coarse grid and they are only supervised by the class labels without any ground truth bounding box.

Flowers-17: For this dataset the baseline 'global' performs better than the 'local', as the flowers are usually centered in the images and the images do not contain any significant background noise (See Figure 5.2). Moreover, the global channel benefits more from the geometric configuration of the images that the local one, since it uses one more layer of spatial pyramid (3 layers). However, combining the global and local channels still yields 3.1% and 5.6% improvement over the baselines global and local respectively (See Table 5.1).

Adding the shared channel to the combined global and local achieves a further improvement of 2.4% over the 'global+local' model and 5.5% improvement over the baseline global. This is interesting as the dataset involves difficult, fine-grained (subclass) classification, suggesting that the sharing framework better exploits the subtle differences between classes.

Figure 5.6 shows images for the shared class-pairs in the Flowers-17 dataset. As the visual descriptors are quantized Lab color values for this dataset, the similarity among classes are also based on the color features. We observe that our method finds intuitive flower pairs and match the similar colored flowers with each other.

Interactions: In this dataset our crop-uniform split operation or the local channel performs better than the global one, as the interactions between two people such as hand-shake and high-five have relatively short durations compared to the whole video sequence and thus it can be important to temporally localize them. The combined global and local method gives 37.2% and improves over the individual global and local (See Table 5.1). Adding the shared models yields 40% and improves the classification accuracy 5.6% and 4.8% over the baseline global and local respectively. This is interesting as the nature of the dataset is quite different from the image classification datasets. Here the localization

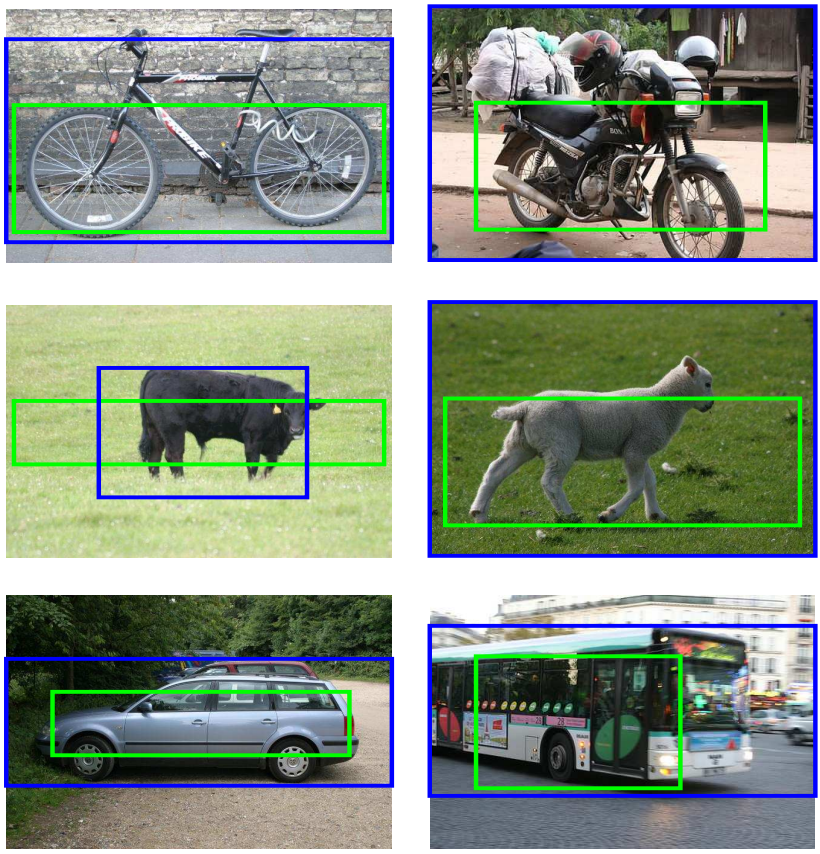


Figure 5.4: Examples of inferred windows for images from the VOC06. Each row consists of two samples for a 'class 1' and a 'class 2'. Green and blue windows correspond to the local and shared features between the class pairs respectively. We observe that 'wheels' are shared between bicycle and motorbike classes. In addition to the lower parts of body such as legs, 'sheep' and 'cow' share some green background. 'car' and 'bus' examples share their side panels including doors and windows.

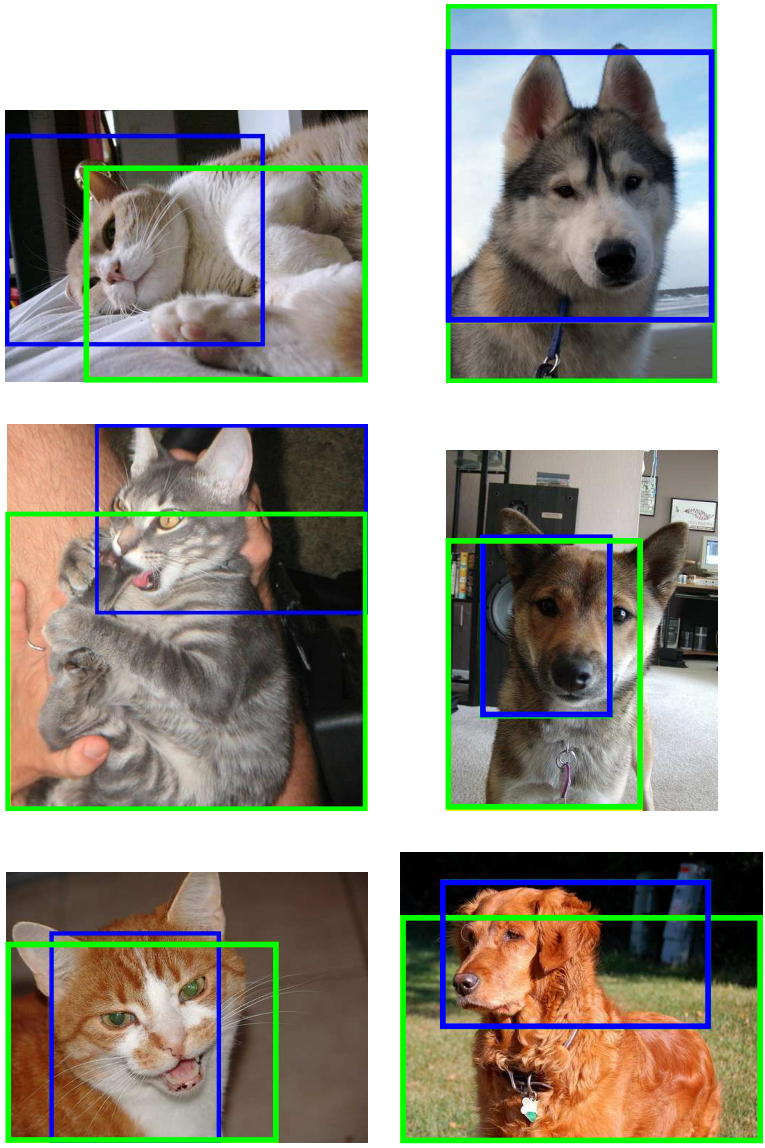


Figure 5.5: Examples of inferred windows for the "cat" and "dog" class from the VOC06. Green and blue windows correspond to the local and shared features between the class pairs respectively. We observe that "faces" of the cats and dogs are inferred for the local models and it is used to differentiate between these two classes. For the shared windows, the whole cat and dog are usually chosen.

Figure 5.6: The shared class pairs for the Flowers-17 dataset. We firstly compute the confusion table between the independently trained classifiers and compute the class pairs by using the procedure in Section 5.4. Using the quantized Lab color values as the descriptors gives the illustrated class-pairs. The image pair shows that our method finds intuitive shared pairs based on their color.

global	local	shared
$1t$	$50t$	$50t$

Table 5.3: The approximate computational cost for each feature during the inference. The computational times are given in relative units. We compute each cost by considering the feature dimensionality and possible window locations in the images. While the global feature has higher dimensionality than the local and shared ones, the local and shared components are required to be localized by scanning all the possible windows on a 8×8 grid.

is purely temporal and we use all the class pairs for sharing except the pair hand-shake and high-five.

5.5.4 Computational Complexity

In this chapter the global features are composed of three layer spatial pyramids and two layers for the local and shared ones. However, the local and shared models are required to be localized in the latent space and thus the inference of these two are roughly 50 times more computationally expensive than the global one. Table 5.3 depicts the relative computational cost of each component during the inference.

As discussed in Section 5.4, in case of sharing the features between all class pairs, the inference time is quadratic in terms of number of classes. In order to alleviate the problem, we choose only a subset of informative class pairs by computing the visually similar ones, as explained in Section 5.4. Yet this procedure does not enforce a strict limit on the number of chosen class pairs. Thus we further analyze the number of enabled class pairs for the evaluated datasets in this chapter. Table 5.4 shows the number of classes and ratio of enabled pairs over the total number of pairs on the datasets. We observe that the number of the activated pairs does not necessarily increase quadratically with the number of classes for the given datasets. The percentage of enabled label pairs is the highest in the Interactions dataset which has only five classes. However, only 5% percent of the pairs are chosen to be activated in the Flowers-17 dataset.

5.6 Conclusion

This chapter provides a method for improved visual classification by sharing localized features between selected pairs of classes. We proposed the combined

dataset	Interactions	VOC06	Flowers-17
number of classes	5	10	17
enabled pairs (#)	9	33	7
total pairs (#)	10	45	136
enabled pairs (%)	90	73	5

Table 5.4: Number and percentage of the enabled pairwise labels for the number of classes in the given datasets. The results show that the percentage of activated pairs do not increase with the number of classes.

use of global, local, and shared windows. The experimental evaluation has shown that this framework is applicable to a variety of visual classification tasks such as the classification of objects, flowers and actions. Though we have limited the approach to learning pairwise class relations in this chapter, the idea could be extended to sharing among larger class groupings by exploiting hierarchical class taxonomies. In the future, we would like to explore this idea further. We also plan to allow for the presence of multiple target classes by considering the recently proposed multi-label structured output techniques [Lampert, 2011].

Chapter 6

Conclusion

In this chapter, we first summarize the main contributions of this manuscript in the next section and then outline the limitations of our method and suggest future directions to extend and improve it in Section 6.2.

6.1 Summary of Contributions

In this manuscript, we have proposed a generic object classification method that introduces richer representations by modeling and learning different aspects of variability in object appearance with only class labels as supervision. We particularly focused on modeling location, size, appearance of objects, their interactions with their surroundings and other object classes. We presented our contributions in three subsequent chapters. In Chapter 3 and 4 we address the intra-class variation, and we target inter-class variability in Chapter 5.

In Chapter 3 we have introduced a novel object representation that relaxes and generalizes the rigid spatial pyramids [Lazebnik et al., 2006] by parameterizing location and size of the spatial pyramid. To do so, we specified two types of spatial parameters: The first type defines a cropping operation. This operation uses a bounding box to discard non-discriminative foreground and background parts. The second one specifies a splitting operation that decomposes an image into non-uniform parts. Since these parameters are not available to us, we formulated our problem as a joint learning of these unobserved and the classification parameters in a discriminative setting.

In Chapter 4, we have extended our method to handle more realistic object

and background appearances. In addition to the varying spatial configuration, object and background appearance within the same class can have a multi-modal distribution. Thus, we improved our method in the previous chapter by modeling a rough layout of object's and its background's components and with only class labels as annotation. In order to enforce coherence and cope with noise in compositions, we considered the pairwise relationships between the components. This way, we can also avoid improbable component configurations and improve the performance of the model.

In Chapter 5 our goal has been to improve object classification by better learning inter-class differences between visually similar classes. To do so, we jointly localize and learn pairwise relations between visually similar classes and this helps to improve classification. In particular our framework combines the information from three different channels: The first one encodes class-specific global context information of an image. The second one represents the class-specific local information. The last one encodes the common appearance distribution between class-pairs. We show that adding such pairwise information helps to discriminate against other classes.

Although our framework is already applicable and effective for many object classification problems, it has certain limitations. The next section addresses these limitations and possible ways of extending and improving our method.

6.2 Suggestions for Future Work

Weakly Supervised Object Detection : An interesting side outcome of our framework is that it provides a rough localization of objects in addition to their label. In this manuscript, we only optimized our method to improve classification by using a coarse localization (*i.e.* quantizing images by a uniform 8x8 grid) that ensures good classification results while being computationally efficient. The use of such a coarse grid implies that the localization results are not very accurate however. This can probably be improved by using a finer grid at the cost of increased computation time. An alternative strategy is non-uniformly sampling image parts as in [Alexe et al., 2010, van de Sande et al., 2011] by eliminating the candidates that are less likely to have objects.

A possible problem with using only supervision of labels is that optimizing for class labels but not for the annotated bounding boxes does not necessarily lead to an optimal localization. We observed that in many cases the most discriminative bounding box for classification is not the one containing the entire object. Differently from object detection methods, our method is not required to localize objects accurately but instead can use bounding boxes to

discard object parts that are not helpful for classification, while keeping the helpful ones in. Moreover it can very well include parts of the background due to the informative context. A promising research direction to improve weakly supervised *detection* can be learning general characteristics of objects (*e.g.* closed boundary and saliency as in [Alexe et al., 2010]) and incorporate them as prior knowledge.

In this manuscript, we have explored to what extent the classification can be improved with annotations limited to class labels and whether an expressive model can still be learned at all in the absence of more detailed annotations. Although annotation can be time consuming and costly, there are already some annotated images in commonly used benchmarks. We can combine the different datasets with and without annotations to form more general datasets and learn classifiers on this combined dataset in a semi-supervised way. It is also possible to incorporate these existing annotations to our framework while modeling the samples without any ground truth annotations with the latent parameters. This can probably help to improve the detection results.

Exploiting Object Hierarchies : In Chapter 5 we have exploited the similarities among groups of visually similar classes to improve classification. We limited these groups to only class pairs and used the ones that are confused during classification. However, our learning framework already supports sharing among larger class groupings. A promising direction to extend our method to larger groupings is the use of hierarchical representations that are typically built by top-down or bottom-up clustering techniques based on a similarity measure between classes. While one can obtain such hierarchy-based on visual similarity [Marszałek and Schmid, 2008, Griffin and Perona, 2008], semantic information can also be used, as recent work [Deselaers and Ferrari, 2011] has shown that semantic similarities are correlated with the visual ones.

In this manuscript, we have limited our framework to binary (*i.e.* is class X present in the image?) and multi-class (*i.e.* which one of a pre-specified number of classes present in the image?) output spaces. While we train our classifiers in such output spaces, the optimization algorithm penalizes misclassification evenly by ignoring similarity relationships between classes. For instance, the zero-one loss function penalizes confusing a car with a dog and confusing a cat with a dog evenly. A more intuitive and human-like error can suggest that confusing a car with a dog should be more severely penalized than confusing a cat with a dog. Such a loss function can be formalized and defined based on a structural representation like a pre-determined taxonomy or hierarchy. We have built our learning method on the Structural SVM [Tsochantaridis et al., 2004] and thus it can possibly be extended to such structured output spaces.

Limits of Learning with Latent Parameters : In the current framework, we model certain aspects of object and background appearances as well as their interactions with latent variables, since the annotation thereof was not present for training. In order to jointly learn classification and those latent variables, we use latent variable models which provide an elegant formulation and principled way for our problems. An interesting question could be the limits of latent learning and the possibility of extending our framework by adding more latent parameters while still improving the classification. We further detail the question here by focusing on two specific challenges for designing latent models.

Designing successful latent variable models demands latent variables to have certain properties such as being visual and discriminative in order to enhance classification. First, the added latent variable requires having a corresponding visual property in images and this visual property has to be represented by appropriate low level features. Second, the corresponding visual features need to be commonly present in images of the class and distinctive enough to aid class separation, since they are fed into a discriminative learning. While we can manually design latent variables as in this manuscript, automatically exploring good latent variables (as in [Elidan et al., 2000, Razavi et al., 2012]) that satisfy these two requirements in a principled way is an important research avenue. Considering the significant amount of textual and visual information available on the internet (such as in Wikipedia), automatically discovering relevant features can help us to learn better object models.

As we discuss in Chapter 3 and 4, optimization of classification with latent variable models is a non-convex problem and thus it can be quite sensitive to the initialization of latent parameters. In the mentioned chapters, we have designed initialization strategies for specific latent parameters to avoid trivial solutions and local minima during optimization to some extent. However, initialization of latent parameters in images is a challenging and open problem. A promising direction can be the use of generative models as in [Parizi et al., 2012], which obtains a distribution over latent variables instead of picking a single latent variable for each image. Another possible direction worth exploring in future work is gradually including harder samples as in [Kumar et al., 2010].

Bibliography

[Agin and Binford, 1973] Agin, G. J. and Binford, T. O. (1973). Computer description of curved objects. In *Proceedings of the 3rd international joint conference on Artificial intelligence*, pages 629–640. Morgan Kaufmann Publishers Inc.

[Ahonen et al., 2006] Ahonen, T., Hadid, A., and Pietikainen, M. (2006). Face description with local binary patterns: Application to face recognition. *PAMI, IEEE Transactions on*, 28(12):2037–2041.

[Alexe et al., 2010] Alexe, B., Deselaers, T., and Ferrari, V. (2010). What is an object? In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 73–80. IEEE.

[Alexe et al., 2012] Alexe, B., Heess, N., Teh, Y. W., and Ferrari, V. (2012). Searching for objects driven by context. In *Advances in Neural Information Processing Systems 25*, pages 890–898.

[Baumberg, 2000] Baumberg, A. (2000). Reliable feature matching across widely separated views. In *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, volume 1, pages 774–781. IEEE.

[Bay et al., 2008] Bay, H., Ess, A., Tuytelaars, T., and Van Gool, L. (2008). Speeded-up robust features (surf). *Computer vision and image understanding*, 110(3):346–359.

[Belhumeur et al., 1997] Belhumeur, P. N., Hespanha, J. P., and Kriegman, D. J. (1997). Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 19(7):711–720.

[Bengio et al., 2009] Bengio, Y., Louradour, J., Collobert, R., and Weston, J. (2009). Curriculum learning. In *Proc. Int. Conf. on Machine Learning (ICML)*, pages 41–48. ACM.

[Bilen et al., 2011] Bilen, H., Namboodiri, V. P., and Van Gool, L. (2011). Object and Action Classification with Latent Variables. In *Proceedings of The British Machine Vision Conference*.

[Bilen et al., 2012] Bilen, H., Namboodiri, V. P., and Van Gool, L. (2012). Classification with global, local and shared features. In *Proceedings of The DAGM-OAGM Conference*.

[Bilen et al., 2013a] Bilen, H., Namboodiri, V. P., and Van Gool, L. (2013a). Classification with global, local and shared features. In *Workshop on Fine-Grained Visual Categorization (FGVC2)*.

[Bilen et al., 2013b] Bilen, H., Namboodiri, V. P., and Van Gool, L. J. (2013b). Object and action classification with latent window parameters. *International Journal of Computer Vision*, pages 1–15.

[Binford, 1971] Binford, T. O. (1971). Visual perception by computer. In *IEEE conference on Systems and Control*, volume 261, page 262.

[Bishop et al., 2006] Bishop, C. M. et al. (2006). *Pattern recognition and machine learning*, volume 1. springer New York.

[Blaschko et al., 2010] Blaschko, M. B., Vedaldi, A., and Zisserman, A. (2010). Simultaneous object detection and ranking with weak supervision. In *Proc. of Advances in Neural Information Processing Systems (NIPS)*.

[Boiman et al., 2008] Boiman, O., Shechtman, E., and Irani, M. (2008). In defense of nearest-neighbor based image classification. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE.

[Boureau et al., 2010] Boureau, Y.-L., Bach, F., LeCun, Y., and Ponce, J. (2010). Learning mid-level features for recognition. In *CVPR*, pages 2559–2566.

[Breiman, 2001] Breiman, L. (2001). Random forests. *Machine learning*, 45(1):5–32.

[Chatfield et al., 2011] Chatfield, K., Lempitsky, V., Vedaldi, A., and Zisserman, A. (2011). The devil is in the details: an evaluation of recent feature encoding methods. In *British Machine Vision Conference*.

[Choi et al., 2010] Choi, M. J., Lim, J. J., Torralba, A., and Willsky, A. S. (2010). Exploiting hierarchical context on a large database of object categories. In *CVPR 2010*, pages 129–136. IEEE.

[Cover and Hart, 1967] Cover, T. and Hart, P. (1967). Nearest neighbor pattern classification. *Information Theory, IEEE Transactions on*, 13(1):21–27.

[Csurka et al., 2004] Csurka, G., Dance, C., Fan, L., Willamowski, J., and Bray, C. (2004). Visual categorization with bags of keypoints. In *Workshop on statistical learning in computer vision, ECCV*, volume 1, page 22.

[Dalal and Triggs, 2005] Dalal, N. and Triggs, B. (2005). Histograms of oriented gradients for human detection. In *In CVPR*, pages 886–893.

[Dekel et al., 2004] Dekel, O., Keshet, J., and Singer, Y. (2004). Large margin hierarchical classification. In *International Conference on Machine Learning (ICML)*, pages 27–35.

[Demšar, 2006] Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine Learning Research*, 7:1–30.

[Desai et al., 2009] Desai, C., Ramanan, D., and Fowlkes, C. (2009). Discriminative models for multi-class object layout. In *ICCV 2009*, pages 229–236. IEEE.

[Deselaers et al., 2010] Deselaers, T., Alexe, B., and Ferrari, V. (2010). Localizing objects while learning their appearance. In *Computer Vision–ECCV 2010*, pages 452–466. Springer.

[Deselaers and Ferrari, 2011] Deselaers, T. and Ferrari, V. (2011). Visual and semantic similarity in imagenet. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 1777–1784. IEEE.

[Dickinson, 2009] Dickinson, S. (2009). The evolution of object categorization and the challenge of image abstraction. *Object categorization: computer and human vision perspectives*, 7.

[Duchenne et al., 2011] Duchenne, O., Joulin, A., and Ponce, J. (2011). A graph-matching kernel for object categorization. In *ICCV 2011*, pages 1792–1799. IEEE.

[Dunn, 1961] Dunn, O. (1961). Multiple comparisons among means. *Journal of the American Statistical Association*, 56(293):52–64.

[Edelman, 2009] Edelman, S. (2009). On what it means to see, and what we can do about it. *Object Categorization: Computer and Human Vision Perspectives*, pages 69–86.

[Elidan et al., 2000] Elidan, G., Lotner, N., Friedman, N., Koller, D., et al. (2000). Discovering hidden variables: A structure-based approach. In *NIPS*, volume 13, pages 479–485.

[Everingham et al., 2010] Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., and Zisserman, A. (2010). The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88(2):303–338.

[Everingham et al., a] Everingham, M., Zisserman, A., Williams, C. K. I., and Van Gool, L. The PASCAL Visual Object Classes Challenge 2006 (VOC2006) Results. http://www.pascal-network.org/challenges/VOC/voc2006/results.pdf.

[Everingham et al., b] Everingham, M., Zisserman, A., Williams, C. K. I., and Van Gool, L. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results.

[Farquhar et al., 2005] Farquhar, J., Szedmak, S., Meng, H., and Shawe-Taylor, J. (2005). Improving" bag-of-keypoints" image categorisation: Generative models and pdf-kernels.

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

[Felzenszwalb et al., 2010] Felzenszwalb, P. F., Girshick, R. B., McAllester, D. A., and Ramanan, D. (2010). Object detection with discriminatively trained part-based models. *IEEE Trans. Pattern Anal. Mach. Intell.*, 32(9):1627–1645.

[Felzenszwalb and Huttenlocher, 2000] Felzenszwalb, P. F. and Huttenlocher, D. P. (2000). Efficient matching of pictorial structures. In *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, volume 2, pages 66–73. IEEE.

[Fergus et al., 2010] Fergus, R., Bernal, H., Weiss, Y., and Torralba, A. (2010). Semantic label sharing for learning with many categories. In *ECCV*, pages 762–775.

[Fergus et al., 2003] Fergus, R., Perona, P., and Zisserman, A. (2003). Object class recognition by unsupervised scale-invariant learning. In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, volume 2, pages II–264. IEEE.

[Fix and Hodges, 1951] Fix, E. and Hodges, J. L. (1951). Discriminatory analysis, nonparametric discrimination: Consistency properties. *US Air Force School of Aviation Medicine*, Technical Report 4(3):477+.

[Friedman, 1937] Friedman, M. (1937). The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the American Statistical Association*, 32(200):675–701.

[Gehler and Nowozin, 2009] Gehler, P. V. and Nowozin, S. (2009). On feature combination for multiclass object classification. In *ICCV*, pages 221–228.

[Ghamrawi and McCallum, 2005] Ghamrawi, N. and McCallum, A. (2005). Collective multi-label classification. In *Proceedings of the 14th ACM international conference on Information and knowledge management*, pages 195–200. ACM.

[Griffin and Perona, 2008] Griffin, G. and Perona, P. (2008). Learning and using taxonomies for fast visual categorization. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE.

[Hariharan et al., 2012] Hariharan, B., Malik, J., and Ramanan, D. (2012). Discriminative decorrelation for clustering and classification. In *ECCV 2012*, pages 459–472.

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

[Hastie et al., 2001] Hastie, T., Tibshirani, R., and Friedman, J. J. H. (2001). *The elements of statistical learning*, volume 1.

[Heitz and Koller, 2008] Heitz, G. and Koller, D. (2008). Learning spatial context: Using stuff to find things. In *ECCV 2008*, pages 30–43.

[Ji et al., 2008] Ji, S., Tang, L., Yu, S., and Ye, J. (2008). Extracting shared subspace for multi-label classification. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 381–389. ACM.

[Joachims, 2005] Joachims, T. (2005). A support vector method for multivariate performance measures. In *Proceedings of the 22nd international conference on Machine learning*, pages 377–384. ACM.

[Julesz, 1981] Julesz, B. (1981). Textons, the elements of texture perception, and their interactions. *Nature*.

[Jurie and Triggs, 2005] Jurie, F. and Triggs, B. (2005). Creating efficient codebooks for visual recognition. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 1, pages 604–610. IEEE.

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

[Kirby and Sirovich, 1990] Kirby, M. and Sirovich, L. (1990). Application of the karhunen-loeve procedure for the characterization of human faces. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 12(1):103–108.

[Kolmogorov, 2006] Kolmogorov, V. (2006). Convergent tree-reweighted message passing for energy minimization. *TPAMI*, 28(10):1568–1583.

[Krapac et al., 2011] Krapac, J., Verbeek, J., and Jurie, F. (2011). Modeling Spatial Layout with Fisher Vectors for Image Categorization. In *International Conference on Computer Vision*, Barcelona, Spain.

[Kumar et al., 2010] Kumar, M. P., Packer, B., and Koller, D. (2010). Self-paced learning for latent variable models. In *Advances in Neural Information Processing Systems*, pages 1189–1197.

[Lampert, 2011] Lampert, C. (2011). Maximum margin multi-label structured prediction.

[Lampert et al., 2008] Lampert, C., Blaschko, M., and Hofmann, T. (2008). Beyond sliding windows: Object localization by efficient subwindow search. In *CVPR 2008*, pages 1 –8.

[Laptev and Lindeberg, 2003] Laptev, I. and Lindeberg, T. (2003). Space-time interest points. In *ICCV*, pages 432–439.

[Laptev et al., 2008] Laptev, I., Marszałek, M., Schmid, C., and Rozenfeld, B. (2008). Learning realistic human actions from movies. In *Proc. of Computer Vision and Pattern Recognition (CVPR)*.

[Lazebnik et al., 2006] Lazebnik, S., Schmid, C., and Ponce, J. (2006). Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *CVPR*, pages 2169–2178.

[Leibe et al., 2004] Leibe, B., Leonardis, A., and Schiele, B. (2004). Combined object categorization and segmentation with an implicit shape model. *Workshop on Statistical Learning in Computer Vision, ECCV*, pages 17–32.

[Leibe et al., 2006] Leibe, B., Mikolajczyk, K., and Schiele, B. (2006). Efficient clustering and matching for object class recognition. In *Proc. BMVC*, pages 789–798.

[Li-Jia Li and Fei-Fei, 2010] Li-Jia Li, Hao Su, E. P. X. and Fei-Fei, L. (2010). Object bank: A high-level image representation for scene classification & semantic feature sparsification. In *Advances in Neural Information Processing Systems (NIPS)*.

[Lindeberg, 1998] Lindeberg, T. (1998). Feature detection with automatic scale selection. *International journal of computer vision*, 30(2):79–116.

[Liu et al., 2006] Liu, Y., Jin, R., and Yang, L. (2006). Semi-supervised multi-label learning by constrained non-negative matrix factorization. In *Proceedings of the National Conference on Artificial Intelligence*, volume 21, page 421. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999.

[Lowe, 1999a] Lowe, D. (1999a). Object recognition from local scale-invariant features. In *ICCV*, page 1150.

[Lowe, 1984] Lowe, D. G. (1984). Perceptual organization and visual recognition. Technical report, DTIC Document.

[Lowe, 1999b] Lowe, D. G. (1999b). Object recognition from local scale-invariant features. In *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, volume 2, pages 1150–1157. Ieee.

[Lowe, 2004] Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110.

[MacQueen et al., 1967] MacQueen, J. et al. (1967). Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, page 14. California, USA.

[Mairal et al., 2009] Mairal, J., Bach, F., Ponce, J., and Sapiro, G. (2009). Online dictionary learning for sparse coding. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 689–696. ACM.

[Malisiewicz et al., 2011] Malisiewicz, T., Gupta, A., and Efros, A. A. (2011). Ensemble of exemplar-svms for object detection and beyond. In *ICCV*.

[Maree et al., 2005] Maree, R., Geurts, P., Piater, J., and Wehenkel, L. (2005). Random subwindows for robust image classification. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 34–40. IEEE.

[Marszałek and Schmid, 2007] Marszałek, M. and Schmid, C. (2007). Semantic hierarchies for visual object recognition. In *CVPR*.

[Marszałek and Schmid, 2008] Marszałek, M. and Schmid, C. (2008). Constructing category hierarchies for visual recognition. In *Computer Vision–ECCV 2008*, pages 479–491. Springer.

[Messing et al., 2009] Messing, R., Pal, C., and Kautz, H. (2009). Activity recognition using the velocity histories of tracked keypoints. In *ICCV*.

[Mikolajczyk and Schmid, 2001] Mikolajczyk, K. and Schmid, C. (2001). Indexing based on scale invariant interest points. In *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, volume 1, pages 525–531. IEEE.

[Mikolajczyk and Schmid, 2002] Mikolajczyk, K. and Schmid, C. (2002). An affine invariant interest point detector. In *Computer Vision—ECCV 2002*, pages 128–142. Springer.

[Moosmann et al., 2006] Moosmann, F., Triggs, B., and Jurie, F. (2006). Fast discriminative visual codebooks using randomized clustering forests. In Schölkopf, B., Platt, J., and Hoffman, T., editors, *Advances in Neural Information Processing Systems 19*, pages 985–992. MIT Press, Cambridge, MA.

[Muja and Lowe, 2009] Muja, M. and Lowe, D. G. (2009). Fast approximate nearest neighbors with automatic algorithm configuration. In *International Conference on Computer Vision Theory and Applications (VISSAPP'09)*, pages 331–340.

[Murase and Nayar, 1995] Murase, H. and Nayar, S. K. (1995). Visual learning and recognition of 3-d objects from appearance. *International journal of computer vision*, 14(1):5–24.

[Nemenyi, 1963] Nemenyi, P. (1963). *Distribution-free multiple comparisons*. PhD thesis, Princeton.

[Nguyen et al., 2009] Nguyen, M. H., Torresani, L., De la Torre, F., and Rother, C. (2009). Weakly supervised discriminative localization and classification: a joint learning process. In *ICCV*.

[Nilsback and Zisserman, 2006] Nilsback, M.-E. and Zisserman, A. (2006). A visual vocabulary for flower classification. In *CVPR*, volume 2, pages 1447–1454.

[Nilsson, 2010] Nilsson, N. J. (2010). *The quest for artificial intelligence*. Cambridge University Press Cambridge.

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

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

[Oliva and Torralba, 2001] Oliva, A. and Torralba, A. (2001). Modeling the shape of the scene: A holistic representation of the spatial envelope. *International journal of computer vision*, 42(3):145–175.

[Opelt et al., 2006a] Opelt, A., Pinz, A., Fussenegger, M., and Auer, P. (2006a). Generic object recognition with boosting. *IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, pages 416–431.

[Opelt et al., 2006b] Opelt, A., Pinz, A., and Zisserman, A. (2006b). Incremental learning of object detectors using a visual shape alphabet. In *CVPR*, pages 3–10.

[Pandey and Lazebnik, 2011] Pandey, M. and Lazebnik, S. (2011). Scene recognition and weakly supervised object localization with deformable part-based models. In *ICCV 2011*, pages 1307–1314.

[Parizi et al., 2012] Parizi, S. N., Oberlin, J. G., and Felzenszwalb, P. F. (2012). Reconfigurable models for scene recognition. In *CVPR, 2012 IEEE Conference on*, pages 2775–2782. IEEE.

[Patron et al., 2010] Patron, A., Marszalek, M., Zisserman, A., and Reid, I. D. (2010). High five: Recognising human interactions in tv shows. In *BMVC*, pages 1–11.

[Perronnin et al., 2010] Perronnin, F., Sánchez, J., and Mensink, T. (2010). Improving the fisher kernel for large-scale image classification. In *European Conference on Computer Vision (ECCV) (4)*, pages 143–156.

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

[Pinz, 2005] Pinz, A. (2005). Object categorization. *Foundations and Trends in Computer Graphics and Vision*, 1(4).

[Ponce and Chelberg, 1988] Ponce, J. and Chelberg, D. (1988). Finding the limbs and cusps of generalized cylinders. *International Journal of Computer Vision*, 1(3):195–210.

[Pritchett and Zisserman, 1998] Pritchett, P. and Zisserman, A. (1998). Wide baseline stereo matching. In *Computer Vision, 1998. Sixth International Conference on*, pages 754–760. IEEE.

[Quattoni and Torralba, 2009] Quattoni, A. and Torralba, A. (2009). Recognizing indoor scenes. In *CVPR*.

[Rabinovich et al., 2007] Rabinovich, A., Vedaldi, A., Galleguillos, C., Wiewiora, E., and Belongie, S. (2007). Objects in context. In *ICCV 2007*, pages 1–8. IEEE.

[Ranjbar et al., 2012] Ranjbar, M., Vahdat, A., and Mori, G. (2012). Complex loss optimization via dual decomposition. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2304–2311. IEEE.

[Razavi et al., 2012] Razavi, N., Gall, J., Kohli, P., and Van Gool, L. (2012). Latent hough transform for object detection. In *Computer Vision–ECCV 2012*, pages 312–325. Springer.

[Russakovsky et al., 2012] Russakovsky, O., Lin, Y., Yu, K., and Fei-Fei, L. (2012). Object-centric spatial pooling for image classification. In *ECCV 2012*, pages 1–15.

[Sadeghi and Farhadi, 2011] Sadeghi, M. A. and Farhadi, A. (2011). Recognition using visual phrases. In *CVPR*.

[Salakhutdinov et al., 2011] Salakhutdinov, R., Torralba, A., and Tenenbaum, J. (2011). Learning to share visual appearance for multiclass object detection. In *CVPR*.

[Salton and McGill, 1986] Salton, G. and McGill, M. J. (1986). Introduction to modern information retrieval.

[Satkin and Hebert, 2010] Satkin, S. and Hebert, M. (2010). Modeling the temporal extent of actions. In *ECCV*, pages 536–548.

[Schmid and Mohr, 1997] Schmid, C. and Mohr, R. (1997). Local grayvalue invariants for image retrieval. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 19(5):530–535.

[Schüldt et al., 2004] Schüldt, C., Laptev, I., and Caputo, B. (2004). Recognizing human actions: A local svm approach. In *ICPR (3)*, pages 32–36.

[Shapovalova et al., 2012] Shapovalova, N., Vahdat, A., Cannons, K., Lan, T., and Mori, G. (2012). Similarity constrained latent support vector machine: An application to weakly supervised action classification. In *Proc. of European Conf. Computer Vision (ECCV)*.

[Sharma and Jurie, 2011] Sharma, G. and Jurie, F. (2011). Learning discriminative spatial representation for image classification. In *British Machine Vision Conference (BMVC)*.

[Sharma et al., 2012] Sharma, G., Jurie, F., and Schmid, C. (2012). Discriminative spatial saliency for image classification. In *CVPR 2012 IEEE Conference on*, pages 3506–3513. IEEE.

[Shechtman and Irani, 2007] Shechtman, E. and Irani, M. (2007). Matching local self-similarities across images and videos. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE.

[Shotton et al., 2008] Shotton, J., Johnson, M., and Cipolla, R. (2008). Semantic texton forests for image categorization and segmentation. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE.

[Silpa-Anan and Hartley, 2008] Silpa-Anan, C. and Hartley, R. (2008). Optimised kd-trees for fast image descriptor matching. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE.

[Sivic and Zisserman, 2003] Sivic, J. and Zisserman, A. (2003). Video google: A text retrieval approach to object matching in videos. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 1470–1477. IEEE.

[Taskar et al., 2005] Taskar, B., Chatalbashev, V., Koller, D., and Guestrin, C. (2005). Learning structured prediction models: A large margin approach. In *Proceedings of the 22nd international conference on Machine learning*, pages 896–903. ACM.

[Torresani et al., 2010] Torresani, L., Szummer, M., and Fitzgibbon, A. (2010). Efficient object category recognition using classemes. In *ECCV*, pages 776–789.

[Tsochantaridis et al., 2004] Tsochantaridis, I., Hofmann, T., Joachims, T., and Altun, Y. (2004). Support vector machine learning for interdependent and structured output spaces. In *Proc. Int. Conf. on Machine Learning (ICML)*, page 104.

[Turk and Pentland, 1991] Turk, M. and Pentland, A. (1991). Eigenfaces for recognition. *Journal of cognitive neuroscience*, 3(1):71–86.

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

[Tuytelaars and Schmid, 2007] Tuytelaars, T. and Schmid, C. (2007). Vector quantizing feature space with a regular lattice. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8. IEEE.

[Tuytelaars and Van Gool, 2000] Tuytelaars, T. and Van Gool, L. (2000). Wide baseline stereo matching based on local, affinely invariant regions. In *British machine vision conference*, volume 2, page 4.

[Ueda and Saito, 2002] Ueda, N. and Saito, K. (2002). Parametric mixture models for multi-labeled text. In *Advances in neural information processing systems*, pages 721–728.

[van de Sande et al., 2011] van de Sande, K. E., Uijlings, J. R., Gevers, T., and Smeulders, A. W. (2011). Segmentation as selective search for object recognition. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 1879–1886. IEEE.

[van de Sande et al., 2010] van de Sande, K. E. A., Gevers, T., and Snoek, C. G. M. (2010). Evaluating color descriptors for object and scene recognition. *TPAMI*, 32(9):1582–1596.

[van Gemert et al., 2008] van Gemert, J. C., Geusebroek, J.-M., Veenman, C. J., and Smeulders, A. W. (2008). Kernel codebooks for scene categorization. In *ECCV 2008*, pages 696–709.

[Vapnik, 1999] Vapnik, V. (1999). *The nature of statistical learning theory*. springer.

[Vedaldi and Fulkerson, 2008] Vedaldi, A. and Fulkerson, B. (2008). VLFeat: An open and portable library of computer vision algorithms. `http://www.vlfeat.org/`.

[Vedaldi et al., 2009] Vedaldi, A., Gulshan, V., Varma, M., and Zisserman, A. (2009). Multiple kernels for object detection. In *ICCV*, pages 606–613.

[Vedaldi and Zisserman, 2009] Vedaldi, A. and Zisserman, A. (2009). Structured output regression for detection with partial occulsion. In *Proc. of Advances in Neural Information Processing Systems (NIPS)*.

[Viola and Jones, 2004] Viola, P. and Jones, M. J. (2004). Robust real-time face detection. *International journal of computer vision*, 57(2):137–154.

[Wang et al., 2009] Wang, H., Ullah, M. M., Kläser, A., Laptev, I., and Schmid, C. (2009). Evaluation of local spatio-temporal features for action recognition. In *BMVC*, page 127.

[Wang et al., 2010] Wang, J., Yang, J., Yu, K., Lv, F., Huang, T. S., and Gong, Y. (2010). Locality-constrained linear coding for image classification. In *CVPR*, pages 3360–3367.

[Yakhnenko et al., 2011] Yakhnenko, O., Verbeek, J., and Schmid, C. (2011). Region-Based Image Classification with a Latent SVM Model. Research Report RR-7665, INRIA.

[Yang et al., 2009] Yang, J., Yu, K., Gong, Y., and Huang, T. (2009). Linear spatial pyramid matching using sparse coding for image classification. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 1794–1801. IEEE.

[Yu and Joachims, 2009] Yu, C.-N. J. and Joachims, T. (2009). Learning structural svms with latent variables. In *Proc. Int. Conf. on Machine Learning (ICML)*, pages 1169–1176.

[Yue et al., 2007] Yue, Y., Finley, T., Radlinski, F., and Joachims, T. (2007). A support vector method for optimizing average precision. In *ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 271–278.

[Yuille and Rangarajan, 2003] Yuille, A. and Rangarajan, A. (2003). The concave-convex procedure. *Neural Computation*, 15(4):915–936.

[Zhou et al., 2010a] Zhou, X., Yu, K., Zhang, T., and Huang, T. S. (2010a). Image classification using super-vector coding of local image descriptors. In *ECCV 2010*, pages 141–154.

[Zhou et al., 2010b] Zhou, X., Yu, K., Zhang, T., and Huang, T. S. (2010b). Image classification using super-vector coding of local image descriptors. In *ECCV*, pages 141–154.

[Zhu et al., 2010] Zhu, L., Chen, Y., Yuille, A., and Freeman, W. (2010). Latent hierarchical structural learning for object detection. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 1062–1069. IEEE.

List of Publications

Articles in peer-reviewed journals

- Object and Action Classification with Latent Window Parameters.
  H. Bilen, V. P. Namboodiri, L. Van Gool
  International Journal of Computer Vision (IJCV).

International peer-reviewed conferences

- Object Classification with Adaptable Regions.
  H. Bilen, M. Pedersoli, V. P. Namboodiri, T. Tuytelaars, L. Van Gool
  Submitted to IEEE Conference of Computer Vision and Pattern Recognition (CVPR), 2014.

- Classification with global, local and shared features.
  H. Bilen, V. P. Namboodiri, L. Van Gool
  In Proceedings of The DAGM-OAGM Conference, 2012.

- Object and Action Classification with Latent Variables.
  H. Bilen, V. P. Namboodiri, L. Van Gool
  In Proceedings of The British Machine Vision Conference (BMVC), 2011.
  (Best Paper).

Other Publications

- Classification with global, local and shared features.
  H. Bilen, V. P. Namboodiri, L. Van Gool
  In Workshop on Fine-Grained Visual Categorization (FGVC2), 2013.

- Action Recognition: A Region Based Approach
  H. Bilen, V. P. Namboodiri, L. Van Gool
  IEEE Workshop Applications of Computer Vision (WACV), 2011.

FACULTY OF ENGINEERING SCIENCE
DEPARTMENT OF ELECTRICAL ENGINEERING
PSI - VISICS
Kasteelpark Arenberg 10 box 2441
B-3001 Heverlee
www.esat.kuleuven.be/psi/visics