Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Sanja Fidler

Prepoznavanje vizualnih kategorij s podprostorskimi metodami in z naučenim hierarhičnim slovarjem oblik

Doktorska disertacija

Mentor: doc. dr. Gašper Fijavž
So-mentor: prof. dr. Tomaž Košir

Ljubljana, junij 2010

University of Ljubljana
Faculty of computer and information science

Sanja Fidler

Recognizing visual object categories with subspace methods and a learned hierarchical shape vocabulary

Doctoral Thesis Proposal

Supervisor: dr. Gašper Fijavž
Co-supervisor: prof. dr. Tomaž Košir

Ljubljana, junij 2010

Povzetek

Tema doktorskega dela je prepoznavanje vizualnih kategorij na slikah. V prvem delu doktorske disertacije bomo razvili metodo za robustno klasifikacijo objektov na podlagi izgleda, ki temelji na kombinaciji rekonstrukcijskih in diskriminatnih podprostorskih metod. V drugem delu disertacije pa bomo razvili metodo za učenje hierarhičnega kompozicionalnega slovarja oblike za predstavitev večjega števila kategorij objektov ter detekcijo le-teh z naravnih slik.

Linearne podprostorske metode, ki nudijo dobro rekonstrukcijo podatkov, kot je npr. metoda osnovnih komponent (Principal component analysis ali PCA ang.), nam omogočajo učinkovito določanje in izločanje manjkajočih slikovnih elementov, odstopajočih točk na sliki ali okluzij. Po drugi strani pa so diskriminantne metode, kot so linearna diskriminatna analiza (Linear Discriminant Analysis ali LDA ang.) in kanonična analiza komponent (Canonical Component Analysis ali CCA ang.), bolj primerne za problem klasifikacije objektov, vendar zelo občutljive na šumne podatke. V kolikor testna slika vsebuje odstopajoče slikovne elemente (ko je objekt npr. delno zakrit), diskriminantne metode lahko hitro (že v primeru zelo majhnega števila odstopajočih elementov) vrnejo napačno kategorijo objekta na sliki. V tem doktorskem delu smo razvili metodo za kombiniranje diskriminantnih in rekonstrukcijih podprostorskih pristopov za namen optimalne klasifikacije tudi v primeru šumnih testnih podatkov. Ideja predlagane metode je združiti vektorski podprostor pridobljen z diskriminatno metodo z majhnim številom baznih vektorjev pridobljenih z rekonstrukcijskim pristopom. V novem prostoru smo sposobni ugotoviti in izločiti odstopajoče slikovne elemente z robustno metodo podvzorčenja in klasificirati slike (objekte) na podlagi "dobrih" slikovnih točk. Predlagana metoda je tako sposobna robustne klasifikacije/regresije tudi v primeru, ko je zakrit večji del objekta. Učinkovitost metode je prikazana na številnih primerih računalinškega vida.

V drugem ter glavnem delu doktorske disertacije smo razvili hierarhični pristop za predstavitev, učenje in detekcijo vizualnih kategorij objektov na slikah. Hierarhične predstavitve so pomembne saj omogočajo uporabo skupnega nabora značilk med različnimi objekti na več nivojih granularnosti, so sposobne boljše generalizacije, lahko kodirajo eksponentno variabilnost na zelo kompakten način ter omogočajo računsko

učinkovito razpoznavo objektov na slikah. Zaradi omenjenih lastnosti so hierarhične predstavitve primerne za učenje in prepoznavo večjega števila kategorij objektov. Do sedaj pa se v literaturi niso pojavili učinkoviti pristopi za učenje takšnih predstavitev s slik — obstoječi pristopi v glavnem uporabljajo vnaprej določena pravila grupiranja, ki pa niso nujno prilagojena naravni statistiki oblik objektov.

V doktorski disertaciji smo razvili nov pristop za *učenje* hierarhičnega kompozicionalnega slovarja oblik za predstavitev večjega števila kategorij objektov. Pristop deluje na enostavnih robnih elementih ter se nauči njihove pogoste prostorske relacije. Le-te se rekurzivno združijo v vedno bolj kompleksne kompozicije oblike, kjer vsaka kompozicija modelira tudi fleksibilnost kodirane oblike. Na vrhnem nivoju slovarja kompozicije predstavljajo celotno obliko posameznih objektov. Slovar se učimo nivo za nivojem, tako da postopno povečujemo velikost okolice znotraj katere se lahko tvorijo kompozicije. Število ter strukturo kompozicij vsakega nivoja se naučimo samodejno iz danih podatkov. To naredimo tako, da se sprva naučimo prostorske relacije med pari delov (kompozicij iz prejšnjega nivoja slovarja), ter nato njihove kombinacije višjega reda. Spodnji nivoji so naučeni na slikah vseh kategorij objektov, višji pa so naučeni postopno, kategorija za kategorijo. Eksperimentalni rezultati kažejo na računsko učinkovitost metode za predstavitev večjega števila kategorij objektov ter prikažejo kompetitivno natančnost prepoznave v primerjavi z obstoječimi metodami.

Ključne besede: računalniški vid, podprostorske metode, robustna klasifikacija, prepoznava in detekcija vizualnih kategorij objektov, vizualno učenje, hierarhična predstavitev, kompozicionalne hierarhije

Abstract

The topic of the thesis is visual object class recognition and detection in images. In the first part of the thesis, we developed an approach that combines reconstructive and discriminative subspace methods for robust object classification. In the second part, we developed a framework for learning of a hierarchical compositional shape vocabulary for representing multiple object classes and detecting them in images.

Linear subspace methods that provide sufficient reconstruction of the data such as PCA (*Principal Component Analysis*) offer an efficient way of dealing with missing pixels, outliers, and occlusions that often appear in images. Discriminative methods, such as LDA (*Linear Discriminant Analysis*) and CCA (*Canonical Component Analysis*), which on the other hand, are better suited for classification and regression tasks, are highly sensitive to corrupted data. If an image in the test phase contains outliers (e.g. an object in an image is partly occluded), discriminative methods are likely to assign it to the wrong class. In this thesis, we propose an approach that combines discriminative and reconstructive methods in a way that enables near-to-perfect classification performance also in the case when objects during testing time are partly occluded. The idea behind the proposed approach is to augment the subspace basis given by a discriminative approach with a small set of additional basis vectors computed by a reconstructive method. In the space spanned by the augmented basis, we are able to detect and remove outlying pixels using a robust subsampling scheme and classify images based on the inliers. The proposed approach is thus capable of robust classification/regression with a high break-down point. The theoretical results are demonstrated on several computer vision tasks showing that the proposed approach significantly outperforms the standard discriminative methods in the case of missing pixels and images containing occlusions and outliers.

In the second and main part of the thesis, we will present a novel hierarchical framework for representing, learning and detecting object classes in images. Hierarchies are important, because they allow feature sharing between objects at multiple levels of representation, lead to better generalization within and across object classes, are able to code exponential variability in a very compact way and enable fast inference. This makes them potentially suitable for learning and recognizing a higher number of object

classes. However, the success of the hierarchical approaches so far has been hindered by the use of hand-crafted features or predetermined grouping rules.

In this thesis, we present a novel framework for *learning* a hierarchical compositional shape vocabulary for representing multiple object classes. The approach takes simple contour fragments and learns their frequent spatial configurations. These are recursively combined into increasingly more complex and class-specific shape compositions, each exerting a high degree of shape variability. At the top-level of the vocabulary, the compositions are sufficiently large and complex to represent the whole shapes of the objects. We learn the vocabulary layer after layer, by gradually increasing the size of the window of analysis and reducing the spatial resolution at which the shape configurations are learned. Compositions are formed by first learning spatial relations between pairs of parts (features from the previous layer) and then learning their frequent higher-order co-occurrences. The lower layers are learned jointly on images of all classes, whereas the higher layers of the vocabulary are learned incrementally, by presenting the algorithm with one object class after another. The experimental results show that the learned multi-class object representation scales favorably with the number of object classes and achieves a state-of-the-art detection performance at both, faster inference as well as shorter training times. Additionally, the learned multi-class object representation is very compact, needing only a few megabytes when stored on a computer disk. We also demonstrate the usefulness of the features learned in the intermediate layers of the hierarchy for object classification.

Keywords: computer vision, subspace methods, robust classification, visual object class recognition and detection, visual learning, hierarchical representation, part-based representations, compositional hierarchies

IZJAVA O AVTORSTVU

doktorske disertacije

Spodaj podpisani/-a _____Sanja Fidler_____,

z vpisno številko _____,

sem avtor/-ica doktorske disertacije z naslovom

Prepoznavanje vizualnih kategorij s podprostorskimi metodami in z naučenim hierarhičnim slovarjem oblik

S svojim podpisom zagotavljam, da:

- sem doktorsko disertacijo izdelal/-a samostojno pod vodstvom mentorja (naziv, ime in priimek)

  __doc. dr. Gašper Fijavž_____

  in somentorstvom (naziv, ime in priimek)

  _prof. dr. Tomaž Košir_____

- so elektronska oblika doktorske disertacije, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko doktorske disertacije

- in soglašam z javno objavo elektronske oblike doktorske disertacije v zbirki »Dela FRI«.

V Ljubljani, dne __29.3.2010_____ Podpis avtorja/-ice: Sanja Fidler

Acknowledgments

I am very grateful to Prof. Neža Mramor-Kosta, Prof. Radko Osredkar, Prof. Janez Demšar for making this dissertation possible. Without their great effort to make my transfer to Computer Science possible, I would be a PhD student in Math for decades to come.

I would also like to thank the members of my Committee for their time and effort to read the thesis as well as their helpful and insightful comments. In particular, I would like to thank Prof. Tomaž Košir for encouraging me to work in the interdisciplinary field and helping me greatly through all the administrative procedures that were entailed. I am thankful for all the research meetings we had and supporting me whenever I had difficulties. I am grateful to Prof. Gašper Fijavž for all the problem discussions and his many constructive comments. I am thankful to Prof. Antonio Torralba for his excellent suggestions, which helped to improve the thesis and for taking his time to come to the defense of my thesis.

I am grateful to my family, friends and colleagues, who supported me through this long process and their understanding when I was too busy to meet. Special thanks go to my parents, who read and checked the English of most of my papers as well as the thesis and delivering pancakes to my office during many conference deadlines. Thanks to my sister for all the fun trips we made to get away from it all and clear the mind (perhaps too much, even). Most of all, I am thankful to my Grandma, who believed in me and supported me in everything I did since I was a kid and taking so much interest in my work. She is my personal heroine and I wish she would be with us to see me finishing my PhD. I miss her greatly.

Mostly, I am thankful to Prof. Aleš Leonardis and Dr. Marko Boben, without whom this thesis would not be possible. Both have contributed immensely with endless discussions, ideas, and help and I cannot thank them enough. I wish we would continue to collaborate in the future.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

Our main focus will be on visual object class recognition in two-dimensional natural images. Visual categorization of objects has been an area of active research in the computer vision community for decades and represents the 'holy grail' of the computer vision problem. This gave rise to a body of literature, of which surveys can be found in Pinz [117], Dickinson [32] and Fei Fei et al. [42]. One of the main goals behind this line of research is to endow robots with visual competencies necessary to perform cognitive tasks such as interacting and helping the disabled and the elderly [1] or execute invasive tasks. Other applications are numerous, ranging from video surveillance, image and video retrieval, medical image analysis, etc.

According to Trucco and Verri [147] object recognition can be posed with the following four (sub)problems, sorted by increasing complexity:

1. *Is this an object (class) X?* This is the simplest of the questions, whereby the location, the corresponding image patch of the object and the object class/identity is given in advance. To answer, however, we still need to consider different illuminations under which the image was taken and the possibility of the object being occluded. More importantly, if not only the identity of the object is what we are interested in but also its class (Coke bottle versus a bottle class), we must either scan through all of the class examples and decide if there exists a sufficiently good match to the given image patch (which is likely not feasible due to a large number of class examples, where not all of which might be available in the current object database) or in some way represent the distribution of the objects in a class and decide if the given example conforms to it or not.

2. *What is this object (class)?* Given an image patch that contains only one object, the task is to classify it into its corresponding class (i.e. a car, a horse, etc). Here the location of the object is assumed known, but we need to determine to which

class the depicted object belongs. The complexity from the previous problem is thus increased by n times, where n is the number of classes we are interested in.

3. *Where is this object (class) in an image?* Given an image that contains a particular object, the goal is to find its location (as well as scale and orientation) in an image. Here we presume that we know which object (class) appears in an image in advance and we must infer its position. The complexity from the problem under 1 is roughly increased by a factor $300 \times 300 \times 10$ orientations $\times 10$ scales $\approx 10^7$, where 300×300 denotes the size of an average image.

4. *Which objects (classes) are there in an image and where are they?* This is the most general question and involves recognizing the classes of all objects that appear in an image and also localizing them. The complexity from the previous problem is further increased by a factor n, where n represents the number of object classes. The problem in its most general form (where n is the order of 10^4 and where each object can additionally appear in any 3D pose) has been argued to be intractable [149], however, the remarkable performance of the human visual systems shows that good solutions exist.

The thesis will be organized into two parts. In the first part, we will be concerned with problem 2, namely recognizing an object by having known its location in an image. In the second, main part of the thesis, we will deal with the problem that lies somewhere in between 3 and 4 in terms of the underlying complexity. While recognizing all possible objects remains an illusive task, our aim will be to recognize and localize multiple objects with a strong emphasis towards a larger number of object classes.

1.1 Contributions

The thesis makes several original contributions with respect to 1.) robust object recognition/classification with subspace methods, and 2.) hierarchical learning of multi-class object representations for object recognition and localization of objects in images. Additionally, we provide extensive empirical evaluation confirming the plausibility and scalability of hierarchical approaches for object class detection.

The following contributions are made in the field of visual object class recognition.

PART I: Robust classification of objects

1. We extended the broadly used linear subspace classification methods in a way to also account for the occlusion of objects in the classification phase.

Our work on robust classification of objects has been published at three international workshops [54, 53, 137] and a top-ranked computer science journal [56].

PART II: Hierarchical multi-class object representation

1. For object shape representation we developed a *hierarchical compositional shape vocabulary.*

2. We developed an algorithm for fast inference of the proposed representation from real images.

3. Our main contribution is an algorithm for *unsupervised learning* of each layer of the hierarchical vocabulary from a set of training images.

4. We introduced a similarity measure for grouping the learned vocabulary entries.

We experimentally demonstrate several important issues:

1. Applied to a collection of natural images, the approach *learns* shape structures that resemble those emphasized by the Gestalt theory.

2. We show that these generic compositions can be effectively used for object classification.

3. For object detection we demonstrate a competitive speed of detection with respect to the related approaches already for a single class.

4. For multi-class detection we achieve a highly sub-linear growth in size of the hierarchical vocabulary and, consequently, a sub-linear complexity of inference as the number of modeled classes increases.

5. We demonstrate a competitive detection performance with respect to the current state-of-the-art.

Our work on hierarchical multi-class object representation has been published at prestigious international conferences [47, 55, 90, 48, 52, 49], as a book chapter [51] and is currently under review in a top-ranked computer science journal [50].

1.2 Outline of the Thesis

The thesis is organized into two parts.

In Chapter 2 we present our robust subspace approach. We first describe the theoretical background behind discriminative and reconstructive subspace models and show how we can combine them in order to achieve robust object classification. Specifically, it is shown how to form a combined subspace which, on the one hand, enables us to detect and remove outlying pixels using a robust subsampling scheme, and on the other hand, enables us to classify objects based on the inlier pixels. The approach is validated on several problems: classification of COIL objects, face recognition under synthetic and real occlusion as well as significant changes in facial expressions, and on the task of mobile robot localization. We demonstrate that the proposed approach significantly outperforms the baseline discriminative approaches in the case when the object are occluded or images contain noise.

The remaining chapters are devoted to the main part of the thesis, namely our novel approach to learning a hierarchical compositional shape vocabulary.

We start with an introduction in Chapter 3, which motivates the use of hierarchical representations for object class recognition and detection and argues why learning them is crucial in order to achieve good and scalable performance.

Chapter 4 reviews the related work on hierarchical object representations. Specifically, we review part-based models (in particular, we focus on those that use contour information to represent the objects), neural networks, hierarchies of classifiers and compositional hierarchies. Since our proposed architecture is compositional, main emphasis is given on the latter. We review the work on supervised and unsupervised learning of compositional hierarchies for object recognition.

In Chapter 5 we propose our hierarchical learning framework. We first introduce the representation which has the form of a hierarchical compositional shape vocabulary. Next, we show how inference is done theoretically and propose how to perform it in a computationally efficient way. We then present our learning approach which is able to learn the structure and the number of compositions at each hierarchical layer without supervision. How multiple object classes are learned is discussed by proposing three possible training strategies. We also show how the intermediate features can be used for object classification.

The extensive experimental results carried out on several standard object class recognition/detection datasets are presented in Chapter 6. The method has been applied to learning generic features from natural images and the learned representation was used for object classification on the Caltech 101 dataset [41] demonstrating competitive results. The approach has been further utilized for object detection, in particular,

15 object classes have been used. We show performance in terms of speed of inference, size of the representation on disk, shareability of features between object classes, transfer of features across object classes and in terms of detection accuracy. The method is shown to be competitive with existing state-of-the art approaches with respect to the detection accuracy, but outperforms the related work with respect to the speed and scalability in the case of multiple object classes. Several different multi-class training strategies are also compared. We also show the results on learning and detecting objects on a larger scale, specifically, on 175 object classes from the LabelMe dataset [146].

The thesis concludes with a summary, discussion and directions for future work in Chapter 7.

Chapter 2

Robust Classification and Regression by Subsampling

Figure 2.1: Training of discriminative and reconstructive models is done on "clean" images. Classification is robust to outliers present in the test images (i.e., in the case when objects are occluded or images contain noise).

2.1 Introduction

Subspace methods have become a standard tool in the computer vision community for performing various types of visual learning and recognition/classification. These methods, which are based on principles originally used for statistical pattern recognition, fall into two categories. The first are the *reconstructive* and the second *discriminative methods*, both of which exert distinct, yet equally important qualities. It is known that the class of reconstructive methods, such as PCA [104] (and potentially ICA [26], and NMF [86]), produce representations that enable sufficient reconstruction thus being capable of dealing with the problem of missing pixels and outliers [89]. On the other hand, the discriminative methods, such as LDA [34], enable the construction of flexible decision boundaries needed for classification. As both types of methods have shown to be effective for recognition, the latter ones have often proved to yield better re-

sults [11, 97]. However, their usage is severely limited due to their non-robust nature, preventing them to successfully cope with outliers and occlusions, which commonly appear in the visual data.

To be widely applicable, a method should have the ability to perform *robust learning* and *robust classification/regression*. By robust we mean the ability to detect outliers (corrupted pixels) in images and consequently work on uncorrupted subsets of pixels, resulting in a high-breakdown point method. Approaches to *robust learning* of discriminative methods have been explored in literature, although only focusing on detecting an image as a whole as a data outlier and discarding it from the learning process. The vast majority of these methods involve replacing the classical location and scatter matrix estimators by their robust counterparts, such as MVE estimators [25], MCD estimators [69, 75, 124], S-estimators [29, 71], M estimators [94], and by the projection pursuit approach as in [31, 118].

On the other hand, the problem of *robust classification/regression* has rarely been addressed in the literature. This is mainly due to the highly non-robust nature of discriminative methods (with the breakdown point zero) which contain too little information to successfully deal with outliers and occlusions appearing in the visual data. Specifically, the discriminative methods provide decision hyperplanes designed for optimal classification and do not, in general, offer good reconstruction of images, which is necessary for determining the pixels which are far away from their model values (i.e., are outliers).

In contrast to discriminative methods, the reconstructive methods provide a principled way of performing robust recognition exploiting the redundancy in the visual data. These methods, which are known to produce good approximations of the data, have been proven successful in cases when images contain outliers, when the objects of interest in the images are occluded or appear on different backgrounds, or/and in the case where images are taken under varying illumination conditions. Several different robust versions of original methods have been developed, which work well under such non-ideal conditions. Some of these approaches are based on substituting the standard least-squares metric by a robust one [30], while the others calculate the coefficients by utilizing a subsampling and hypothesize-and-test approach [89].

It is therefore evident that an ideal classifier should be able to combine the best of both methods; contain information crucial for classification/regression and also enable a calculation of the necessary coefficients by means of a robust subsampling approach. To the best of our knowledge, robust classification/regression by subsampling has not been tackled before.

In this thesis we present a method, novel in the field of robust classification/regression, which makes the recognition of objects under non-ideal conditions possible, i.e., in sit-

uations when objects are occluded or they appear on a varying background, or when their images are corrupted by outliers. The main idea behind the method is to combine the reconstructive and discriminative models by constructing a basis which, on the one hand, contains the *complete* discriminative information (of a particular discriminative model) necessary for the classification/regression, and, on the other, enables us to determine outliers in images and calculate the necessary coefficients by means of a subsampling approach resulting in a high breakdown point classification/regression. Our theoretical results are evaluated on several computer vision problems showing that the proposed method significantly outperforms the standard discriminative methods in the case of corrupted images.

This chapter is organized as follows. Section 2.2 contains a review of the related work. In Section 2.3 we give a theoretical background on reconstructive and discriminative models. We formulate the problem in Section 2.4. In Section 2.5 we present our robust classification/regression method. The effectiveness of the proposed method is experimentally verified in Section 2.7 (in particular we chose LDA and CCA for demonstration purposes, although our method is general and can be used with other linear discriminative methods as well). Finally, in the last section, a summary and conclusions of this chapter are given.

2.2 Related work

Existing literature contains a number of approaches that combine different subspace methods. The classical approach is to use PCA as a preprocessing step to LDA or CCA to overcome the singularity problems these two methods encounter when dealing with high-dimensional data such as images [11, 163, 160, 19, 142]. The fact that this can be done without losing any discriminative information [160] will serve us as an idea of how to combine both discriminative and reconstructive methods to achieve robustness to image degradations. The majority of other existing methods are concerned with improving the classification power of discriminative methods by incorporating the PCA information in different ways: in [93] and [95] the authors propose to add (or average) the output feature vectors obtained by PCA, ICA and LDA or concatenating them into a single one upon which a designed RBF network returns the classification results. As these methods might outperform the classical discriminative methods under ideal conditions (when the images are "clean"), they still rely on calculating the feature vectors as a dot product between different subspace bases and the testing image vector, and thereby fail when dealing with images which contain outliers or are corrupted by noise.

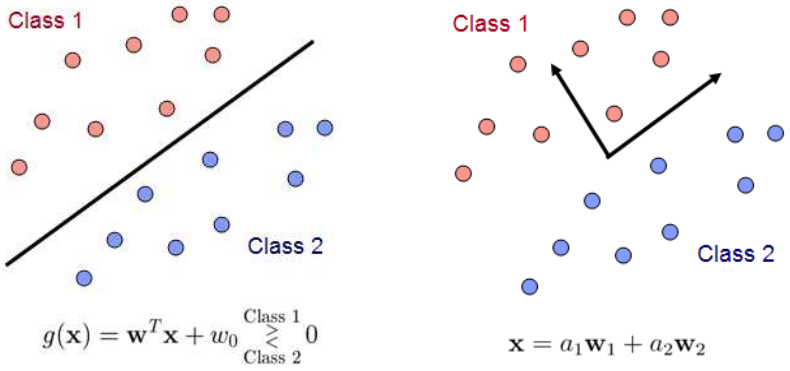
An approach focusing on the classification of degraded images has been proposed

by Stainvas et *al.* [138] and is, in its philosophy, closest to ours. The idea behind it is to improve classification of discriminative methods, which do not contain enough information to deal with corrupted data, by using the reconstruction property of the reconstructive methods. However, in their method this combination is already done in the learning stage by concurrently minimizing the mean squared error (MSE) of the reconstruction and classification outputs resulting in an improved low dimensional representation, which represents a tradeoff between reconstruction and classification confidences. This differs greatly from our approach which offers robustness in the *classification stage* by calculating the feature vectors on *subsets* of pixels in images and is consequently very robust to various image artifacts.

To the best of our knowledge classification with linear discriminative subspace methods of corrupted images have not been previously done by the subsampling approach. The closest to this is the robust PCA method of [89] which makes use of the reconstruction property of PCA to successfully detect outliers and calculate the coefficients on the rest of the pixels. We will use this idea for the discriminative methods, although due to the fact that discriminative models do not approximate the data well, the implementation is not as straightforward as suggested in [68]. With this in mind, we will combine the discriminative and reconstructive methods to achieve perfect classification/regression results (in the limits of each discriminative method) and on the other hand, have the reconstruction abilities for detection of outliers and occlusions.

2.3 Theoretical background

The central problem when working with high-dimensional data in a learning system is to find a suitable representation of the data by means of an "optimal" transformation for the task at hand. The definition of optimality varies from task to task, also depending on the knowledge one has about the learning dataset. In the case when little of such knowledge exists the transformation is usually defined in the sense of optimal dimension reduction, statistical "interestingness", positiveness or simplicity of the transformation. Since the learning process is thereby unsupervised the representation has to be as informative as possible, thus mainly having the property of well approximating the original data. These methods are referred to as the *reconstructive methods*. On the other hand, in the case when one has prior knowledge of the class labels usually discriminative hyperplanes that best separate the classes are sought for. The learning process is supervised, the representation obtained does not usually provide good reconstruction of the data, is more task dependent, but spatially and computationally much more efficient and often gives superior classification/regression results compared to the reconstructive methods. These methods are addressed to as *discriminative methods*. Both types of models are illustrated in Figure 2.2.



(a) (b)

Figure 2.2: (a) Discriminative vs (b) reconstructive linear models.

In the following, we will present a general theoretical background for both, reconstructive and discriminative methods, and introduce the notation. We would like to emphasize that we shall only consider linear subspace methods.

2.3.1 Notation

Let n be the number of images in the training dataset, each of them containing m pixels, and let c be the number of classes the images belong to (when such prior knowledge exists). We write images as vectors and arrange them (according to the classes) in columns of a matrix $X = [\mathbf{x}_1, \ldots, \mathbf{x}_n]$, $\mathbf{x}_i \in \mathbb{R}^m$. For a simpler notation we will assume X to be centered, i.e., having zero mean, unless otherwise specified.

We will use the notation U for the basis of the reconstructive methods and W for the basis of the discriminative methods. With A we will denote the feature matrix composed of feature vectors $\mathbf{a}_i$ expressed in the basis of the reconstructive model. We will also frequently operate with submatrices. For $\mathcal{I}$ and $\mathcal{J}$, being nonempty subsets of the set $\{1, 2, \ldots, n\}$, the symbol $M_{\mathcal{I}:}$ will be used to denote the submatrix of M containing only those *rows* of M whose indices are in the set $\mathcal{I}$, arranged in their natural order, and similarly, $M_{:\mathcal{J}}$ will stand for the submatrix of M containing only those *columns* of M whose indices are in the set $\mathcal{J}$. For a vector $\mathbf{a}$, the symbol $\mathbf{a}_\mathcal{I}$ will be used to denote those elements of $\mathbf{a}$ whose indices are in $\mathcal{I}$. In our calculations we will mainly be operating with two sets, $\mathcal{K} = \{1, 2, \ldots, k\}$ and $\mathcal{N} - \mathcal{K} = \{k+1, k+2, \ldots, n\}$. Thus if a matrix M consists of n *columns*, $M = \left[M_{:\mathcal{K}}, M_{:(\mathcal{N}-\mathcal{K})}\right]$ and if it consists of n *rows*, $M = \begin{bmatrix} M_{\mathcal{K}:} \\ M_{(\mathcal{N}-\mathcal{K}):} \end{bmatrix}$.

2.3.2 Reconstructive methods

Reconstructive methods fall into the category of unsupervised learning since no prior knowledge of the class labels is presumed to be avaliable. The main goal is to find a linear representation

$$X = UA \tag{2.1}$$

$$\text{where} \qquad U = [\mathbf{u}_1, \ldots, \mathbf{u}_n] \in \mathbb{R}^{m \times n}, \quad A \in \mathbb{R}^{n \times n} \tag{2.2}$$

that best describes the data subject to different criteria. Here U is called the *basis matrix*, while the matrix of coefficients, $A = (U^T U)^{-1} U^T X$, is referred to as the *feature matrix*. If U is an orthonormal matrix, A simplifies to $A = U^T X$.

By far most widely known and used method is Principal Component Analysis (PCA) which seeks for a low dimensional representation of the data which minimizes the squared reconstruction error [151]. PCA can also be interpreted as searching for a linear transformation that minimizes the statistical dependencies of second order between the transformed data, thus PCA finds a basis $\{\mathbf{u}_i\}_{i=1}^n$ that yields mutually uncorrelated coefficient vectors. This is in contrast to Independent Component Analysis which finds a basis of mutually independent vectors by also minimizing higher-order statistical dependencies [12, 26]. Another method that recently gained attention is Non-negative Matrix Factorization (NMF) which seeks for a representation that has all the coefficients and the basis vectors non-negative (here the data matrix X is obviously not centered) [86].

After the optimal basis is obtained it can then be reduced to $U_{:\mathcal{K}}$, where k indicates that usually only k, $k \ll n$, basis vectors (those that take up the most variance) are needed to represent $\mathbf{x}$ to a sufficient degree of accuracy as their linear combination

$$\tilde{\mathbf{x}} = \sum_{j=1}^{k} a_j(\mathbf{x}) \mathbf{u}_j = U_{:\mathcal{K}} \big((U_{:\mathcal{K}}^T U_{:\mathcal{K}})^{-1} U_{:\mathcal{K}}^T \mathbf{x} \big) \ . \tag{2.3}$$

Here, $\tilde{\mathbf{x}}$ denotes the approximation to $\mathbf{x}$ and a_j are the coefficients obtained by projecting $\mathbf{x}$ onto the selected basis, $\mathbf{a}_{\mathcal{K}} := [a_1, a_2, \ldots, a_k]^T = (U_{:\mathcal{K}}^T U_{:\mathcal{K}})^{-1} U_{:\mathcal{K}}^T \mathbf{x}$. If the basis U is orthonormal, $\mathbf{a}_{\mathcal{K}} = U_{:\mathcal{K}}^T \mathbf{x}$.

In the theory that follows we will not choose any of the mentioned method in particular and try to stay general throughout the approach. Still, it might be worth emphasizing that the most appealing of the stated methods is PCA since it is optimal in reconstruction error and can therefore detect outlying pixels and occlusions [89] to a larger degree of accuracy than the other methods.

2.3.3 Discriminative methods

Discriminative methods were designed particularly for classification/regression tasks. They assume that prior knowledge about classes of the training data is available, which is then integrated in the supervised learning process to produce a small number of hyperplanes that are capable of separating the training data with no (or little) error.

To be more specific, the objective of discriminative methods is to find a linear function

$$\begin{aligned} g(\mathbf{x}) &= W^T\mathbf{x}\ , \\ \text{where} \quad W &= [\mathbf{w}_1, \ldots, \mathbf{w}_c] \in \mathbb{R}^{m\times c} \end{aligned} \tag{2.4}$$

which is used for transforming the data into a lower-dimensional classification space upon which it is decided, according to some chosen metric, to which class a given sample $\mathbf{x}$ belongs. To find an optimal decision function, a number of different criteria can be employed.

Probably the most widely used for classification is Linear Discriminant Analysis which, in the training stage, finds the projection directions on which the intra-class scatter is minimized whilst the inter-class scatter is maximized. Specifically, LDA maximizes the objective function, also called the Fisher criterion function [60], which is defined as

$$J(W) = \frac{\mathbf{w}^T S_b \mathbf{w}}{\mathbf{w}^T S_w \mathbf{w}}\ , \tag{2.5}$$

where S_b denotes the between-class and S_w the within-class scatter matrix of the training data. In the classification stage the new image samples are projected onto these directions to form feature vectors according to which samples are classified to a certain class.

On the other hand, Canonical Correlation Analysis is best suited for regression tasks [101, 18]. CCA is a supervised method, which relates two sets of observations, one set being composed of training images and the other set of the corresponding measurements (e.g., orientations or positions of an object). In the *training stage* CCA finds pairs of directions (canonical correlation vectors) that yield maximum correlation between the projections of input vectors. This is followed by performing linear regression on the obtained projections (canonical correlation coefficients). More specifically, given n pairs of mean-centered observations ($\mathbf{x}_i \in \mathbb{R}^p,\ \mathbf{y}_i \in \mathbb{R}^q$), $i = 1, \ldots, n$, aligned in the data matrices $X = [\mathbf{x}_1, \ldots, \mathbf{x}_n] \in \mathbb{R}^{p\times n}$ and $Y = [\mathbf{y}_1, \ldots, \mathbf{y}_n] \in \mathbb{R}^{q\times n}$, CCA finds $c = \min(p, q)$ pairs of directions $\mathbf{w}_\mathbf{x} \in \mathbb{R}^p$ and $\mathbf{w}_\mathbf{y} \in \mathbb{R}^q$ that maximize the correlation between the projections $\mathbf{w}_\mathbf{x}^T\mathbf{x}_i$ and $\mathbf{w}_\mathbf{y}^T\mathbf{y}_i$. CCA maximizes the function

$$\rho = \frac{\mathbf{w}_\mathbf{x}^T C_{\mathbf{xy}} \mathbf{w}_\mathbf{y}}{\sqrt{\mathbf{w}_\mathbf{x}^T C_{\mathbf{xx}} \mathbf{w}_\mathbf{x} \mathbf{w}_\mathbf{y}^T C_{\mathbf{yy}} \mathbf{w}_\mathbf{y}}}\ , \tag{2.6}$$

where $C_{\mathbf{xx}}$, $C_{\mathbf{yy}}$, $C_{\mathbf{xy}}$, and $C_{\mathbf{yx}}$ are within-set and between-set covariance matrices of the input data. In the *regression stage*, the orientation (or position) of the object is estimated by using canonical correlation coefficients obtained from a novel image of the object.

From here on we will not deal with robustness in the training stage and presume that the training stage of each method used was already performed. Our main concern is *robustness in the classification or regression stage*. By robust we mean the ability to correctly classify a novel image which contains a large number of corrupted pixels (outliers).

2.4 Defining the problem

The comparison between discriminative and reconstructive methods for classification and regression tasks has been a subject of extensive research and testing [11, 97]. The general conclusion was that under ideal circumstances (when images do not contain artifacts such as noise or outliers) discriminative methods outperform the reconstructive methods. The explanation for this is rather obvious: the discriminative methods are supervised and the information can thus be more efficiently integrated into the learning process. These methods in most cases offer much lower-dimensional linear subspaces than the reconstructive methods. But there is a trade-off to this: by having fewer basis vectors, these methods do not usually provide good approximation of the data which is necessary for successful detection of outliers (corrupted pixels) and occlusion.

Images often contain noise or outliers, that is pixels that do not belong to objects being depicted. Therefore tools must exist that are able to extract reliable information based on only uncorrupted subsets of pixels. Since both, reconstructive as well as discriminative methods, rely on calculating the dot product $U^T\mathbf{x}$ (appearing in the calculation of coefficients in (2.3)) and $W^T\mathbf{x}$ (needed in a linear classifier g in (2.4)), respectively, they obviously take into account *all* pixels in an image $\mathbf{x}$. The results can therefore be wrong by having even a small amount of outlying pixels.

However, this undesirable property of linear methods has been successfully overcome for reconstructive methods (in particular PCA) by employing the *robust coefficient estimation procedure* [89]. The basic idea behind the approach is to translate the original dot product calculation into solving an over-determined set of equations using only subsets of pixels. The obtained coefficients are used for back-projection and the pixels that deviate the most from the expected approximation error are pronounced to be outliers. The better the reconstruction the given basis provides, the more reliable detection of outliers is, and consequently, the more exact the obtained coefficients are. The details of the approach are given in Section 2.6.

As was already mentioned the discriminative methods usually give only a small number of basis vectors which do not offer a satisfactory reconstruction property that would enable a correct detection of outlying pixels in an image and consequently a reliable calculation of the linear classifier g. In this thesis we will present a method which shows how to construct a basis that, on one hand, contains sufficient reconstructive information to enable the use of the subsampling approach [89] and, on the other hand, contains all discriminative information for classification/regression.

We would like to emphasize that our aim is not to improve classification/regression power of the standard discriminative methods in the case of ideal data, but to achieve similar results also in the case of corrupted data.

2.5 Robust classification/regression approach

The classification/regression of discriminative models is based on a linear function

$$g(\mathbf{x}) = W^T \mathbf{x} \ , \tag{2.7}$$

which is used for transforming the data into a lower-dimensional classification space upon which it is decided, according to some chosen metric, to which class a novel image $\mathbf{x}$ belongs.

What we will do is rewrite the dot product in (2.7) into a form that will enable robust estimation. This will be done by incorporating the basis of a reconstructive model into our classification function by employing a few linear algebra operations.

To begin with, let $U \in \mathbb{R}^{m \times n}$ denote the complete basis of a reconstructive model. Let us first point out that our robust method will not need the complete reconstructive basis, this is meant exclusively to give a justification to our final calculations. For the purpose of clarity let us also assume that the reconstructive basis is orthonormal. The extension to an non-orthogonal basis is then just a matter of matrix manipulation.

To rewrite the expression in (2.7) we will use the fact that both bases, U and W, lie in the span of the training data vectors. Since $X = UA$ (by definition in (2.1)) this obviously holds for the reconstructive models. Moreover, because U has rank n it spans *exactly* the same space as the training data[1]. As it might be intuitively obvious that the discriminative basis also "lives" in the learning data space, there is no proof to cover all the discriminative models, therefore each of them has to be dealt with separately[2]. Since U spans the same space as X and W is a subspace of this space,

[1]To be exact, U is usually of rank $n-1$, because the vectors in X are mean-centered, or even smaller if some input images are linearly dependent. However, the same observation holds also for X, thus this fact does not influence our derivation.

[2]The proof in the LDA case is due to the recent paper of Yang et al [160] who showed that LDA

the immediate consequence is that the discriminative basis can be written in the basis of the reconstructive model, i.e. $W = UV$, where $V \in \mathbb{R}^{n \times c}$. Note that V is a matrix that can be calculated already in the training stage as a projection of W onto U (if U is orthogonal, then $V = U^T W$, otherwise $V = (U^T U)^{-1} U^T W$).

Similarly, assuming a novel image $\mathbf{x}$ (which we want to classify) follows a distribution of one of the classes in the training set, it can therefore be written with little error as a linear combination of the basis vectors of the reconstructive model, $\mathbf{x} \approx a_1 \mathbf{u}_1 + \cdots + a_n \mathbf{u}_n = U\mathbf{a}$. The classification function now takes the following form:

$$g(\mathbf{x}) = W^T \mathbf{x} = (V^T U^T)(U\mathbf{a}) = V^T \mathbf{a} \ . \tag{2.8}$$

We have rewritten the function into an expression that uses the feature vectors corresponding to the reconstructive model. This is a promising start, since an efficient algorithm already exists for robust calculation of the coefficient vector $\mathbf{a}$ of the reconstructive model in cases when the data contains outliers, occlusion or non-gaussian noise [89]. We will employ this method, but not just yet. Notice that the expression in (2.8) demands *all* n coefficients for its calculation. Since in most computer vision applications the number of training images n is large, the computational complexity of the robust estimation of all the coefficients would be too prohibitive to make this method applicable in practice. The idea is to use only a truncated basis $U_{:\mathcal{K}}$ of $k \ll n$ basis vectors which is usually used for calculations involving reconstructive methods. The number k is chosen so that the truncated basis approximates the data to a good degree of accuracy.

However, this truncated reconstructive basis is not sufficient for our classification task. In particular, it provides only k coefficients $\mathbf{a}_\mathcal{K} = [a_1, \ldots, a_k]^T$ which makes the calculation of (2.8) impossible. To see this we rewrite the expression in (2.8):

$$g(\mathbf{x}) = V^T \mathbf{a} = \left[V_{\mathcal{K}:}^T,\, V_{(\mathcal{N}-\mathcal{K}):}^T\right] \begin{bmatrix} a_\mathcal{K} \\ a_{(\mathcal{N}-\mathcal{K})} \end{bmatrix} = V_{\mathcal{K}:}^T \mathbf{a}_\mathcal{K} + V_{(\mathcal{N}-\mathcal{K}):}^T \mathbf{a}_{(\mathcal{N}-\mathcal{K})} \ . \tag{2.9}$$

The function g could, in principle, be estimated only according to the truncated coefficient vector (by calculating only the first term in the sum), but by doing so we would very likely be losing valuable discrimination information contained in the last $n - k$ coefficients. While the first k coefficients contain the most of the reconstructive

can be performed in the PCA transformed space and then backprojected to the PCA space to get the image LDA vectors without losing any discriminative information. This automatically implies that the LDA vectors lie in the span of the PCA basis which spans the same linear space as the learning data, which concludes our argument. For the CCA, it was also shown that the CCA vectors lie in the span of the training data [19, 101].

information, there is no guarantee that most of the discriminative information is present in the first k of them as well.

Obviously some extra information needs to be added to the truncated reconstructive basis $U_{:\mathcal{K}}$ to retain the complete discrimination power of g (i.e. also enable the calculation of the second term of the sum in (2.9)). These will be done by augmenting the truncated reconstructive basis with a small number of additional vectors.

Let us define $\tilde{W} := U_{:(\mathcal{N}-\mathcal{K})}V_{(\mathcal{N}-\mathcal{K}):} \in \mathbb{R}^{m\times c}$. The matrix $\tilde{W}$ is composed of c vectors arranged in its columns which are linear combinations of the last $n-k$ basis vectors of a reconstructive model. Each of them is orthogonal to all of the first k vectors of the reconstructive basis, however they are not mutually orthogonal. In order to enable easier calculations later on, the matrix $\tilde{W}$ can be orthogonalized adequately:

$$\tilde{W}_\perp = \tilde{W}(\tilde{W}^T\tilde{W})^{-1/2} \ . \tag{2.10}$$

Next, let us define $\widehat{U}$ and $\widehat{V}$ as:

$$\widehat{U} = \left[U_{:\mathcal{K}}, \tilde{W}_\perp\right] = \left[U_{:\mathcal{K}}, \tilde{W}(\tilde{W}^T\tilde{W})^{-1/2}\right] \in \mathbb{R}^{m\times(k+c)} \tag{2.11}$$

$$\widehat{V} = \begin{bmatrix} V_{\mathcal{K}:} \\ (\tilde{W}^T\tilde{W})^{1/2} \end{bmatrix} \in \mathbb{R}^{(k+c)\times c} \ . \tag{2.12}$$

The new basis $\widehat{U}$ is the basis $U_{:\mathcal{K}}$ extended by $c \ll n$ additional vectors, while $\widehat{V}$ is the matrix $V_{\mathcal{K}:}$ also extended by c row vectors.

Now, it easy to show that $\widehat{U}$ and $\widehat{V}$ contain *all* of the discriminative information contained in W. The new classification function $\widehat{g}(\mathbf{x}) := (\widehat{U}\widehat{V})^T\mathbf{x}$ can be expressed as

$$\begin{aligned}
\widehat{g}(\mathbf{x}) &= (\widehat{U}\widehat{V})^T\mathbf{x} \\
&= \left(\left[U_{:\mathcal{K}}, \tilde{W}(\tilde{W}^T\tilde{W})^{-1/2}\right]\begin{bmatrix} V_{\mathcal{K}:} \\ (\tilde{W}^T\tilde{W})^{1/2} \end{bmatrix}\right)^T\mathbf{x} \\
&= (U_{:\mathcal{K}}V_{\mathcal{K}:} + \tilde{W})^T\mathbf{x} \\
&= (U_{:\mathcal{K}}V_{\mathcal{K}:} + U_{:(\mathcal{N}-\mathcal{K})}V_{(\mathcal{N}-\mathcal{K}):})^T\mathbf{x} \\
&= \left(\left[U_{:\mathcal{K}}, U_{:(\mathcal{N}-\mathcal{K})}\right]\begin{bmatrix} V_{\mathcal{K}:} \\ V_{(\mathcal{N}-\mathcal{K}):} \end{bmatrix}\right)^T\mathbf{x} \\
&= (UV)^T\mathbf{x} = W^T\mathbf{x} = g(\mathbf{x})
\end{aligned} \tag{2.13}$$

and is thus equivalent to the original classification function $g(\mathbf{x})$. This concludes our argument.

Since the new basis $\widehat{U} := [\widehat{\mathbf{u}}_1, \widehat{\mathbf{u}}_2, \ldots, \widehat{\mathbf{u}}_{k+c}]^T$ contains the truncated reconstructive basis $U_{:\mathcal{K}}$ and thus offers a good reconstruction of images, the image $\mathbf{x}$ can be well

approximated[3] in this extended basis:

$$\mathbf{x} \approx \widehat{a}_1\widehat{\mathbf{u}}_1 + \cdots + \widehat{a}_k\widehat{\mathbf{u}}_k + \cdots + \widehat{a}_{k+c}\widehat{\mathbf{u}}_{k+c} := \widehat{U}\widehat{\mathbf{a}}. \tag{2.14}$$

As the matrix $\widehat{U}$ is orthogonal, the coefficients can be obtained in the least square sense as:

$$\widehat{\mathbf{a}} = \widehat{U}^T\mathbf{x} \ . \tag{2.15}$$

But when the image $\mathbf{x}$ is corrupted by outliers, the reconstructive property of $\widehat{U}$ in (2.14) enables us to employ the method of [89] to successfully detect outliers of $\mathbf{x}$ and calculate the coefficient vector $\widehat{\mathbf{a}} = [\widehat{a}_1, \ldots, \widehat{a}_{k+c}]^T$ robustly using only the non-occluded pixels in the image $\mathbf{x}$ as described in Section 2.6.

Since the robust estimation of $\widehat{\mathbf{a}}$ is a good approximation to (2.15) and

$$g(\mathbf{x}) = \widehat{V}^T(\widehat{U}^T\mathbf{x}) = \widehat{V}^T\widehat{\mathbf{a}} \ , \tag{2.16}$$

the classification function g can be calculated as the dot product of the extended matrix $\widehat{V}$ and $\widehat{\mathbf{a}}$. Since the coefficient vector $\widehat{\mathbf{a}}$ can be obtained in a robust manner, the new calculation of g is also robust to outliers and occlusions.

To summarize, we constructed a basis $\widehat{U}$ of $k+c$ vectors (where $k+c \ll n$) which carries the *complete* discriminative information (in the limits of a chosen discriminative method), enables the detection of outliers and occlusions and offers the calculation of the necessary coefficient vector $\widehat{\mathbf{a}}$ on the uncorrupted pixels. Once the coefficients are obtained, the classification function g can be calculated with (2.16) as a dot product of the coefficient vector $\widehat{\mathbf{a}}$ and the extended matrix $\widehat{V}$. We must emphasize that both, $\widehat{U}$ and $\widehat{V}$, are calculated already in the training stage and need no further calculations in the classification stage. These procedures are summarized in Algorithms 1 and 2.

2.6 Robust coefficient estimation

For completeness, we briefly summarize the robust coefficient estimation procedure developed in [89].

Let $\mathbf{x} \in \mathbb{R}^m$ be an image vector containing m pixels and U a basis matrix of a reconstructive model which provides the approximation of $\mathbf{x}$ by its reduced basis:

$$\mathbf{x} \approx a_1\mathbf{u}_1 + \cdots + a_k\mathbf{u}_k,$$

[3]The approximation (2.14) is even more exact than using only k reconstructive basis vectors, since the new basis contains a few more extra vectors (linear combinations of the last $n-k$ reconstructive basis vectors) carrying some additional variance.

Algorithm 1 : Learning phase

Input: $X \in \mathbb{R}^{m \times n}$

Output: $\widehat{U} \in \mathbb{R}^{m \times (k+c)}$, $\widehat{V} \in \mathbb{R}^{(k+c) \times c}$

1: recMethod$(X) \rightarrow U \in \mathbb{R}^{m \times n}, A \in \mathbb{R}^{n \times n}$

2: discMethod$(A) \rightarrow V \in \mathbb{R}^{n \times c}$

3: $\tilde{W} = U_{:(\mathcal{N}-\mathcal{K})} V_{(\mathcal{N}-\mathcal{K}):} \in \mathbb{R}^{m \times c}$

4: $\widehat{U} = \left[U_{:\mathcal{K}}, \tilde{W}(\tilde{W}^T \tilde{W})^{-1/2} \right] \in \mathbb{R}^{m \times (k+c)}$

5: $\widehat{V} = \begin{bmatrix} V_{\mathcal{K}:} \\ (\tilde{W}^T \tilde{W})^{1/2} \end{bmatrix} \in \mathbb{R}^{(k+c) \times c}$

Algorithm 2 : Classification phase

Input: $\mathbf{x} \in \mathbb{R}^m$, $\widehat{U} \in \mathbb{R}^{m \times (k+c)}$, $\widehat{V} \in \mathbb{R}^{(k+c) \times c}$

Output: $g(\mathbf{x}) \in \mathbb{R}^c$

1: Project $\mathbf{x}$ into $\widehat{U}$ in a robust way to obtain $\widehat{\mathbf{a}} \in \mathbb{R}^{k+c}$.

2: $g(\mathbf{x}) = \widehat{V}^T \widehat{\mathbf{a}} \in \mathbb{R}^c$

where the coefficient vector $\mathbf{a}_\mathcal{K} = [a_1, \ldots, a_k]^T$ is usually calculated as $\mathbf{a}_\mathcal{K} = (U_{:\mathcal{K}}^T U_{:\mathcal{K}})^{-1} U_{:\mathcal{K}}^T \mathbf{x}$. The problem appears when $\mathbf{x}$ contains outliers since the product $U_{:\mathcal{K}}^T \mathbf{x}$ takes into account *all* pixels, thereby also the corrupted ones, and can consequently give a wrong value for $\mathbf{a}_\mathcal{K}$. In order to overcome this problem we need to robustly solve the over-determined linear system of equations

$$\begin{aligned} x^1 &= a_1 u_1^1 + a_2 u_2^1 + \cdots + a_k u_k^1 \\ x^2 &= a_1 u_1^2 + a_2 u_2^2 + \cdots + a_k u_k^2 \\ &\vdots \\ x^m &= a_1 u_1^m + a_2 u_2^m + \cdots + a_k u_k^m \end{aligned} \tag{2.17}$$

where x^i denotes the i-th pixel in an *image* vector. Hypothetically speaking, if the approximation of $\mathbf{x}$ would be of zero error, only k equations would be needed to calculate $\mathbf{a}_\mathcal{K}$. But since generally the approximation error is not zero, yet still very small, we could take into account only p, where $k < p << m$, equations from (2.17) to determine the coefficient vector $\mathbf{a}_\mathcal{K}$ to a satisfactory degree of accuracy. This is because the methods that provide good reconstruction of data can exploit the redundancy present in the visual data.

The robust coefficient estimation procedure is based on a hypothesize-and-test paradigm using subsets of image pixels. The basic idea is to randomly choose a set of p pixels $\mathcal{H} \subset \{i \mid i = 1, \ldots, m\}$, $|\mathcal{H}| = p$, in the image $\mathbf{x}$ (each such choice is called a hypothesis), and take into account only these pixels when determining the coefficients.

The task is to find values $\{a_j\}_{j=1}^k$ such that the expression

$$E(\mathcal{H}) = \sum_{i \in \mathcal{H}} \left(x^i - \sum_{j=1}^{k} a_j u_j^i \right)^2$$

is minimal. This is achieved by solving the linear system $G\mathbf{a} = \mathbf{d}$, where G is the $k \times k$ matrix with the entries $g_{ij} = \langle \mathbf{u}_i^{\mathcal{H}}, \mathbf{u}_j^{\mathcal{H}} \rangle$, and $\mathbf{d}$ is the vector with entries $d_j = \langle \mathbf{x}^{\mathcal{H}}, \mathbf{u}_j^{\mathcal{H}} \rangle$. Here, $\mathbf{x}^{\mathcal{H}}$ denotes the pixels in $\mathbf{x}$, which belong to hypothesis $\mathcal{H}$, i.e., $\mathbf{x}^{\mathcal{H}} = \{x^i \mid i \in \mathcal{H}\}$. The obtained coefficients $\{a_j\}_{j=1}^k$ are then used to calculate the reconstruction in the selected pixels $\mathbf{x}^{\mathcal{H}}$. Based on the error distribution in these points, their number is reduced by a factor α (α-trimming), until the maximum error falls below a predefined threshold Θ. This value is determined as twice the average reconstruction error in a single pixel of an image in the training set: $\Theta = \frac{2}{mn} \sum_{i=1}^{n} \|\mathbf{x}_i - \tilde{\mathbf{x}}_i\|^2$. The factor 2 is used because the data $\tilde{\mathbf{x}}_i$ are reconstructions of the training images only and we deal also with images not present in the training set.

To increase the probability of determining correct coefficients, h hypotheses are generated. For each hypothesis, the robust coefficient estimation is performed. Finally the best hypothesis is chosen based on the largest number of compatible pixels in the reconstructions (related to Θ). The remaining pixels (pixels in which the reconstruction error is larger than Θ) are pronounced to be outliers.

2.7 Experimental results

In this section we present experimental results to clearly demonstrate the advantages of the proposed method. We approached four traditional computer vision tasks using subspace methods utilizing the proposed technique for robust estimation of subspace coefficients: *object* and *face recognition*, which are classification tasks and are approached using the extended LDA approach, and *estimation of objects' orientation* and *self-localization of a mobile robot*, which are regression tasks therefore are addressed using the extended CCA approach. In the past a plethora of methods for solving these well known computer vision problems have been proposed; many of them are appearance-based. These approaches very often do not deal with occlusions and similar artifacts in images. In addition they are usually very task specific. Our approach is more general and particularly addresses the problem of robustness. We selected these four problems because they nicely present the sensitivity of the standard subspace approaches to non-gaussian noise (occlusions) and demonstrate the ability of the proposed method to overcome this shortcoming on different image domains. Many of the previously proposed subspace-based methods could be extended with the proposed approach, which would increase their robustness.

2.7.1 Robust estimation of LDA coefficients

Object recognition

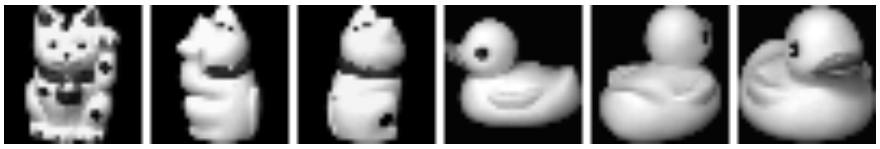
First we demonstrate the performance of the extended LDA approach on a simple two-objects problem. The goal was to find a hyperplane which separates the appearances of two objects. In this experiments we show two major issues that we want to emphasize in this article. Firstly, we show that by performing LDA in a truncated PCA space the results degrade unless this subspace is augmented with the additional discriminative information. And secondly, we show how can the proposed representation, which holds discriminative and reconstructive information, deal with occlusions.

We performed the experiment on two objects from the COIL dataset [106]. In the training stage 12 images of each object were used (some of them are shown in Figure 2.3(a)), while the remaining 60 were used for testing. In the first part of the experiment we used non-occluded test images, while in the second part, when we evaluated the robustness of the proposed approach, we occluded each test image with a square of a random intensity at a randomly chosen position (Figure 2.3(b)).

The results for three different approaches are shown in Table 2.1 and Figure 2.4. In the first row (*LDA*) the results for the standard LDA approach are displayed. The second row (*LDAonK*) shows the results of the estimation of LDA coefficients in the truncated PCA subspace (only the first 12 principal vectors and components, which captured 85% of the overall variance, were retained). Finally, the results in the third row (*LDAaPCA*) were obtained using the proposed method; in this case the 11-dimensional principal subspace (due to centering, 12 training images result into a 11-dimensional space) was augmented with one additional basis vector holding discriminative information contained in the discarded principal vectors (yielding 12-dimensional subspace). The estimation of LDA coefficients was then performed in this augmented subspace. Figure 2.4 presents the results in a graphical form by plotting the values of the LDA coefficient (projection of the input image to the LDA vector) for each test image. The projections of the images of the first object are depicted as squares (empty squares for training images and filled squares for test images), while the projections of the images of the second object are denoted as circles. Here the deviations from the optimal values can be clearly observed. Table 2.1 displays quantitative results – the values of the fisher criterion (Eq. (2.5)) calculated considering test images.

First, all three approaches were applied to the non-occluded test images. The results of this method are presented in the columns indicated with '*ground truth*' in Table 2.1 and Figure 2.4. Since the test images were not occluded, the standard LDA approach performed very well. The projections of training images of two objects are perfectly separated and the generalisation to test images is rather good as well; therefore the

(a)



(b)

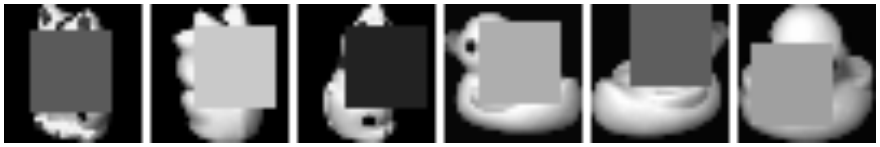


Figure 2.3: (a) A few training images and (b) a few occluded test images of two COIL objects.

recognition rate is 100% and the values of fisher criterion are very high. When the estimation of LDA coefficients was performed in the truncated PCA subspace the projections of training images were not perfect anymore and the value of the Fisher criterion was significantly smaller. This indicates that by truncating the full principal subspace some significant information, which is necessary for the optimal calculation of LDA coefficients, is lost. This is even more evident in Figure 2.5, which plots the results of performing LDA in PCA space of different dimensions. By increasing the subspace dimension (k) we decreased the discarded information and the results of the *PCAonK* approach converged to the optimal ones. The optimal results were also achieved when the truncated principal subspace was extended with the additional vector. The *LDA* and *PCAaLDA* approaches produced equivalent results. Therefore, the appended basis vector captured all the LDA-relevant information, which was contained in the discarded principal vectors.

Then we evaluated the robustness of the proposed approaches on the occluded test images. First we applied the standard non-robust way of calculating LDA coefficients (using dot product) in all three approaches (in Table 2.1 and Figure 2.4 indicated as *non-robust*). The results are rather poor. The occlusions on the test images seriously affected the calculation of the LDA coefficients, thus the recognition of the objects was very unreliable. Then we treated outliers as *missing pixels* and omitted them during the computation of subspace coefficients. In the case of *LDA* approach, the LDA coefficients were computed by solving systems of linear equations (2.17), which arose from non-missing pixels only. In the case of *LDAonK* and *LDAaPCA* the PCA coefficients were computed in this way, and were then projected in the LDA space using the standard approach. The values of the Fisher criteria and the plots of the LDA coefficients show that *LDAonK* and *LDAaPCA* produced more reliable results than *LDA* approach due to higher dimensionality of the PCA subspace. Furthermore, it is also evident that *LDAaPCA* approach outperformed *LDAonK* approach due to richer representation. Finally, we let the methods for robust estimation of subspace coefficients to detect

outliers. Since the outlier detection was based on the reconstruction error (which is shown in the last column of Table 2.1), the *LDA* approach produced inferior results. Linear discriminant vectors span only two dimensional subspace, which does not enable sufficient reconstruction of images and detection of outliers. *LDAonK* and *LDAaPCA* approach yielded significantly better results utilizing reconstructive properties of the PCA method performed on 12 dimensional subspace. Also in this case the best results were obtained using *LDAaPCA* approach.

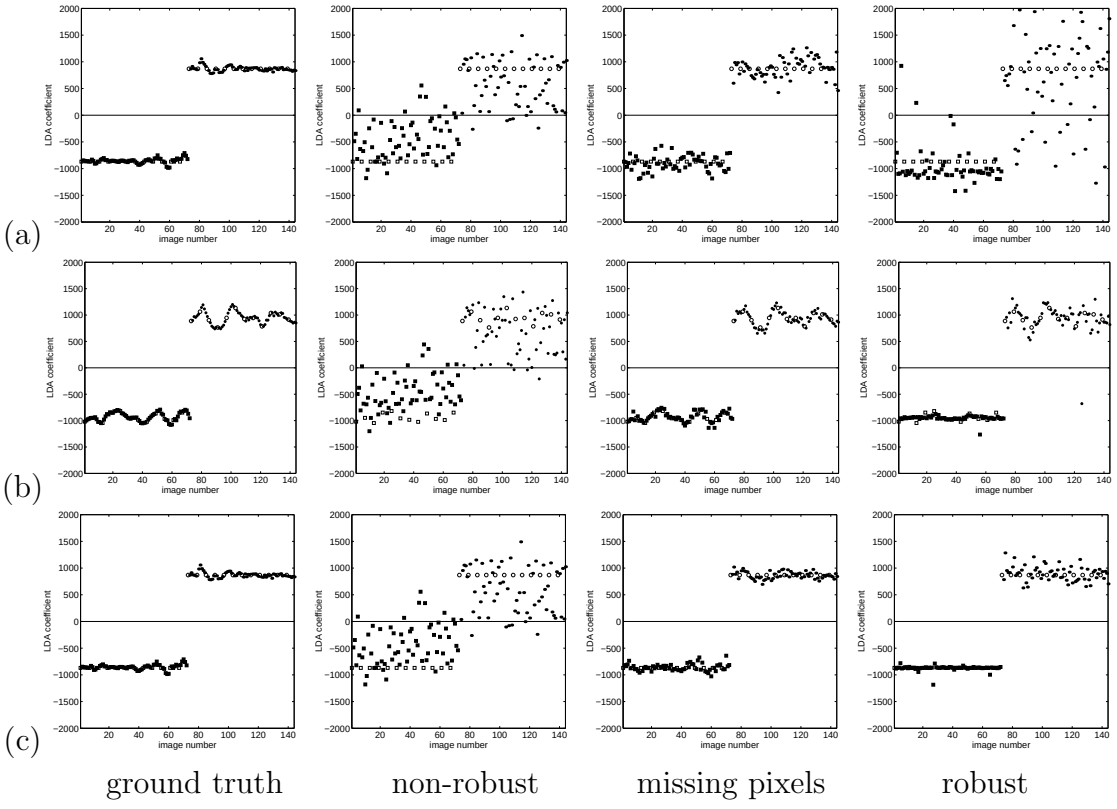


Figure 2.4: Results on two COIL objects. (a) LDA, (b) LDAonK, (c) LDAaPCA.

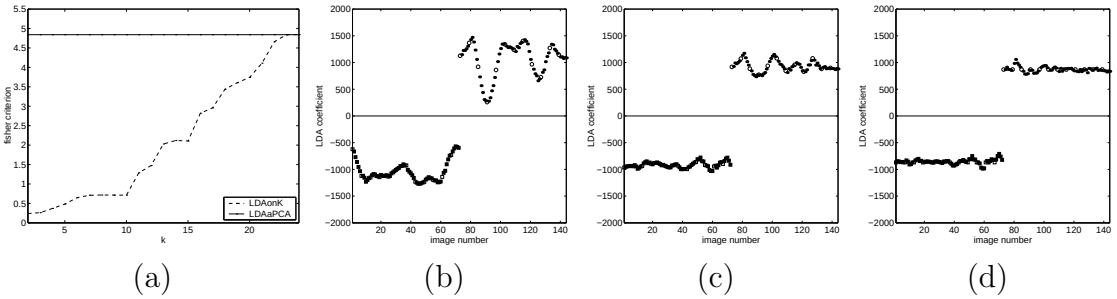


Figure 2.5: Results on ground truth images of two COIL objects: (a) for various k, (b) LDAonK approach for k=3, (c) for k=13, (d) for k=23.

Table 2.1: Results on two COIL objects.

	Fisher criterion				MARE
	ground truth	non-robust	missing pixels	robust	ground truth
LDA	4.84	0.04	0.45	0.03	40.17
LDAonK	1.47	0.05	1.08	0.39	17.29
LDAaPCA	4.84	0.04	2.65	1.06	17.65

Face recognition

We performed face recognition experiments on two testbeds: CMU expression face dataset and ORL face dataset.

The CMU expression face dataset [4] is composed of 75 images of varying facial expressions of each of 13 subjects. The images were collected in the same lighting conditions and have been well-registered by the eyes location. Figure 2.6(a) depicts images of five subjects. In the training stage we used 10 images of "neutral" facial expressions of each subject (see Figure 2.6(b)), while all other images (of various facial expressions) were used in the test stage (Figure 2.6(c)). In this experiment we will show that the proposed robust approach can handle changes in appearance of the faces due to different facial expressions. In addition, we also added a black square of different dimensions to each test image (see Figure 2.6(d)) and tested how severe occlusions can still be handled by the proposed method.

Similar experiments were performed also on the ORL face dataset from Olivetti Research Laboratory in Cambridge, UK [125], where the appearances of individual subjects are more heterogeneous. The dataset contains 10 different images of 40 distinct subjects. For some of the subjects, the images were taken at different times, slightly varying lighting, facial expressions (open/closed eyes, smiling/non-smiling) and facial details (glasses/no-glasses). All the images were taken against a dark homogeneous background and the subjects are in up-right, frontal position (with tolerance for some side movement). One half of the images was used for training (5 images per person, see Figure 2.9(a,b)), while the other half was occluded with varying amount of occlusion (Figure 2.9(c)) and used for testing.

The first row of Figure 2.7 demonstrates the execution of the robust algorithm for estimation of subspace coefficients (see Appendix for details) from the second image in the Figure 2.6(c). At the beginning a number of pixels was randomly chosen (denoted as white points in Figure 2.7(a)). The initial values of subspace coefficients were calculated from these selected pixels. These pixels were then subjected to a few α-trimming

[4] http://amp.ece.cmu.edu/projects/FaceAuthentication/.

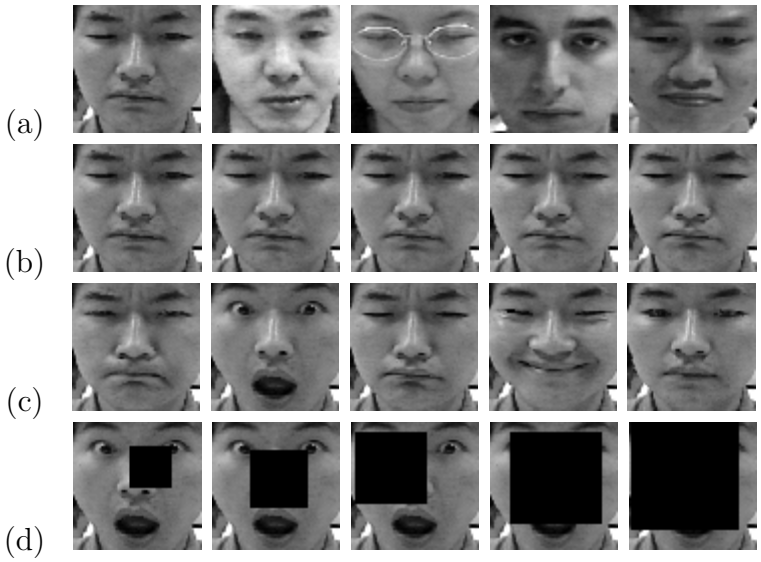
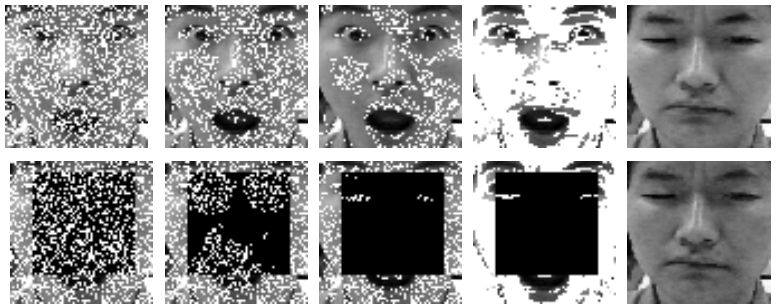


Figure 2.6: Images from CMU expression dataset. (a,b) training images, (c) non-occluded test images, (d) test image occluded with 10%, 20%, 30%, 50% and 70% occlusion.

iterations considering the reconstruction error in these pixels (Figure 2.7(b-c)). Finally, all compatible points were also appended (Figure 2.7(d)). As one can observe, the pixels on the mouth and eyes were discarded, since these regions significantly differ from the appearance in the training images. The subspace coefficients were calculated only considering regions, which were consistent with the appearance of the training images, thus the reconstructed image is very similar to the training images of the (correct) subject (Figure 2.7(e)). The second row of Figure 2.7 demonstrates the execution of the robust algorithm on a test image occluded with 50% of occlusion. As one can observe, in the α-trimming phase almost all the points in the occluded region were eliminated, and then they were not included in the set of compatible points. Therefore, the subspace coefficients were calculated only from "good" pixels, which resulted in a good reconstruction (the occlusion completely disappeared).

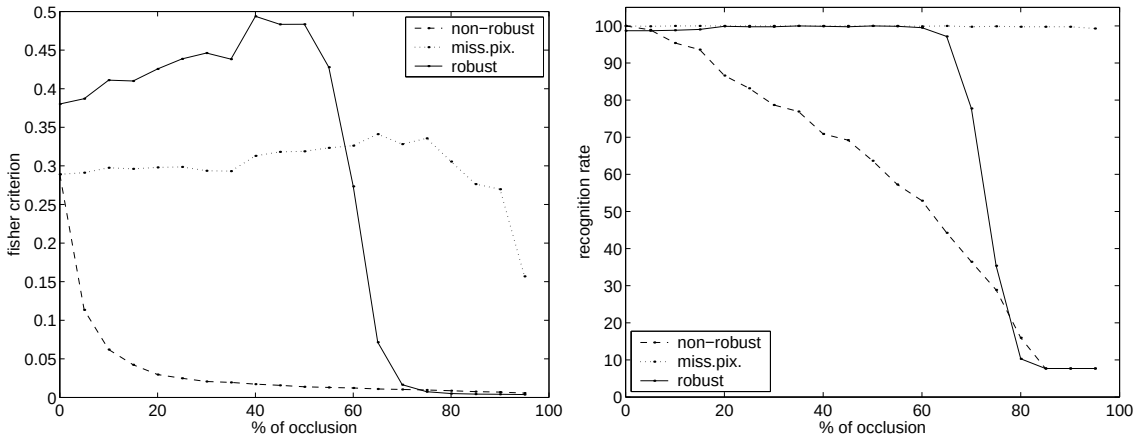
Figure 2.8 shows the results obtained by performing LDA classification on the entire test set of 845 CMU images occluded with different amount of occlusions (0-95%) in three different ways: using the standard non-robust approach (indicated as *non-robust*), and by the proposed robust approach with known (*miss.pix.*), and unknown (*robust*) positions of outliers (black squares). One can observe that the standard non-robust approach was considerably affected by the occlusions and its efficiency rapidly decreased with the increase of the percentage of outliers. In contrast, the proposed robust method performed very well. When the positions of the outliers were known, the non-missing pixels contained information sufficient for reliable discrimination between 13 subjects



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

Figure 2.7: Robust estimation of coefficients. (a) Initially selected points, (b,c) points selected after first and third alpha-trimming iteration, (d) all compatible points, (e) reconstructed image.

for almost all levels of occlusion. Even when the robust method had to automatically detect outliers, the results were degraded only after the occlusion reached more than 60%, which demonstrates the high break-down point of the proposed method.



(a) (b)

Figure 2.8: Results on CMU expression face dataset with test images containing different amounts of occlusion: (a) Fisher criterion, (b) recognition rate.

Similar results we also obtained on the ORL face dataset. Due to a higher number of subjects and larger variability of their appearances the results were in general a little inferior to the results obtained on the CMU dataset, however the comparison between the non-robust and the robust method yields similar conclusions. The proposed method performed significantly better. Its performance dropped down considerably only after 50% of occlusion was reached, as can be seen in Figure 2.10. In this first part of the experiment five dimensional principal subspace ($k = 5$) was augmented with additional

basis vectors. Five principal vectors sufficiently increased the reconstructive power of the method to enable a reliable detection of outliers.

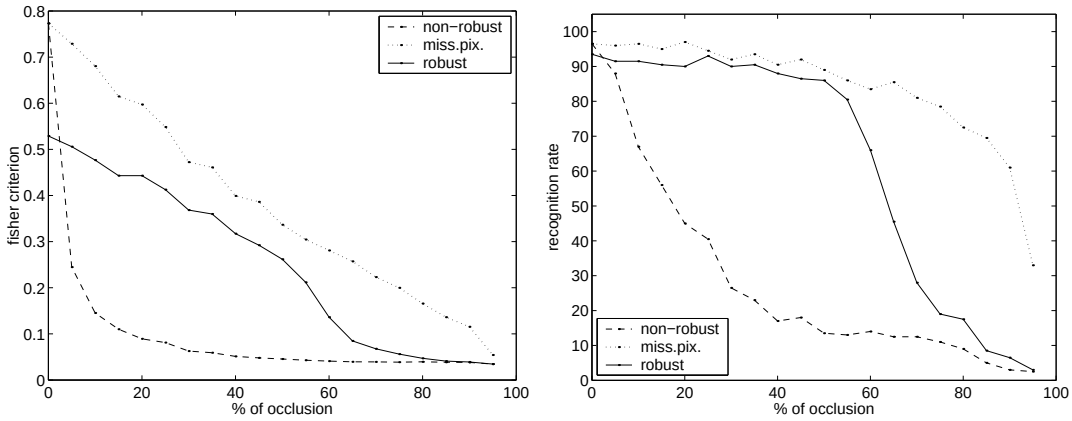
In the second part of the experiment we tested how the different dimensionalities of the basic PCA subspace influence the results. The results are depicted in Figure 2.11. The values of Fisher criteria and recognition rates obtained on the test images occluded with 10% (filled circles and dots) and 50% (empty circles) of occlusions are presented for known (dotted line) and unknown (solid line) positions of outliers. As expected, the non-robust method (indicated as *NR*) did not perform well. When we applied the method for robust estimation of coefficients in the subspace spanned by the LDA vectors only ($k = 0$), the results improved, but they were still significantly inferior to the results of the proposed method. By having also the reconstructive basis (augmented with additional vectors) the reconstructive power of the method increased and the detection of the outliers became more reliable. The results were better for all values of k, but with the increase of k the results improved even further. However, a satisfying level of the evaluation criteria was achieved already when only a few (i.e., 5) principal vectors were used.

Figure 2.9: Images from ORL face dataset. (a,b) training images, (c) test image occluded with 10%, 20%, 30%, 50% and 70% occlusion.

2.7.2 Robust estimation of CCA coefficients

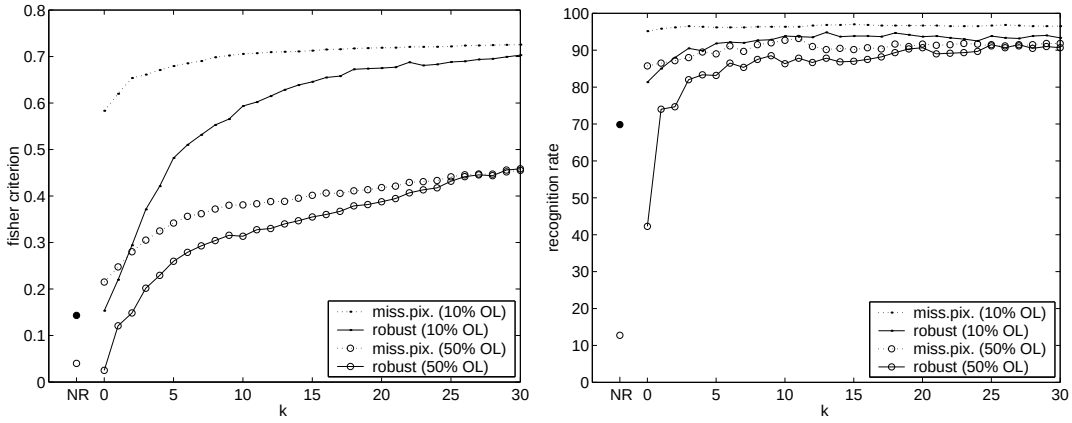
Estimation of orientation

The experiment on orientation estimation was performed on a set of 120 images of a toy fish, which were taken from the views evenly distributed around the object (see Figure 2.12(a)). The goal was to learn the relation between the appearances of the object and their orientations using CCA in the training stage and then to use this



(a) (b)

Figure 2.10: Results on ORL face dataset with test images containing different amounts of occlusion and $k = 5$: (a) Fisher criterion, (b) recognition rate.



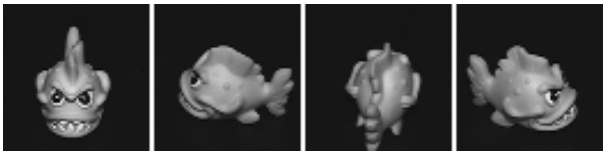
(a) (b)

Figure 2.11: Results on ORL face dataset with test images containing 10% and 50% of occlusion for standard LDA approach (NR) and for proposed method with different k: (a) Fisher criterion, (b) recognition rate.

knowledge to estimate the orientation of the object in a novel image in the test stage. Every fourth image was used in the training stage, while the remaining 90 images were used for testing.

For each training image its orientation (two-dimensional vector indicating the direction from which the image was taken - sine and cosine of the angle) was known. We estimated a linear mapping from the two-dimensional vectors of canonical correlation coefficients to orientation vectors using the least squares minimization method. This mapping function was then used to estimate the orientations of the test images from their canonical correlation coefficients.

(a)



(b)

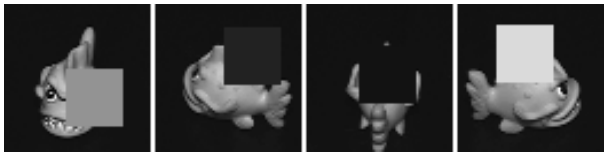


Figure 2.12: (a) Four non-occluded images, and (b) four occluded test images of toy fish used in the orientation estimation CCA experiment.

First we used non-occluded images also in the test stage. The results are shown in Figure 2.13(a) and Table 2.2(a). The plots show the actual orientations (x axis) and the estimated orientations (y axis) of the object in the training images (squares) and test images (circles). The results were rather poor when we used only the truncated vectors of principal components (e.g., for k=3 or k=8). By increasing the number of preserved principal vectors, the results converged to the optimal result achieved either by using all 30 principal components or augmented truncated PCA basis (denoted by $APCA$) or by performing CCA on the original images without the PCA preprocessing.

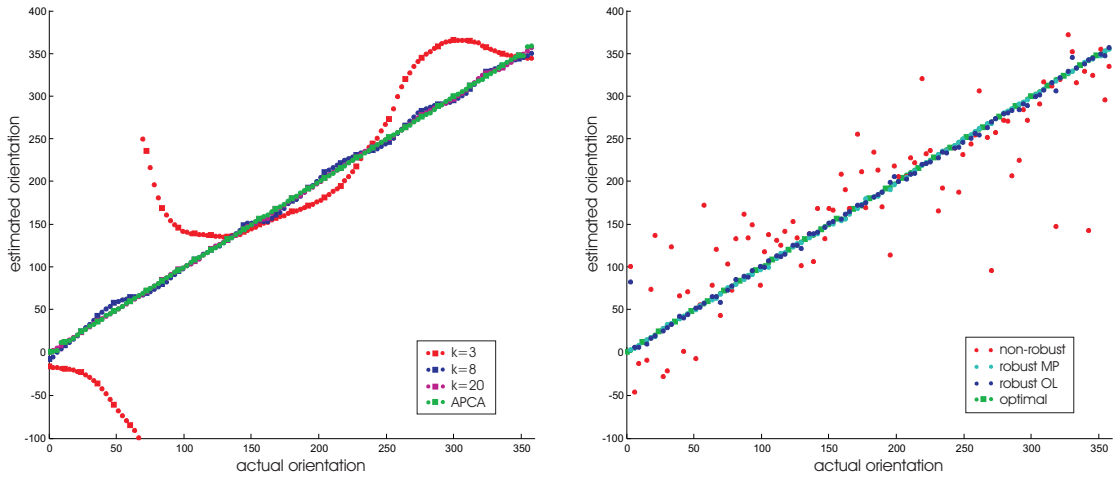
Then we added a square of a random intensity to a randomly chosen position in each test image (Figure 2.12(b)). The results are presented in Figure 2.13(b). When we used a standard non-robust method the obtained projections of occluded test images were affected by the outlying pixels, thus the estimates are very inaccurate (denoted by *non-robust*).

We then applied the PCA preprocessing step and used the robust method for estimation of coefficients. First we assumed that the positions of outliers were known; the outliers were considered as missing pixels and principal coefficients were estimated from inliers only ($robustMP$). The results are very close to the optimal ones.

Then we considered the test images without any additional information about outliers and applied the proposed robust method ($robustOL$). The outliers did not affect the estimation of the principal coefficients considerably, thus they did not have a significant influence on the simple projection in the regression stage either. Therefore, the results of the pose estimation are good; the deviations from the optimal estimates are mostly rather small.

Table 2.2(b) presents mean absolute orientation errors in degrees for different dimensions of the principal subspace for the cases where the positions of outliers were assumed to be known (MP) and not known (OL). One can observe that the errors in the first case are very small even when a small number of principal components were

used. As expected, the results were slightly worse when the outliers were not known, since the robust procedure first had to detect outliers. In this case a higher number of principal components should be used, since the top few principal components do not contain enough information for a reliable detection of outliers. However, these results are still significantly better than the results of the standard non-robust method.



(a) (b)

Figure 2.13: Estimation of orientation: results on (a) non-occluded images for different k, (b) occluded test images.

Table 2.2: Mean absolute orientation errors: (a) non-occluded, (b) occluded test images.

k	error
3	42.08
5	5.17
8	3.61
10	2.25
15	1.25
20	0.97
APCA	0.67

(a)

k	MP	OL
3+2	2.62	20.33
5+2	2.33	14.71
8+2	1.72	6.81
10+2	1.56	4.51
15+2	1.22	3.40
20+2	1.14	2.56
non-robust		36.92

(b)

Mobile robot localization

CCA-based localization relates appearances of the environment taken from different locations with location vectors and uses this relation for estimating the current location

of the robot. Thus, in the *learning stage*, we acquired several panoramic images from different positions, which together formed a good depiction of the environment. For each training image the corresponding position was known (i.e., the two-dimensional vector indicating two offsets from the starting position). Then the CCA was applied to these two sets of training data. As a result the canonical correlation vectors of both input sets were obtained. We then projected all training images onto their CCA vectors and obtained vectors of canonical correlation coefficients. Finally we estimated a linear mapping from these two-dimensional CCA coefficient vectors to the corresponding location vectors using the least squares minimization method. Later, in the *localization stage*, we estimated the location of the robot simply by projecting a new panoramic image onto the canonical correlation vectors and mapped the obtained canonical correlation coefficients using the mapping function estimated in the learning stage [136].

The results are shown in Figs. 2.15(a-d), which depict an augmented map of the lab, where the experiment was performed. The training images were taken from the positions marked with squares (62 training panoramic images taken 60 cm apart unwarped and resized to 26×50 pixels; some of them are shown in Figure 2.14(a)). The actual path of the robot in the test stage (ground truth) is indicated with gray filled circles (100 test images), while the estimated path is marked with black filled circles and thin solid lines. The error at each test location is depicted as a dashed line. The quantitative results in terms of mean localization error (the average distance between the actual and the estimated location) are also presented in Table 2.3.

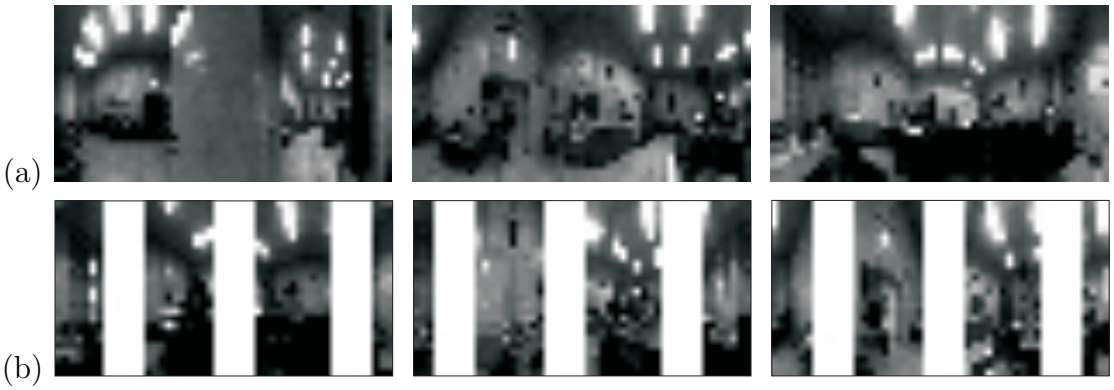


Figure 2.14: Mobile robot localization: (a) three training panoramic images, (b) three occluded test images.

First we applied the described approach using the standard CCA method on the non-occluded test images. The results are depicted in Figure 2.15(a). The mean localization error was 34.8 cm as shown in the column indicated with *ground truth* in Table 2.3. This is the best result the standard CCA without nonlinear extensions ([101, 136]) can achieve.

Then we occluded test images with vertical bars (see Figure 2.14(b)). The results (indicated as *non-robust*) have been significantly degraded; the localization errors are large (118 cm in average), as can also be seen in Figure 2.15(b). Then we excluded the vertical bars from the computation. The results have significantly improved (indicated as *missing pixels* in Table 2.3, see also Figure 2.15(c)). The proposed approach (*CCAaPCA*) yielded the best results (42.9 cm).

Finally we applied a complete robust method for estimation of subspace coefficients, which also had to detect and then discard occlusions (vertical bars). In the case of *CCA* approach only two-dimensional coefficient vectors did not hold sufficient information needed for reconstruction, thus the outliers could not be detected, thus this method completely failed. In the case of *CCAonK* approach 14-dimensional truncated principal subspace enabled better reconstruction and outlier detection, thus the localization has improved considerably. By augmenting the 12-dimensional truncated principal subspace with additional two basis vectors, which hold CCA-relevant information as well (thus having 14-dimensional augmented subspace), the results have improved even further (see Figure 2.15(d) and *robust* column in Table 2.3). This experiment thus also demonstrates that the proposed method outperforms the standard one.

Table 2.3: Mean localization error in cm.

	ground truth	non-robust	missing pixels	robust
CCA	34.8	118.0	63.2	236.6
CCAonK	44.6	112.0	48.0	55.7
CCAaPCA	34.8	118.0	42.9	49.6

2.8 Summary and Conclusions

The importance of the discriminative methods has often been emphasized in the literature for their strong ability of classification/recognition, which is one of the main tasks of computer vision. However, the fact that they cannot successfully cope with outliers and occlusions that commonly appear in real-world settings has severely limited their domain of applicability. The concept of robust classification appears to be an elusive task and thus has rarely been tackled in the literature.

In this chapter we proposed a method that is novel in the area of robust classification/regression. It combines the properties of both discriminative and reconstructive methods, preserving the classification power from the former and enabling robust behavior stemming from the latter. The robust approach exploits several techniques, i.e.,

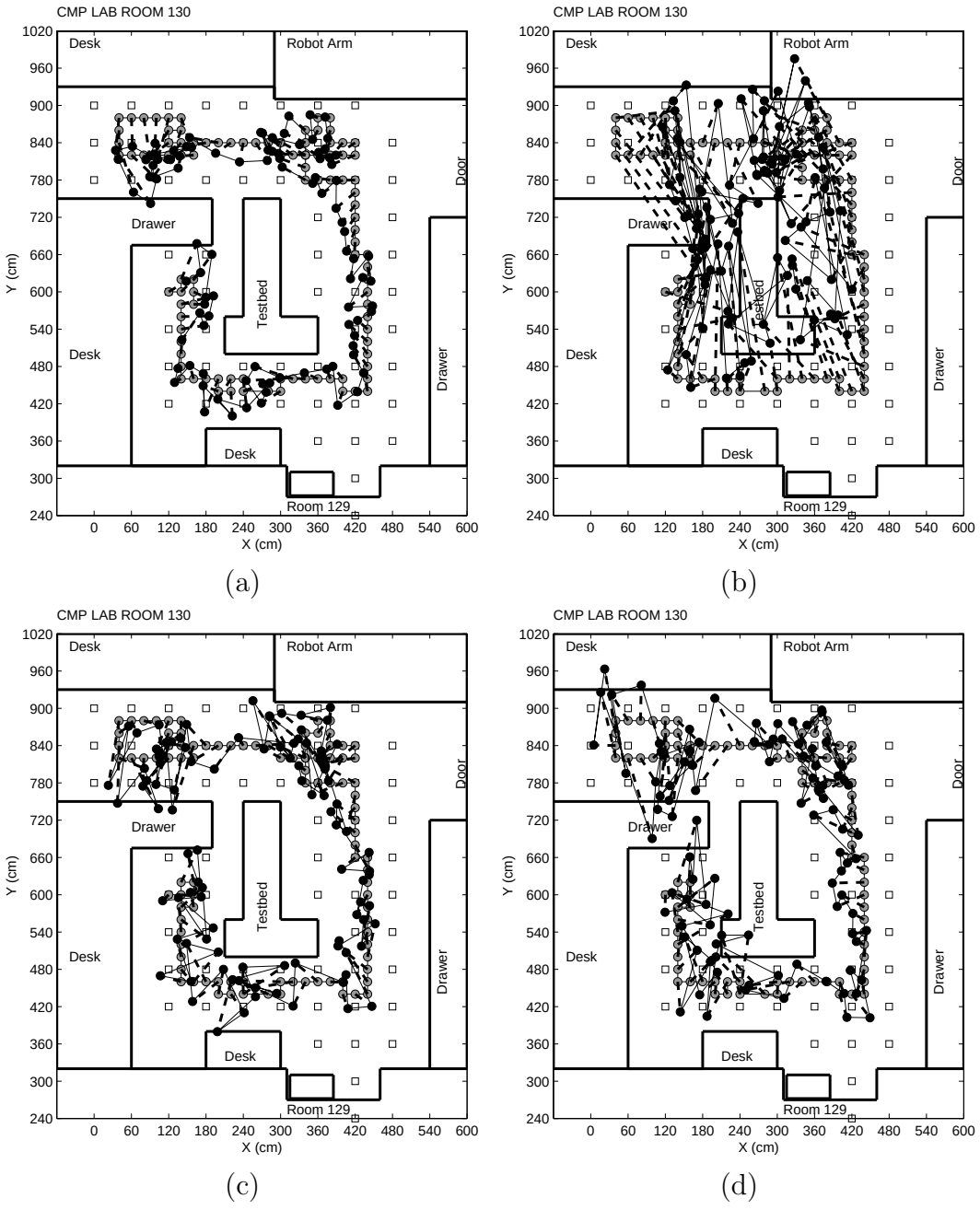


Figure 2.15: Mobile robot localization results: (a) CCA on non-occluded images (*ground truth*), (b) CCA on occluded test images (*non-robust*), (c) CCAaPCA considering *missing pixels*, (d) *robust* CCAaPCA.

robust estimation and the hypothesize-and-test paradigm, which, combined together in a general framework, achieve the goal. We evaluated the theoretical results on several computer vision tasks showing that the proposed method significantly outperforms the standard discriminative methods. A general conclusion drawn from these experiments

is that our robust method can tolerate much higher levels of outliers (occlusions) than the standard discriminative methods.

The applications of the proposed method are numerous. All tasks that can be accomplished by the classical discriminative methods can also be achieved within the framework of our proposed approach, only in a more robust way.

There is, however, an inherent limitation of the subspace appearance models for the general task of object class recognition and, especially, object detection. Firstly, these approaches (as applied in this thesis — working directly with pixel values) do not capture the shape of the objects but only the appearances (i.e. the intensity pixel values) of the objects. This makes them more suitable for recognizing specific objects such as faces and much less applicable to the general problem of visual categorization. More importantly, if these models were to be used for object detection, we would need to employ the *sliding window* approach (applying the classification model in each possible subwindow in an image) which is a computationally very inefficient way of searching for an object in images. Although, speed-ups of the sliding window approaches have recently be proposed by the branch and bound technique [85].

In the following chapters we present a more general approach to object class recognition and detection.

Chapter 3

Hierarchical Compositional Approach: Motivation

Figure 3.1: Examples of object categories. There are around $40,000$ visual object categories estimated to exist in the world [16]. There is significant visual variability of objects within each class and across objects classes. Each object can appear in images in many 3D poses, articulations, under various illumination conditions and can be occluded.

Visual object class recognition has been an area of active research in the vision community for decades [32]. Ultimately, the goal is to recognize and detect an increasing number of object classes in images within an acceptable time frame. The problem entangles three highly interconnected issues: the internal object **representation** which should compactly capture the high visual variability of objects and generalize well over each class, means of **learning** the representation from a set of images with as little supervision as possible, and an effective **inference** algorithm that robustly matches

the object representation against the image.

The increased popularity of the problem witnessed in the recent years and the advent of powerful computer hardware have led to a seeming success of categorization approaches on the standard datasets such as Caltech 101 [41]. However, the high discrepancy between the accuracy of object classification and detection/segmentation [39] suggests that the problem still poses a significant and open challenge. The recent preoccupation with tuning the approaches to specific datasets might have precluded the attention from the most crucial issue: the representation [116].

Given images of complex scenes, objects must be inferred from the pixel information through some recognition process. This requires an efficient and robust matching of the internal object representation against the representation produced from the scene. Despite the seemingly effortless performance of human perception, the diversity and the shear number of visual object classes (Figure 3.1) appearing in various scales, 3D positions and articulations, which additionally interact with each other (occlusion, clutter, etc.), have placed a great obstacle to the task. In fact, it has been shown by Tsotsos in 1990 [149] that the unbounded visual search is NP complete and thus approximate, hierarchical solutions might be the most promising/plausible way to tackle the problem. This line of architecture is also consistent with the findings on biological systems [122, 27]. A number of authors have further emphasized these computational considerations [38, 66, 6, 131, 72], suggesting that matching should be performed at multiple hierarchical stages, in order to gradually and coherently limit the otherwise computationally prohibitive search space [38, 149, 23, 7, 100, 6, 66, 47, 131, 55]. While hierarchies presented a natural way to represent objects in the early vision works [38, 67, 99, 33], surprisingly, they have not become an integral part of the modern vision approaches.

Using vocabularies of visual features has been a popular choice of object class representation and has yielded some of the most successful performances for object detection to date [44, 146, 111, 133, 46, 88]. However, the majority of these works are currently using flat coding schemes where each object is represented with either no structure at all by using a bag-of-words model or only simple geometry induced over a set of intermediately complex object parts. In this paper, our aim is to model *hierarchical compositional structure* of the objects and do so for multiple object classes.

Modeling structure (geometry of objects) is important for several reasons. Firstly, since objects within a class have usually distinctive and similar shape, it allows for an efficient shape parametrization with good generalization capabilities. It further enables us to *parse* objects into meaningful components which is of particular importance in robotic applications where the task is not only to detect objects but also to execute higher-level cognitive tasks (manipulations, grasping, etc). Thirdly, by inducing the structure over the features the representation becomes more robust to background

clutter.

Hierarchies incorporate structural dependencies among the features at multiple levels: objects are defined in terms of parts, which are further composed from a set of simpler constituents, etc [38, 15, 61, 6, 70, 21, 152, 166, 55, 109]. Such architectures allow sharing of features between the visually similar as well as dissimilar classes at multiple levels of specificity [149, 6, 109, 55]. If we are to model for example cars and people, we can still re-use a number of features of smaller granularity and compress the joint representation. Sharing of features means sharing common computations and increasing the speed of the joint detector [146]. More importantly, shared features lead to better generalization [146] and can play an important role of regularization in learning of novel classes with few training examples. Furthermore, since each feature in the hierarchy recursively models certain variance over its parts, it captures a high structural variability and consequently a smaller number of features are needed to represent each class. A number of hierarchical recognition systems have been proposed and confirmed the success of such representations in object categorization tasks [64, 7, 61, 140, 127, 131, 153, 102, 107, 146, 130, 119, 109, 145].

It must be emphasized, however, that hierarchical representations do not necessarily imply computational efficiency and representational plausibility. In this thesis, we will argue for a special form of a multilayered architecture — a *compositional hierarchy*. The nodes in such a hierarchy are formed as compositions that, recursively, model loose spatial relationships between their constituent components. The abundant computational arguments accumulated throughout the history of computational vision speak in favor of its efficiency, robustness to clutter, and flexibility to capture structural variability. While the classical neural networks have been commonly thought to be a faithful model of the hierarchical processing in the brain, interestingly, ideas of compositional units have also started to emerge in the neuroscience community [9, 27, 150].

There have been a number of attempts at compositional categorical representations [38, 33, 66, 23, 7, 6], however, the lack of automation might have been a major contributing factor that prevented better realizations of these ideas. By learning, we minimize the amount of time consuming human involvement in object labeling [166, 91] and avoid bias of pre-determined grouping rules or manually crafted features [131, 126, 66, 129]. Second, learning the representation statistically yields features most shareable between the classes, which may not be well predicted by human labelers [36]. However, the complexity of learning a hierarchical representation bottom-up and without supervision is immense: there is a huge number of possible feature combinations, the number of which exponentially increases with each additional layer — thus an effective learning algorithm must be employed.

In this thesis the idea is to represent the objects with a *learned hierarchical compositional shape vocabulary* that has the following architecture. The vocabulary at each

layer contains a set of hierarchical deformable models which we will call *compositions*. Each composition is defined recursively: it is a hierarchical *generative* probabilistic model that represents a geometric configuration of a small number of parts which are themselves hierarchical deformable models, i.e., compositions from a previous layer of the vocabulary. We present a framework for *learning* such a representation for multiple object classes. Learning is statistical and is performed bottom-up. The approach takes simple oriented contour fragments and learns their frequent spatial configurations. These are recursively combined into increasingly more complex and class-specific shape compositions, each exerting a high degree of shape variability. In the top-level of the vocabulary, the compositions are sufficiently large and complex to represent the whole shapes of the objects. We learn the vocabulary layer after layer, by gradually increasing the size of the window of analysis and the spatial resolution at which the shape configurations are learned. The lower layers are learned jointly on images of all classes, whereas the higher layers of the vocabulary are learned incrementally, by presenting the algorithm with one object class after another. We assume supervision in terms of a positive and a validation set of class images — however, the structure of the vocabulary is learned in an *unsupervised* manner. That is, the number of compositions at each layer, the number of parts for each of the compositions along with the distribution parameters are inferred from the data without supervision.

In the remainder of this chapter we argue about the plausibility of the principles employed, namely, the benefits of using a *hierarchical compositional representation* versus other hierarchical architectures, and *unsupervised learning* versus supervised learning approaches.

3.1 Hierarchical Compositionality

Compositionality refers to a property of hierarchical representational systems that define their internal nodes in terms of simpler constituent components according to a set of production rules [15]. The rules of composition usually take the form of the Gestalt laws of grouping [108, 46] or similar forms of predefined bindings [38, 7, 131, 145, 166] that in some form or another incorporate spatial relations into the compositional features. Computational benefits of compositionality in terms of storage, processing demands, robustness to clutter and the exponential expressive power have long been emphasized in the computer vision literature [14, 7, 66, 165, 166, 55, 23]. We substantiate these issue below.

Storage demands. In the current state-of-the-art flat representations millions of distinctive image patches (with dimensions ranging around 25×25 pixels) or local descriptors such as SIFT or HoG must be stored to produce good recognition results.

In the classical hierarchies such as neural networks the number of necessary features to warrant competitive performance is grantedly significantly lower (the number ranges from $10 - 20$ in the lowest layer and increases to the order of a few thousand in the top-most layer), however, each hierarchical unit still must encode weights to *all* feature types from a layer below covering a certain spatial neighborhood. Conversely, as the complexity and size or representation also grows with the number of layers in compositional hierarchies, each higher-level composition encodes only pointers to a small number of its constituent parts and a modest amount of additional information binding the parts spatially. Furthermore, since all the higher-level compositions are constructed from a smaller common vocabulary from a layer below, it is easier to compare and generalize between them. Consequently, extending the hierarchical vocabularies to novel compositions can operate in a more controlled manner leading to more compact and parsimonious hierarchical representations.

Processing complexity. Since each hierarchical unit is shared among many more complex higher layer compositions, most of the computations performed during an online recognition stage are inherently common and can thus be only performed once. Such sharing of computations greatly reduces the computational cost of matching with respect to searching for each complex interpretation in isolation. Moreover, as processing of images is done by sequential (hierarchical) testing of compositional hypotheses, recognition towards the final categorical nodes proceeds in a more controlled and fast manner by pruning the object hypothesis space along the hierarchical path.

Robustness to clutter, repeatability of detection. Each hierarchical node makes inference over a certain size of a local neighborhood, usually referred to as its *receptive field*. The level of hierarchy brings about larger and larger portions of an image that the nodes "cover" and which are likely to contain many structures pertaining to different objects in the scene or rarer structures that the hierarchical units are not essentially tuned to. The classical neural networks that define the units as some non-linear function of an integrative weighted sum over its entire receptive field both spatially as well as in all constituent feature types (schematically depicted in Figure 3.2) are inherently prone to error since the signal coming from multiple objects is essentially mixed. This can be alleviated by enforcing sparsity on the feature weights to enable a focus on only particular substructures of receptive fields. In turn, compositions are inherently sparse — they are designed to respond to only small spatial subsets of their receptive fields in which the presence of only a few feature types is accounted for (depicted in Figure 3.2). This ensures that clutter has little effect on the activity within the hierarchical recognition process and additionally permits faster processing over the traditional neural networks approaches.

Expressive power. Even a small number of feature types defining the outset of the hierarchy can construct a large number of possible combinations, which becomes even

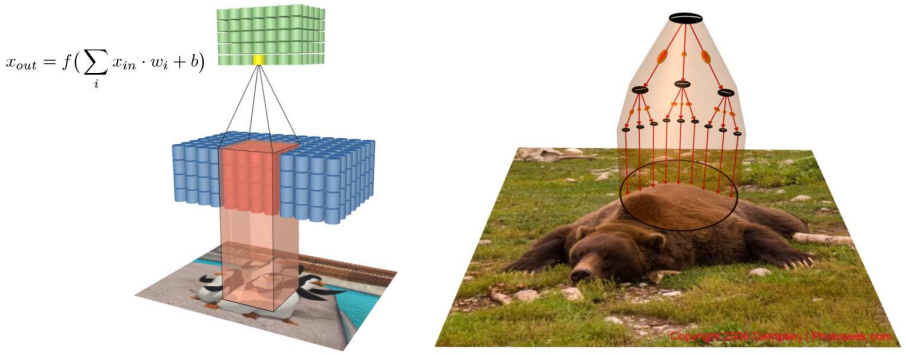
more pronounced with the level of hierarchy. Importantly, as the vocabulary is expected to grow with exponential tendency as new layers (compositions of compositions that essentially should converge to objects themselves) are added and the complexity as well as distinctiveness of the representation increase, the principle of indexing ensures tractability of the recognition process.

Feedforward and feedback. There has been a long-standing debate about what can or cannot be achieved in a strictly feedforward manner in vision in general and in hierarchical categorization approaches in particular [144, 76, 87, 148]. There is neurophysiological evidence proving good categorization performance in the first feedforward pass by humans [144, 155], while many authors emphasize the importance of both, feedforward and feedback and the iterative process between the two [76, 87, 152, 148]. The ability to traverse back from the final recognition nodes inferred from the scene back to the original pixels that produced the high-level decision is important for segmentation as well as looping between bottom-up and top-down inference on ambiguous visual input. This kind of reciprocal inference presents an impediment for the neural network approaches [74] and their closely related architectures [121] since the firing response of a hierarchical unit is too reductive. The information from a cube-like receptive field over lower-layer feature responses is conveyed in only one value — the weighted sum. This makes it difficult to determine and trace back what has in fact caused the response (depicted in Figure 3.2) while also making inference less controlled and reliable. Conversely, in compositional architectures the representation inferred from the visual scene is essentially a graph in which each node has only a small number of incident descendants. Such a representation inherently allows for iterative loops between the data (image) and high-level inferences, whereby the segmentation of objects is simply an inverse process of recognition.

A part of the biological evidence could potentially support such a line of compositional architecture [112, 150, 78, 9, 27]. Additionally, attempts have been made to map the mathematical theory of compositionality onto the neuronal structure of the visual cortex [15].

3.2 Statistical, bottom-up learning

The appealing properties of compositional hierarchies and their advantages over the related hierarchical architectures might prove them a suitable form of representing visual information. However, while learning presents an integral part of the neural networks approaches, most compositional approaches have been hindered by the use of predetermined sets of features or grouping rules. Here, we argue for the importance of learning, specifically, we emphasize the critical role of unsupervised, bottom-up



(a) (b)

Figure 3.2: (a) Neural networks, (b) Compositional hierarchy.

learning.

Bottom-up learning. There seems to be a consensus that the higher-level concepts such as selectivity to object categories are learned since, evidently, a genetic predisposition towards e.g. mobile phones and similar ever-evolving technological gadgets would seem far-fetched. Interestingly, there are opposing views on whether the tunings in the early cortical areas are learned or hard-wired by evolution. The diverse physiology underlying different brain areas suggests specific functionalities and computations performed. This striking systematicity surely is a result of evolution and it undoubtedly guides and controls what the cells can or cannot become tuned to. However, it is highly improbable that all the low-level sensitivities are instilled genetically — the brain must, after all, adjust its perceptual functioning with respect to its sensory receptors and the input it receives.

Computationally speaking, the categorical representations are built upon a set of features that must at some point operate on the image data. The design of these features (for example T- and L-junctions, etc) should not rely on our intuition but rather be learned from the data in order to conform well to the local structures of images. The features/models in the lowest level of the hierarchy should thus be brought down close to the images by performing simple operations with little semantic value. The subsequent learning should then be designed in order to statistically build more complex and semantic models in composition.

Once the *visual building blocks* are learned, learning of objects becomes tractable since only a small number of descriptive structural features are needed to explain them away. Thus, categorical learning can proceed mainly in the higher hierarchical layers and can thus operate fast and with no or minimal human supervision.

Unsupervised learning. Features and their higher level combinations should be learned in an unsupervised manner (at least in the first stages of the hierarchy) in

order to avoid hand-labeling of massive image data as well as to capture the regularities within the visual data as effectively and compactly as possible [10, 122, 35, 70, 47, 55]. Moreover, there are strong implications that the human visual system is driven by these principles as well [59].

By learning the compositional binding priors the representation becomes adjustable to the structural variability of objects. Consequently, it enables a computationally feasible recognition process where the majority of the exponential number of possible compositional groupings are made unimportant (i.e. unrepeatable) by the statistics of natural images.

Incremental learning. Desirably, the hierarchical vocabulary should be extended incrementally as new images/objects are seen by the system. Performed in this way, we avoid batch processing of masses of images (which likely might not even be possible), while on the other hand, we ensure the representation is open to continuous adaptation of the visual environment.

The issue of incrementality in hierarchical architectures is not completely apparent. If features are changed, removed or added at any layer exclusive of the top-most one, all the features on the layers above must be adjusted accordingly. This problem is particularly evident in the neural network type of hierarchies where adding one feature results in the inefficient restructuring of the weights of the complete representation. In compositional hierarchies this problem concerns only a small subset of higher-level features that compositionally emerge from the point of change. Furthermore, by learning the representation sequentially, i.e. optimally adjusting layer after layer to the regularities present in the natural signals, we guarantee that very little encoded information (if something at all) will need to be re-adapted in the deepest layers of the hierarchy as new data is encountered.

3.3 Outline of the approach

The remainder of the thesis is organized as follows. In Chapter 4 we review the related work. Section 5.1 presents our hierarchical compositional representation of object shape with recognition and detection described in Section 5.2. In Section 5.3 our learning framework is proposed. The experimental results are presented in Chapter 6. The thesis concludes with a summary and discussion in Chapter 7.

Chapter 4

Related Work: Hierarchical Categorization Models

This chapter reviews the related literature on the topic of object class recognition with the focus on hierarchical approaches. In Section 4.1 we first review a selected subset of part-based representations, focusing on those that use shape as the primary cue for object representation. The following sections are devoted to hierarchical models. In Section 4.2 we review some of the most successful neural network approaches. Section 4.3 describes hierarchical classification approaches. In Section 4.4 we review the literature on hierarchical compositional approaches to object class recognition. This Section is split into two parts which categorize these works by the amount of supervision needed to train the models.

4.1 Part-based approaches

The seminal work on part-based representations dates back to the work by Fischler and Eschlager [58] who defined objects in terms of a small number of semantic parts and loose geometric springs between them. This work motivated several recent approaches which introduced powerful supervised learning algorithms to train the models from annotated examples [134, 77].

A number of part models have been proposed for object class recognition and it is out of scope of this thesis to review them here. We focus on those that use contour information in order to represent the objects.

Contour fragments have been employed in the early work by Selinger and Nelson [105] and a number of follow-up works have used a similar approach. Opelt et al. [111] extracted longer *boundary fragments* which were used in a Hough voting scheme (depicted in Figure 4.1). The fragments were learned in a boosting framework which

optimized sharing of features between multiple object classes. The framework additionally allows for incremental learning of object class representations and a sublinear growth in the number of features with respect to the number of classes was reported. Such scaling properties were previously reported in the works by Krempp et al. [84] and Torralba et al. [146]. The main problem that we see with [111] is in extraction of the long boundary fragments, for which long chains of connected edge pixels must be tracked in an image. This is computationally expensive as well as not very robust (edges in images are likely to be broken or occluded). In our approach, the complexity of features at the level of boundary fragments is attained by hierarchical combination of smaller edge features. Since the features will be shared at all layers of the hierarchy, this makes the method computationally more efficient.

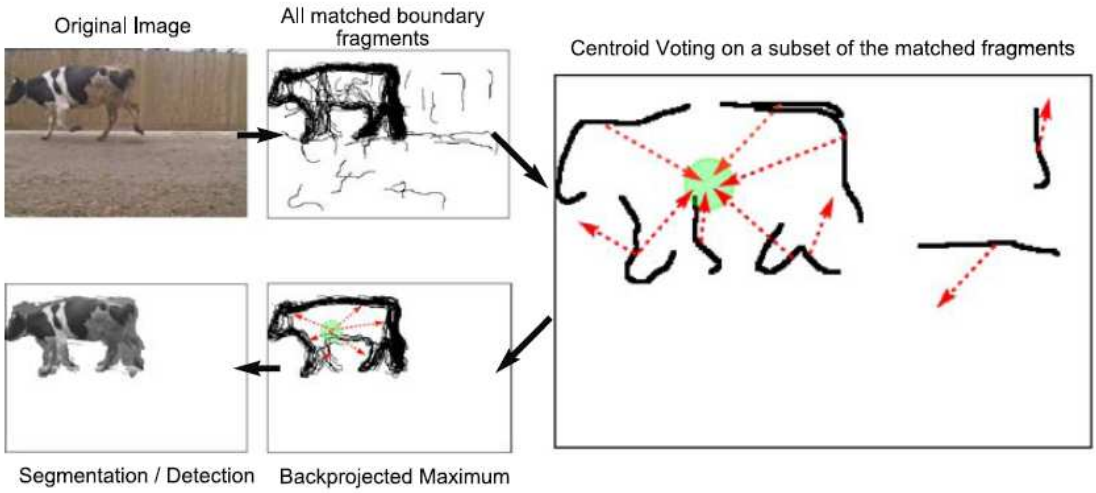


Figure 4.1: The approach by Opelt et al. [111] represents objects by long *boundary fragments* which vote for the center of an object class in images. This figure is taken from [111].

Ferrari et al. [45, 46] extracted so-called kPAS features which were learned by combining k roughly straight contour fragments. Objects were then represented as histograms over the kPAS types and a SVM classifier was learned to separate foreground objects from the background. The classifier was used with the sliding window approach to recognize object classes and localize them in images. The later extension used Hough voting of kPAS features for object center to form initial object hypotheses. Back projection and shape verification was further used to eliminate spurious hypotheses [45].

Fergus et al. [43, 44] represent objects as constellations of object parts and proposes an approach that learns the model from images without supervision and in a scale invariant manner. This approach has been later extended for incremental learning of multiple object classes [40].

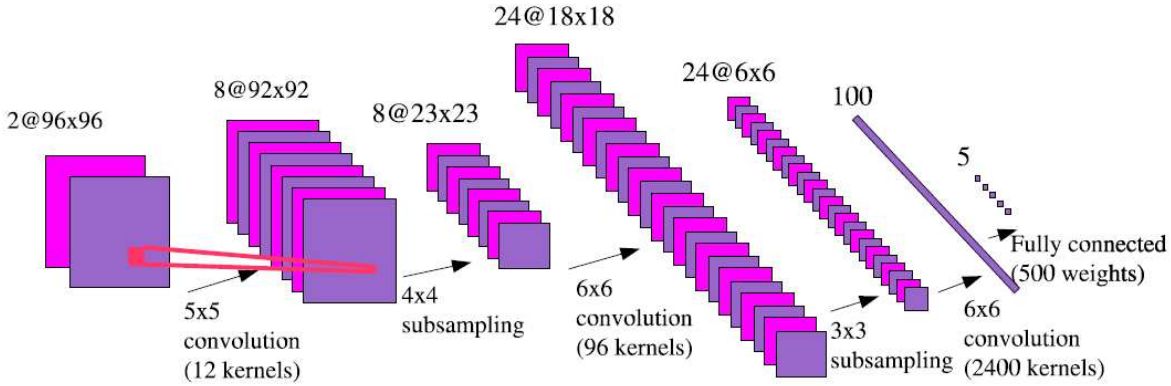


Figure 4.2: Convolutional nets [74, 119, 79] consist of a hierarchy of linear or non-linear convolutions. The top layer is a classification layer, which decides on the label of a class. This figure is taken from [74].

4.2 Neural networks

Among the NN representatives, Convolutional nets [74, 119, 79] have been most widely and successfully applied to generic object recognition. The approach builds a hierarchical feature extraction and classification system with fast feed-forward processing. The hierarchy stacks one or several feature extraction stages, each of which consists of filter bank layer, non-linear transformation layers, and a pooling layer that combines filter responses over local neighborhoods using an average or max operation, thereby achieving invariance to small distortions [79] (the approach is illustrated in Figure 4.2). Training is done with a gradient-based algorithm that minimizes a loss function. The top layer is usually a classification layer (usually SVM is used) which decides on the object class of an input pattern. Most commonly, CNNs have only been applied to classification of objects and not also to object detection.

Hinton [72] proposes a hierarchical generative model to represent the objects. The model trains a network of Restricted Boltzmann machines using a wake-sleep algorithm. The approach has been limited to binary images and applied mostly to digit recognition.

One of the main drawbacks of these approaches may be that they usually model the *appearance* of images (local patches) and not also the *shape* of the objects, which makes them less robust and flexible to local shape variations usually present in objects.

The majority of neural network approaches have also a shallow architecture (only 2 or at most 3 layers with features that are rather simple and not yet object or object part-like), while it is widely believed that going beyond the shallow architectures would move the field forward. Learning *deep belief networks* is also emphasized in the current DARPA project calls, showing the importance of the problem. Note that our approach is able to learn multiple layers (usually we build 6 layers) and achieves that the whole

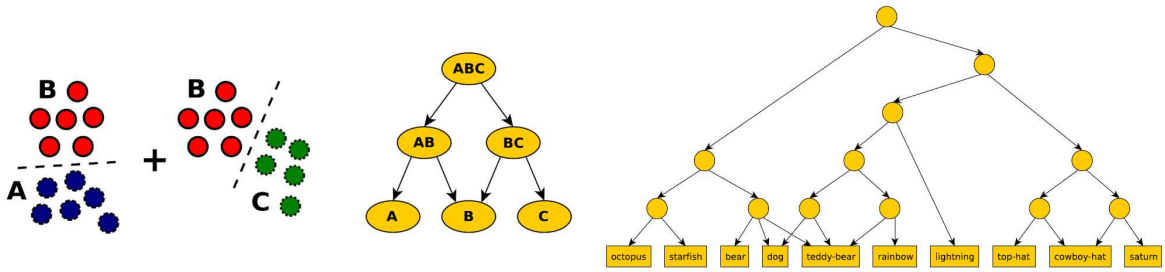


Figure 4.3: Building class taxonomies as a hierarchy of (binary) classifiers [96]. The input space is a high-dimensional vector of feature responses, however, the classes are organized by a hierarchical decision tree structure. Classification is done in time logarithmic to the number of classes. Picture taken from [96].

shapes of the objects are coded in a robust and invariant way.

4.3 Hierarchies of classifiers

The main idea of these types of approaches, which seem to be currently prevalent in the hierarchical literature, is the following. Given a class hierarchy, one can efficiently classify samples by descending the decision tree. This usually results in logarithmic complexities. In principle, any classifier can be used in the nodes of the hierarchy to make the decision about the direction of descent. In practice, Support Vector Machines are widely used for this task. Input patterns (objects) are usually represented with a high-dimensional feature vector, formed as a histogram over the responses of various feature detectors. For detection, a sliding window approach is used. The main difference between these approaches and ours is that they use a *class* hierarchy while our approach models the hierarchical representation of object *structure* (shape).

Approaches like vocabulary trees [107] that speed up feature matching are related to these works due to their hierarchical nature. Similarly, kd-trees partition a hierarchical space of features and can perform component-wise classification [115]. Both of these approaches, however, are usually applied to retrieve *object instances* in large collections of images and not to generic class recognition.

Our hierarchy is generative with respect to object structure and does not address the taxonomic organization of object classes. It is thus not directly related to these approaches. However, to scale to a large number of object classes, we might want to consider combining visual taxonomies with our approach in the future.

A review and a comparison of a wide range of representative approaches is given in [96] (one approach is illustrated in Figure 4.3).

4.4 Compositional hierarchies

Several compositional approaches to object categorization/recognition have been proposed in the literature. We separate them into two parts with respect to the degree of supervision/prior knowledge used in building the models. In Section 4.4.1 we review the models that either need labeled object parts (or even smaller object components) or employ pre-designed features or grouping rules. In Section 4.4.2 we review the approaches to unsupervised learning of object structure which are the most related to the work presented in this thesis.

4.4.1 Supervised learning approaches

In two early influential works, Sarkar and Boyer [126] proposed the use of Gestalt laws to group the elementary features together, while Dickinson et al. [33] suggested a bottom-up approach to 3D recognition of objects by matching of aspect graphs.

Following the work on interpretation trees by Grimson et al. [67], Ettinger [38] proposed a hierarchical representation based on corners, T-junctions and points of high curvature which attained admirable speed and performance given the limited hardware available at that time. A similar approach was later deployed by [123, 62], which, however, targeted a more symbolic description of objects detached of the low-level contour extraction. An object was automatically broken into a hierarchy of parts based on local symmetries and curvature, achieving a more semantic, medial-axis representation.

Geman and Bienenstock [15, 14, 66, 80] presented theoretical formulations for compositional systems (the approach is depicted in Figure 4.4) and also attempted to map the theory to the working flow of the visual cortex. Objects are formed by recursive combination of simple base elements, where, however, the grouping rules and the features are manually designed. The approach has been successfully utilized on reading license plates.

Zhu and Mumford [166] proposed a stochastic grammar to represent objects (the approach is illustrated in Figure 4.5). The main drawback of the approach is the high amount of annotation needed, whereby the objects, parts and their recursive constituents must all be defined (labeled in images) by hand. Similarly, various works by Yuille et al. [165, 24] also made use of the AND-OR graph representation and relied on supervised training of the model's parameters. It is also not clear how such approaches could scale to multiple object classes, since sharing of features may not be well predicted by human labelers.

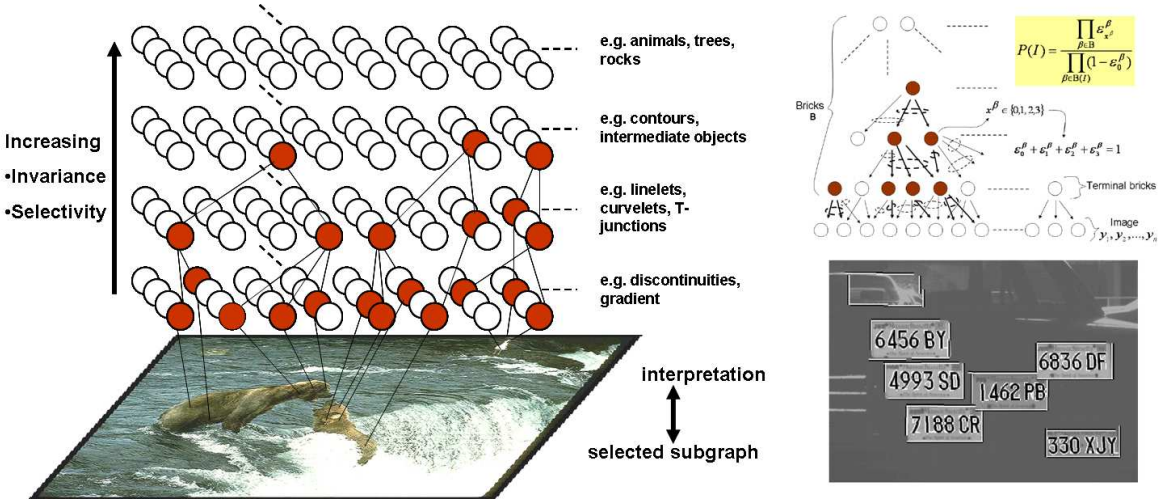


Figure 4.4: S. Geman et al. [66] represent objects by a compositional hierarchy. The features are pre-designed. The approach has been successfully utilized in reading license plates. Picture taken from [80].

4.4.2 Approaches to unsupervised structure learning

Work on *unsupervised hierarchical learning* has been relatively scarce. Most unsupervised approaches fall under the domain of neural networks [119, 98] which are conceptually very different from compositional hierarchies [66, 51].

Utans [154] has been the first to address unsupervised learning of compositional representations. The approach learned hierarchical mixture models of feature combinations, and was utilized on learning simple dot patterns.

Based on the Fukushima's model [64], Riesenhuber and Poggio [121] introduced the HMAX approach which represents objects with a 2-layer hierarchy of Gabor feature combinations (the approach is schematically depicted in Figure 4.6). The original HMAX used a vocabulary of pre-determined features, while these have subsequently been replaced with randomly chosen templates [103, 130]. Since no statistical learning is involved, as much as several thousands of features are needed to represent the object classes. Even with a single class, it takes several minutes to process a relatively small image.

The HMAX approach was also extended with more elaborated learning algorithms. Masquelier and Thorpe [98] proposed an unsupervised hebbian learning to build the hierarchy. While the approach seems to pick up intuitive, yet very complex, object parts, the whole hierarchy still have only two layers (one fixed and one learned) and the spatial variability is only indirectly built into the model (by max pooling operations). It would be interesting to see whether the approach could recover the structure of whole objects if more layers were learned.

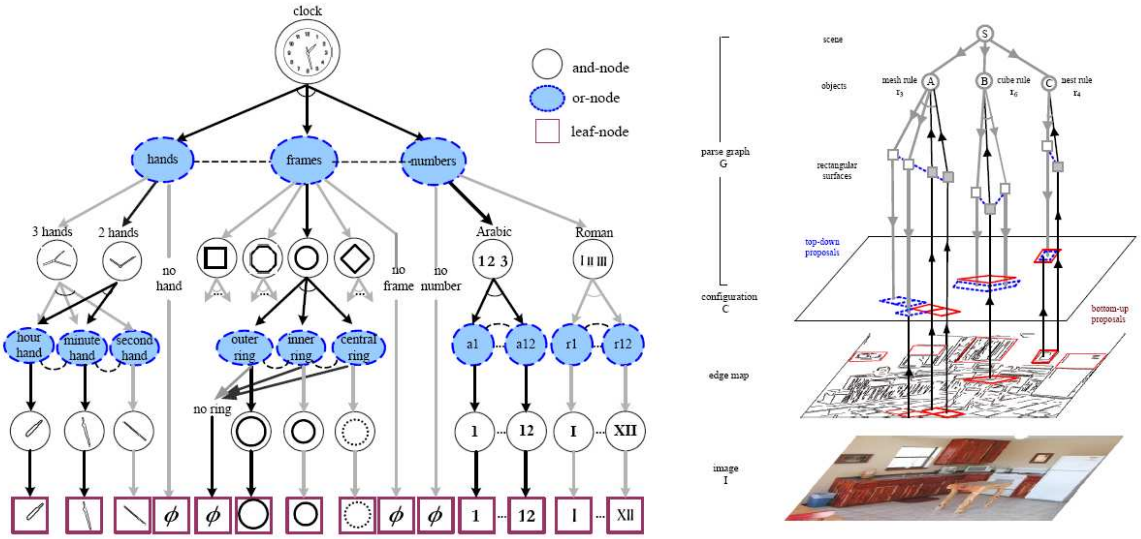


Figure 4.5: Zhu and Mumford [166] represent objects by a hierarchical stochastic grammar. In order to train the model, the object parts as well as their smaller constituents need to be labeled. Picture taken from [166].

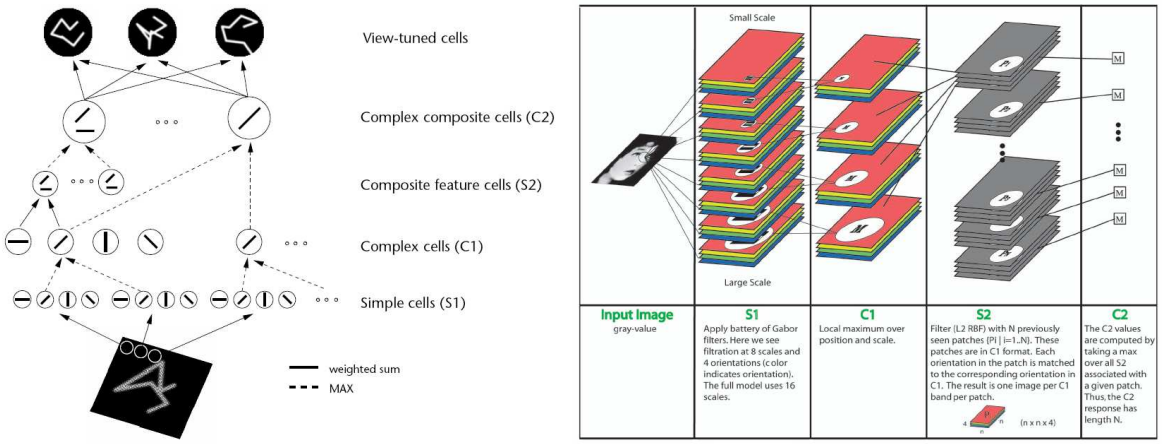


Figure 4.6: The HMAX approach [121, 130] represents objects by a compositional hierarchy. The original HMAX used a vocabulary of pre-determined features, while these have subsequently been replaced with randomly chosen templates. Picture taken from [130].

Bouchard and Triggs [21] extended the constellation model [44] to a 3−layer hierarchy and a similar representation was proposed by Torralba et al. [146, 141]. For tractability, all of these models are forced to use very sparse image information, where prior to learning and detection, a small number (around 30) of interest points are detected. Using highly discriminative SIFT features might limit their success in cluttered images or on structurally simpler objects with little texture. The repeatability of the

SIFT features across the classes is also questionable [130]. On the other hand, our approach is capable of dealing with several tens of thousands of contour fragments as input. We believe that the use of repeatable, dense and indistinctive contour fragments provide us with a higher repeatability of the subsequent object representation facilitating a better performance.

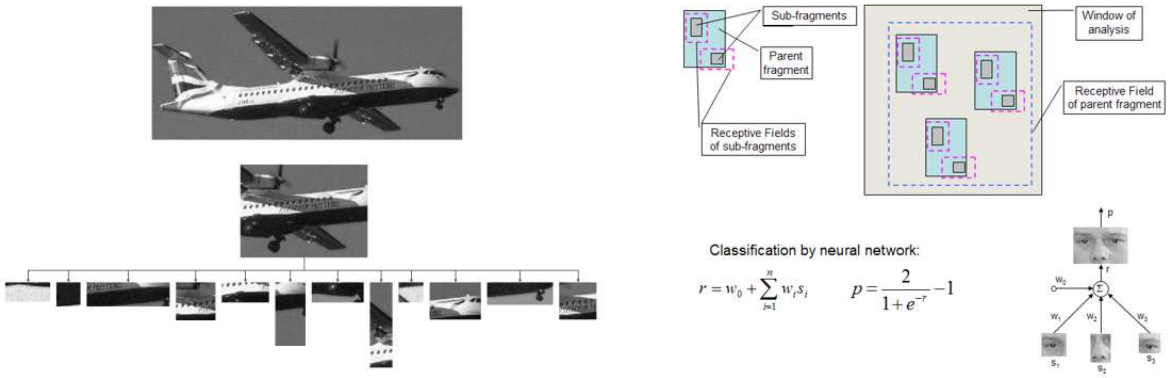


Figure 4.7: Ullman et al. [153, 152, 37] learn the object hierarchy by *decomposing* the object in recursively smaller image patches. Picture taken from [152].

Epshtein and Ullman [37] approached the representation from the opposite end; the hierarchy is built by decomposing object relevant image patches into recursively smaller entities (the approach is depicted in Figure 4.7). The approach has been utilized on learning each class individually while a joint multi-class representation has not been pursued. Mikolajczyk et al. [102] also used image patches to represent the classes, however, the authors obtained the hierarchical model by clustering the features based on appearance and not geometry. Since the patches are rigid with respect to shape deformations, a large number of them is needed to represent each class sufficiently well [102].

Todorovic and Ahuja [145] proposed a data-driven approach where a hierarchy for each object example is generated automatically by a segmentation algorithm (the approach is depicted in Figure 4.8). In learning, the largest repeatable subgraphs are found to represent the classes of the objects. Since bottom-up processes are usually unstable, exhaustive grouping is pursued which shows in training and inference times.

Ommer and Buhmann [109] proposed an unsupervised hierarchical learning approach, which has been successfully utilized for object classification. The features at each layer are defined as histograms over a larger, spatially constrained area. Our approach explicitly models the spatial relations among the features, which should allow for a more reliable detection of objects (not only classification) with lower sensitivity to background clutter.

The learning frameworks most related to ours include the work by Scalzo and

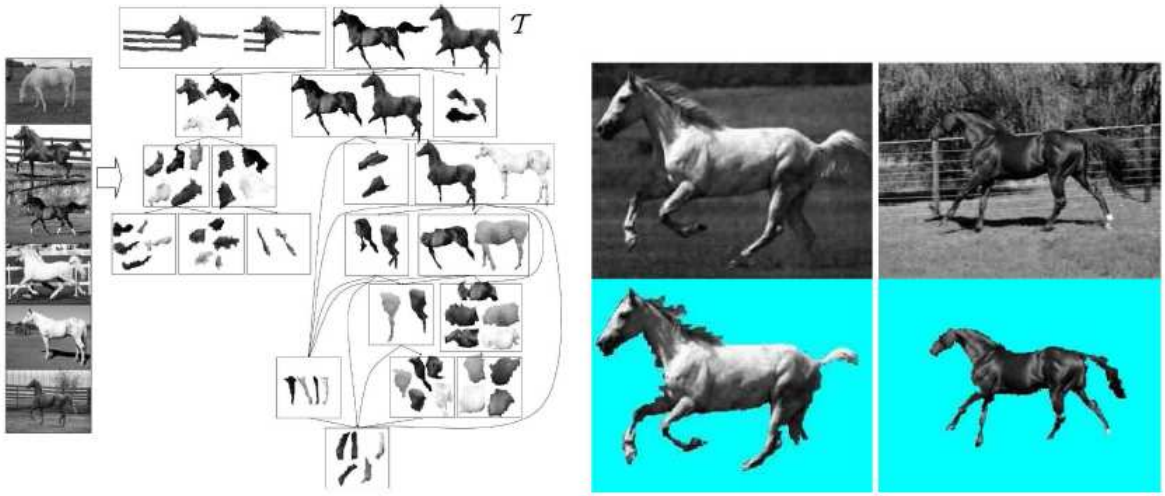


Figure 4.8: Ahuja and Todorovic [145, 5] learn a hierarchical segmentation tree-based models from examples. The hierarchy captures the recursive containment and spatial layout of regions making up the model. The model also produces object segmentation (right). Picture taken from [145].

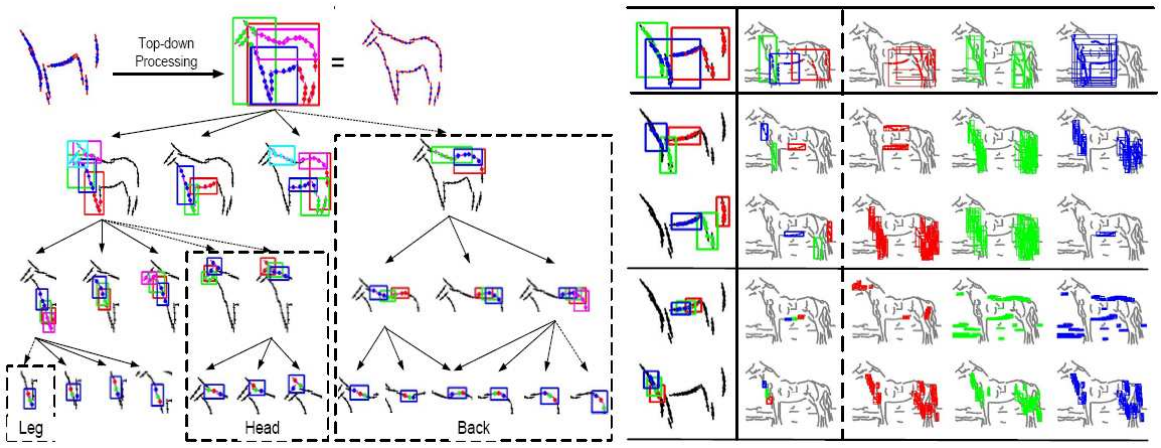


Figure 4.9: The recent approach by Zhu and Yuille [164] learns a compositional hierarchy of contour fragments in a bottom-up way. This approach is the most similar to ours. Picture taken from [164].

Piater [127], Fleuret and Geman [61] and just recently L. Zhu et al. [164] (Figure 4.9). However, all of these methods build *separate* hierarchies for each object class. This, on the one hand, avoids the massive number of possible feature combinations present in diverse objects, but, on the downside, does not exploit the shareability of features among the classes.

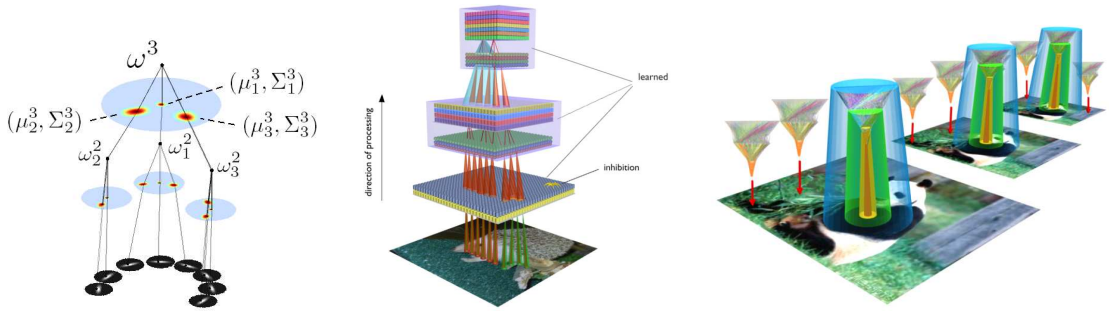
Conceptually, our approach is the closest to the work on multi-class shape repre-

sentation by Amit and Geman et al. [8, 7, 6] which also inspired our work. There are, however, substantial differences: while the conceptual representation is similar, the compositions there are designed by hand, the hierarchy has only three layers (edgelets, parts and objects), and the application is mostly concerned with reading licence plates.

To the best of our knowledge, we are the first to 1.) *learn* the most generic and repeatable shape structures at multiple hierarchical layers from images without any supervision, which, interestingly, resemble those predicted by the Gestalt theory [158] (some non-hierarchical work has been done in the domain of the ICA [92]), 2.) learn a multi-class generative compositional representation in a bottom-up manner from simple contour fragments, 3.) demonstrate logarithmic scalability of a hierarchical generative approach when the number of classes increases — in terms of the speed of inference, storage, and training times.

Chapter 5

Learning a Hierarchical Compositional Shape Vocabulary



(a) (b) (c)

Figure 5.1: (a) Example of a composition from Layer 3, (b) The hierarchical recognition architecture, (c) Vocabulary is applied in each image point (robustness to position of objects) and several image scales (robustness to objects' size). The cylinders indicate the sizes of receptive fields of compositions at different layers.

The complete hierarchical framework addresses three major issues: the representation, inference and learning.

5.1 The representation: a hierarchical compositional shape vocabulary

Our aim is to model the distribution of object *shapes* in a given class and we wish to do so for multiple object classes in a computationally efficient way. The idea is to represent the objects with a *learned hierarchical compositional shape vocabulary* that has

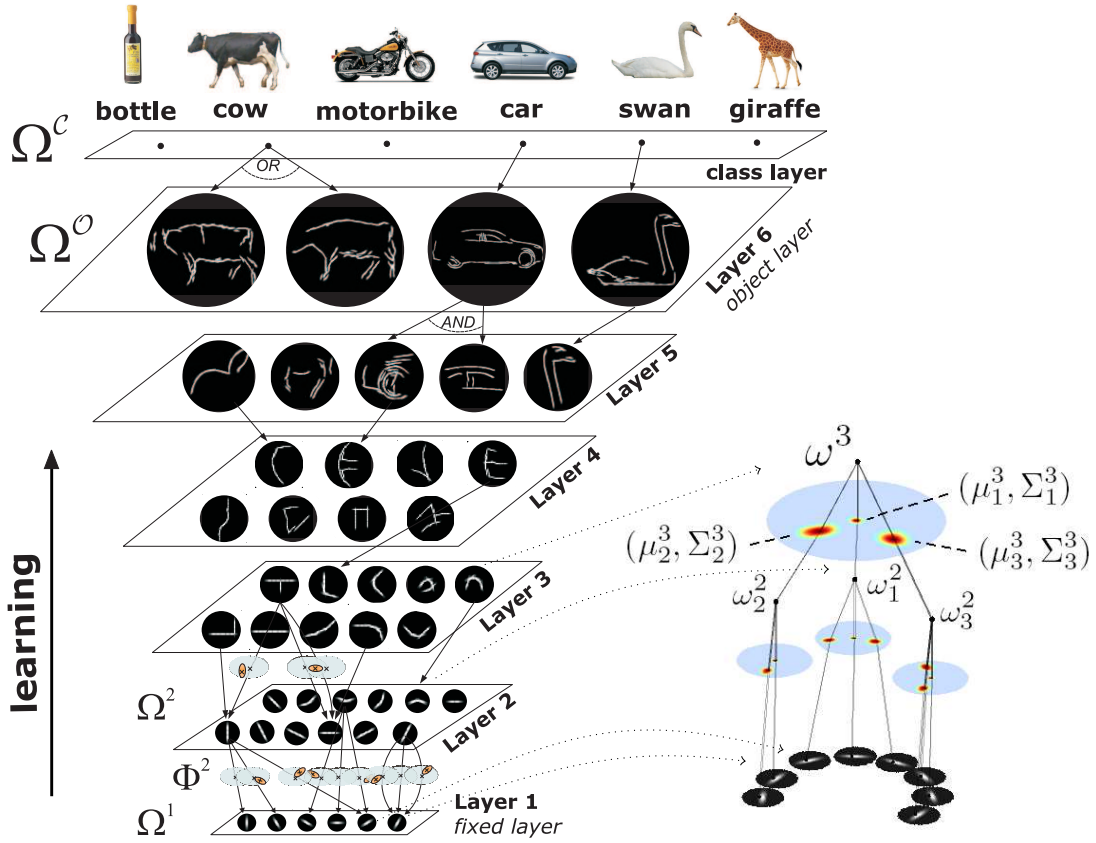


Figure 5.2: **Left**: Illustration of the hierarchical vocabulary. The depicted shapes represent the compositions ω^ℓ and the links between them denote the compositional relations between them. Note that there are no actual images stored in the vocabulary – the depicted shapes are the samples from the generative model. **Right**: An example of the complete structure of a composition from Layer 3. The blue patches are the limiting sizes within which the model can generate shapes, while the red ellipses denote the Gaussians representing the spatial relations between the parts.

the following architecture. The vocabulary at each layer contains a set of hierarchical deformable models which we will call *compositions*. Each composition is defined recursively: it is a hierarchical generative probabilistic model that represents a geometric configuration of a small number of parts which are themselves hierarchical probabilistic models, i.e., compositions from a previous layer of the vocabulary. Different compositions can share models for the parts, which makes the vocabulary efficient in size and results in faster inference. We will set up our learning algorithms in a way that, during the inference, different compositions at a particular layer of the vocabulary will compete for the explanation of an input image. That is, each patch in an image will, in most cases, be explained by only one of the compositions in the vocabulary.

Specifically, each part in a composition has an "appearance" which is defined as a (discrete) distribution over the set of compositions from the previous layer of the vocabulary. The geometric configuration of parts is modeled by relative spatial relations between each of the parts and one part called a *reference part*. The hierarchical topology of a composition is assumed to be a tree (and will be learned in a way to guarantee this), meaning that none of its parts spatially overlap. This is an assumption we make to simplify inference. Different compositions can have different topologies (different tree structures) which will be *learned* in an *unsupervised manner* from images, along with the corresponding parameters. The number of compositions that make a particular layer in the vocabulary will also not be set in advance but determined through learning. We will, however, assume supervision on the object level, i.e., the set of positive and validation images of objects in each class needs to be given. The amount of supervision the approach needs is further discussed in Section 5.3.3.

At the lowest (first) layer, the hierarchical vocabulary consists of a small number of base models that represent short contour fragments at coarsely defined orientations. The number of orientations is assumed given. At the top-most layer the compositions will represent the shapes of the whole objects. For a particular object class, a small number of top-layer compositions will be used to represent the whole distribution of shapes in the class, i.e., each top-layer composition will model a certain aspect (3D view or an articulation) of the objects within the class. The left side of Figure 5.2 illustrates the representation.

Due to the recursive definition, a composition at any layer of the vocabulary is thus an "autonomous" deformable model. As such, a composition at the top layer can act as a detector for an object class, while a composition at an intermediate layer acts as a detector for a less complex shape (for example, an L junction). Since we are combining deformable models as we go up in the hierarchy, we are capturing an increasing variability into the representation while at the same time preserving the spatial arrangements of the shape components.

We will use the following notation. Let Ω denote the set and structure of all compositions in the vocabulary and Θ their parameters. Since the vocabulary has a hierarchical structure we will write it as $\Omega = \Omega^1 \cup \Omega^2 \cup \cdots \cup \Omega^{\mathcal{O}} \cup \Omega^C$ where $\Omega^\ell = \{\omega_i^\ell\}_i$ is a set of compositions at layer ℓ. Whenever it is clear from the context, we will omit the index i. With $\Omega^{\mathcal{O}}$ we denote the top, *object layer* of compositions (we will use $\mathcal{O} = 6$ in this thesis, and this choice is discussed in Section 5.3.3), which roughly code the whole shapes of the objects. The final, object *class layer* Ω^C is not compositional, but only pools all of the corresponding object layer compositions for each class separately. Specifically, a model for a particular object class $c \in \Omega^C$ is represented as a disjunction (OR) of a (sub)set of object layer compositions.

The definition of a composition with respect to just one layer below is akin to that

of the constellation model [43]. A composition ω^ℓ, where $\ell > 1$, consists of P parts (P can be different for different compositions) with appearances $\boldsymbol{\theta_{app}}^\ell = [\theta_{app}{}_j^\ell]_{j=1}^P$ and geometric parameters $\boldsymbol{\theta_g}^\ell = [\theta_{g_j}^\ell]_{j=1}^P$. The appearance $\theta_{app}{}_j^\ell$ of a part is a discrete distribution over the compositions from the previous layer. For example, $\theta_{app}{}_j^\ell(\omega_k^{\ell-1}) = 0.8$ means that the j-th part of ω^ℓ is 0.8 likely to be the k-th composition $\omega_k^{\ell-1}$ from layer $\ell - 1$. The geometric relations between the position of each of the parts relative to the reference part are modeled by two-dimensional Gaussians with parameters $\theta_{g_j}^\ell = (\mu_j^\ell, \Sigma_j^\ell)$. For the convenience of notation later on, we will also represent the location of a reference part with respect to itself with a Gaussian having zero mean and small variance, ϵ^2Id.

The models at the first layer of the vocabulary are defined over the space of image features, which will in this thesis be n-dimensional Gabor feature vectors (explained in Section 5.2.1). Each model ω_i^1 has an appearance distribution over the feature vectors, which is taken to be an n-dimensional Gaussian with parameters (μ_i^1, Σ_i^1).

Lastly, the class layer models are only specified by an appearance distribution over the object layer compositions. A class model c thus has the appearance distribution $\theta_{app}{}_c^C$, which is non-zero only for those object layer compositions that represent the object shapes in class c. We will denote this set by $\{\omega_{i_c}^{\mathcal{O}}\}_{i_c} \subset \Omega^{\mathcal{O}}$, with $\theta_{app}{}_c^C(\omega_{i_c}^{\mathcal{O}}) > 0$ for each i_c and $\theta_{app}{}_c^C(\omega_{i'_c}^{\mathcal{O}}) = 0$ for all other i'_c. The sets of object layer compositions for two different classes are always disjoint.

Note that each composition can only generate shapes within a limited spatial extent, that is, the window around a generated shape is of limited size and the size is the same for all compositions at a particular layer ℓ of the vocabulary. We will denote it with r^ℓ. The limiting size r^ℓ increases exponentially with the level of hierarchy. Figure 5.3 depicts the gradual increase in complexity and size of the compositions in the hierarchy.

The complete hierarchical vocabulary will be denoted with $\mathcal{V} = (\Omega, \Theta)$, while the vocabulary of compositions at a particular layer will be denoted with $\mathcal{V}^\ell = (\Omega^\ell, \Theta^\ell)$.

5.2 Recognition and detection

The model is best explained by first considering inference. Thus, let us for now assume that the representation is already known and we are only concerned with recognizing and detecting objects in a query image given the model. How we learn the representation will be explained in Section 5.3.

Subsection 5.2.1 explains the image features we use, which form a set of observations, and how we extract them from an image. In Subsection 5.2.2 we describe the inference of the hidden states of the model, both theoretically and practically. In

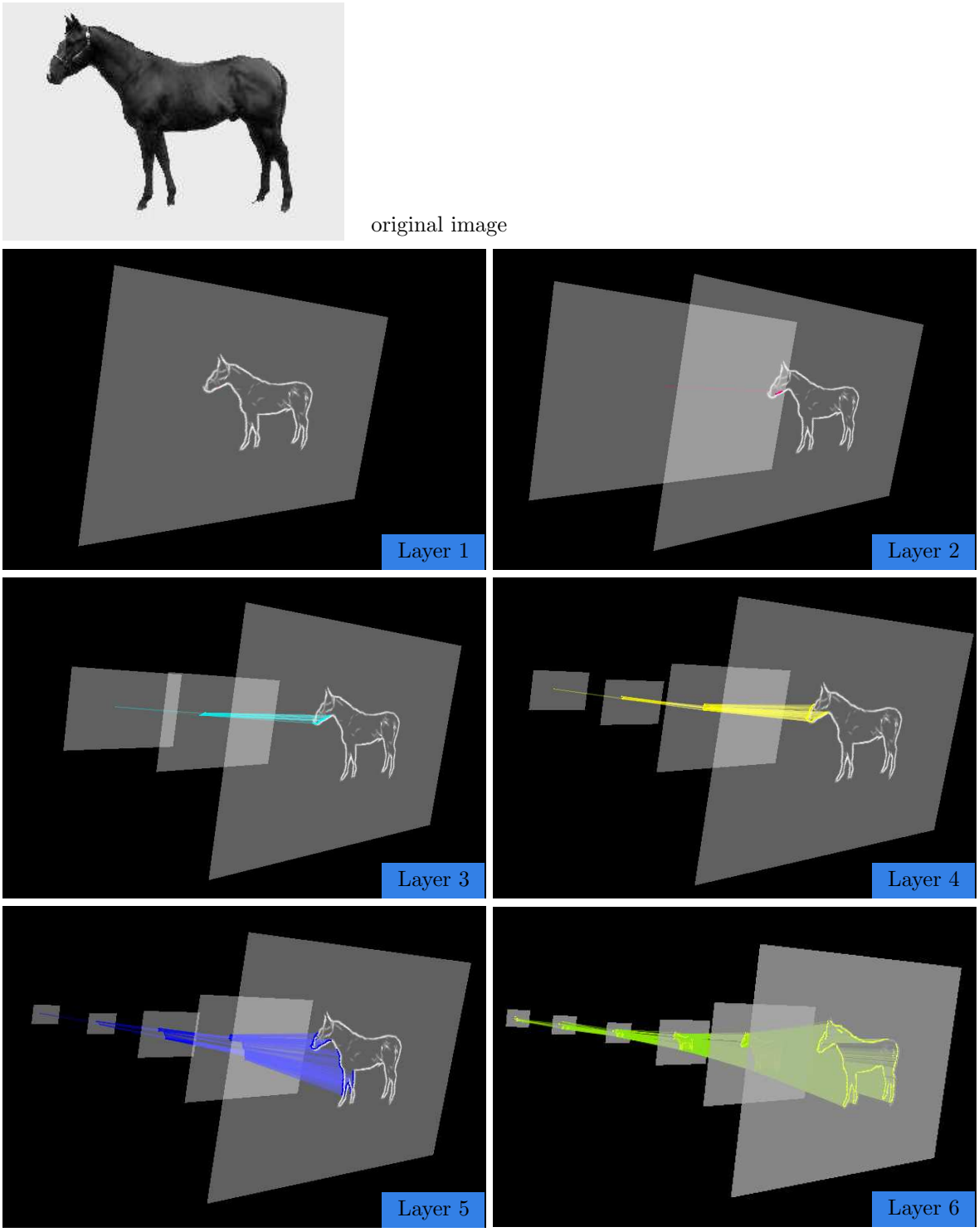


Figure 5.3: The figure shows inferred parse trees in an image corresponding to compositions chosen from different layers of the vocabulary. The complexity and the size of shapes explained by the compositions gradually increases with the level of hierarchy. A composition at the first layer explains only a single point (edge fragment), while a composition at the top layer explains the whole shape of the object.

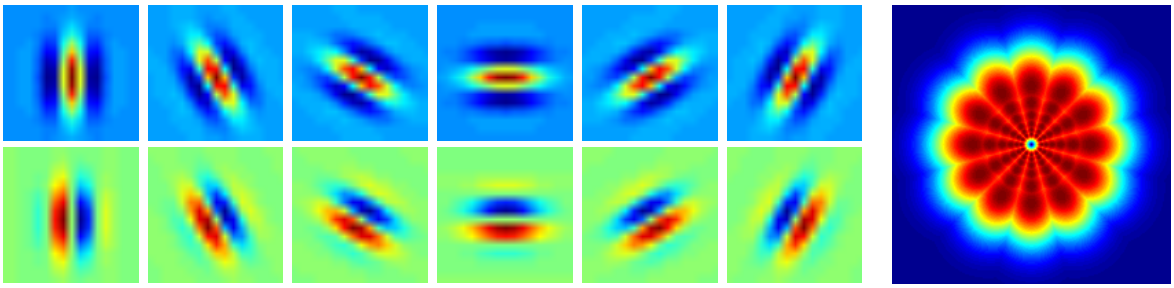


Figure 5.4: Examples of filters. Top row: Even Gabor filters. Bottom row: Odd Gabor filters. Right: Spectra of the filterbank.

Subsection 5.2.3 we explain how we detect objects given the inferred hidden states.

5.2.1 Extracting image features

Let I denote a query image. The features which we extract from I should depend on how we define the base models at the first layer Ω^1 of the vocabulary. In this thesis we choose to use oriented edge fragments, however, the learning and the recognition procedures are general and independent of this particular choice.

In order to detect oriented edges in an image we use a Gabor filter bank:

$$g_{\lambda,\varphi,\gamma,\sigma}(x,y,\psi) = e^{-\frac{u^2+\gamma^2v^2}{2\sigma^2}} \cos\left(\frac{2\pi u}{\lambda} + \varphi\right)$$
$$u = x\cos\psi - y\sin\psi, \quad v = x\sin\psi + y\cos\psi,$$

where (x, y) represents the center of the filter's domain, and the parameters in this thesis are set to $(\lambda, \gamma, \sigma) = (6, 0.75, 2)$. A set of two filter banks is used, one with even, $\varphi = 0$, and the other with odd, $\varphi = -\frac{\pi}{2}$, Gabor kernels defined for n equidistant orientations, $\psi = i\frac{\pi}{n}$, $i = 0, 1, \ldots, n-1$. For the experiments in this thesis we use $n = 6$.

We convolve the image I with both filter banks and compute the total energy for each of the orientations [114]:

$$\mathcal{E}(x,y,\psi) = \sqrt{r_0^2(x,y,\psi) + r_{-\pi/2}^2(x,y,\psi)}, \tag{5.1}$$

where $r_0(x, y, \psi)$ and $r_{-\pi/2}(x, y, \psi)$ are the convolution outputs of even and odd Gabor filters at location (x, y) and orientation ψ, respectively. We normalize $\mathcal{E}$ to have the highest value in the image equal to 1. We further perform a non-maxima suppression over the total energy $\mathcal{E}$ to find the locations of the local maxima for each of the orientations. This is similar to performing the Canny operator and taking the locations of

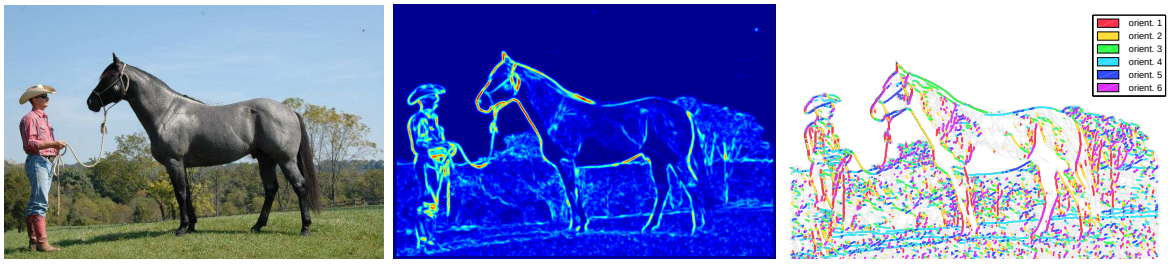


Figure 5.5: Left: Original image. Middle: Maximum over orientations, $\arg\max_\psi E(x, y, \psi)$. Right: Points in which features, $\mathbf{f}$, are extracted. Different colors denote the dominant orientations encoded in each feature $\mathbf{f}$.

the binary responses. At each of these locations (x, y), the set of which will be denoted with $\mathbf{X} = \{(x, y)\}$, we extract Gabor features $\mathbf{f} = \mathbf{f}(x, y)$ which are n-dimensional vectors containing the orientation energy values in (x, y), $\mathbf{f}(x, y) = \left[\mathcal{E}(x, y, i\frac{\pi}{n})\right]_{i=0}^{n-1}$. Specifically, the feature set $\mathbf{F}$ is the set of all feature vectors extracted in the image, $\mathbf{F} = \{\mathbf{f}(x, y), (x, y) \in \mathbf{X}\}$. The number of features in $\mathbf{F}$ is usually around 10^4 for an image of average size and texture. The image features $(\mathbf{F}, \mathbf{X})$ serve as the *observations* to the recognition procedure. Figure 5.5 shows an example of feature extraction.

Scale. The feature extraction as well as the recognition process is performed at several scales of an image: we use a Gaussian pyramid with two scales per octave and process each scale separately. That is, feature extraction and recognition up to the object level $\mathcal{O}$ are performed at every scale and independently of other scales. For detection of object classes we then consider information from *all* scales as explained in Section 5.2.3. Not to overload the notation, we will just refer to one scale of I.

5.2.2 Inference

Performing inference in a query image with a given vocabulary entails inferring a hierarchy of hidden states from the observations (which are the image features $(\mathbf{F}, \mathbf{X})$). The hidden states at the first layer will be the only ones receiving a direct input from the observations, while all other states will receive input from a hidden layer below.

Each hidden state z^ℓ has two variables $z^\ell = (\omega^\ell, x^\ell)$, where ω^ℓ denotes a composition from the vocabulary and x^ℓ a location in an image (to abbreviate notation from here on, we will use x denote the location vector instead of writing (x, y)). Note that ω^ℓ is in fact a hierarchical probabilistic model, so the first variable is just a reference to the model. The notation x^ℓ is used to emphasize that x is from a spatial grid with resolution corresponding to hidden layer ℓ, i.e., the spatial resolution of the locations of the hidden states will be increasingly coarser with the level of hierarchy.

We address three problems:

1. **Image likelihood.** Calculating the likelihood of an observed image patch J under a particular composition (hierarchical model), say ω^ℓ: $p(J|\omega^\ell)$.

2. **Most probable hidden state activation.** Finding the most likely tree $\mathcal{T}^*(J, \omega^\ell)$ of hidden state activations to have generated an observed image patch J given the model ω^ℓ: $\mathcal{T}^*(J, \omega^\ell) = \arg\max_{\mathcal{T}} p(\mathcal{T}|J; \omega^\ell)$.

3. **Multi-class object detection and recognition.** Finding the interpretation of an image in terms of object classes:

   (a) Finding the patches J of objects and the class c they belong to: $\{(J_i, c_i) : p(J_i|c_i) > \tau_{c_i}$ and $J_i \cap J_k = \emptyset;\ \forall\,(i,k), i \neq k\}$, where τ_{c_i} is a class-specific threshold that separates foreground (object) against the background.

   (b) Finding object segmentations which will be defined by the *leaves* (the observations) of the most probable hidden state tree activations $\mathcal{T}^*(J_i, c_i)$ for the detected objects (J_i, c_i).

Calculating the likelihood

Since we will be dealing with images of much larger size than r^ℓ, it is convenient to define $I(x^\ell)$ to be a *patch* extracted from image I, centered in x^ℓ with radius r^ℓ,

$$I(x^\ell) = \{(\mathbf{f}, x) \in (\mathbf{F}, \mathbf{X}) :\ ||x - x^\ell||_2 \leq r^\ell\} \tag{5.2}$$

Notice that $I(x^\ell)$ is not a patch cropped directly from an image, but a collection of features in a circular region around x^ℓ.

For a given composition, e.g. ω^ℓ, we would like to calculate the likelihood $p(I(x^\ell)|\omega^\ell)$ of an observed image patch $I(x^\ell)$ under ω^ℓ. By denoting $z^\ell = (\omega^\ell, x^\ell)$ as a hidden state centered in x^ℓ (in the center of the patch), then the following formulations are equivalent and will be used interchangeably throughout the thesis: $p(I(x^\ell)|\omega^\ell) = p(I(x^\ell), z^\ell|\omega^\ell) = p(I(x^\ell)|z^\ell)$. Next, let $\mathcal{T}$ denote a tree of hidden states where the root (top) node is z^ℓ and the leaves (layer 0) are the observations. To calculate the likelihood, we thus need to sum over all possible such trees:

$$p(I(x^\ell)|\omega^\ell) = \sum_{\mathcal{T}} p(I(x^\ell), \mathcal{T}|\omega^\ell) \tag{5.3}$$

Since there is an exponential number of possible trees of hidden states, calculating this expression directly is intractable. However, we can exploit the recursive definition of the compositions to get also a recursive formulation for the likelihood.

We first write down the joint distribution. Let $\mathcal{T}^{\ell'}$ denote the set of states at layer ℓ' of the tree $\mathcal{T}$, let $\mathcal{T}(z^{\ell'})$ be the subtree of $\mathcal{T}$ that has the root $z^{\ell'} \in \mathcal{T}^{\ell'}$, and let

$\mathcal{T}^{\ell-1} = [z_j^{\ell-1} := (\omega_j^{\ell-1}, x_j^{\ell-1}))]_{j=1}^{|\mathcal{T}^{\ell-1}|}$ be the hidden states at layer $\ell-1$ of the tree (just one layer below the root). We will re-write the joint $p(I(x^\ell), \mathcal{T}|\omega^\ell)$ into a recursive formulation. We first factorize it in the following way

$$p(I(x^\ell), \mathcal{T}|\omega^\ell) = \prod_{j=1}^{P} \underbrace{p_F(I(x^\ell), \mathcal{T}(z_j^{\ell-1}), z_j^{\ell-1}|z^\ell)}_{\text{foreground under part } j} \cdot \underbrace{\prod_{k=P+1}^{|\mathcal{T}^{\ell-1}|} p_B(I(x^\ell), \mathcal{T}(z_k^{\ell-1}), z_k^{\ell-1}|z^\ell)}_{\text{background}} \tag{5.4}$$

The foreground term can be further factorized:

$$p_F(I(x^\ell), \mathcal{T}(z_j^{\ell-1}), z_j^{\ell-1}|z^\ell) = p_F(z_j^{\ell-1}|z^\ell) \cdot p_F(I(x^\ell), \mathcal{T}(z_j^{\ell-1})|z_j^{\ell-1}; \omega^\ell) \tag{5.5}$$

Since conditioning on layer $\ell-1$ renders lower layers independent of ℓ, the second term becomes

$$p_F(I(x^\ell), \mathcal{T}(z_j^{\ell-1})|z_j^{\ell-1}; \omega^\ell) = p(I(x_j^{\ell-1}), \mathcal{T}(z_j^{\ell-1})|\omega_j^{\ell-1}) \tag{5.6}$$

Given how we defined the compositions, the term $p_F(z_j^{\ell-1}|z^\ell)$ factorizes into appearance and geometry:

$$p_F(z_j^{\ell-1}|z^\ell) = \underbrace{p(\omega_j^{\ell-1} \mid \theta_{app_j}{}^\ell)}_{\text{appearance}} \underbrace{p(x_j^{\ell-1} \mid x^\ell; \theta_{g_j}{}^\ell)}_{\text{geometry}} = \theta_{app_j}{}^\ell(\omega_j^{\ell-1})\, \mathcal{N}(x_j^{\ell-1} - x^\ell \mid \mu_j^\ell, \Sigma_j^\ell) \tag{5.7}$$

where the first term is a probability of j-th part under ω^ℓ having the appearance $\omega_j^{\ell-1}$ (see the definition of the representation in Section 5.1). The second term is the spatial compatibility between the location of state $z_j^{\ell-1}$ and location x^ℓ of the parent under the model for part j.

Similarly as in (5.5), the background term in (5.4) factorizes as:

$$\begin{aligned} p_B(I(x^\ell), \mathcal{T}(z_k^{\ell-1}), z_k^{\ell-1}|\omega^\ell) &= p_B(z_k^{\ell-1}|z^\ell)\, p_B(I(x_k^{\ell-1}), \mathcal{T}(z_k^{\ell-1})|z_k^{\ell-1}; \omega^\ell) \\ &= \alpha \cdot \big(1 - p(I(x_k^{\ell-1}), \mathcal{T}(z_k^{\ell-1})|\omega_k^{\ell-1})\big), \end{aligned} \tag{5.8}$$

where the probability $p_B(z_k^{\ell-1}|z^\ell)$ that a state $z_k^{\ell-1}$ belongs to the background is taken to be α, and the likelihood of the observations under a background distribution is taken to be 1−the likelihood of the same observation to be the foreground.

The joint distribution in (5.4) can now be written in a recursive form:

$$\begin{aligned} p(I(x^\ell), \mathcal{T}|\omega^\ell) = &\Big(\prod_{j=1}^{P} \theta_{app_j}{}^\ell(\omega_j^{\ell-1})\, \mathcal{N}(x_j^{\ell-1} - x^\ell \mid \mu_j^\ell, \Sigma_j^\ell) \cdot p(I(x_j^{\ell-1}), \mathcal{T}(z_j^{\ell-1})|\omega_j^{\ell-1}) \Big) \\ &\cdot \Big(\alpha^{N_B} \prod_{k=P+1}^{|\mathcal{T}^{\ell-1}|} \big(1 - p(I(x_k^{\ell-1}), \mathcal{T}(z_k^{\ell-1})|\omega_k^{\ell-1})\big) \Big), \end{aligned} \tag{5.9}$$

where $N_B = |\mathcal{T}^{\ell-1}| - P$ is the number of hidden states in $\mathcal{T}^{\ell-1}$ belonging to the background.

Due to the recursive formulation of the joint, we can now also get a recursive formulation for the likelihood in (5.3). Let us define $q(I, z) = p(I(x)|\omega)$, where $z = (\omega, x)$. By manipulating the sums in (5.3) and (5.9), we obtain the following form for the likelihood:

$$q(I, z^\ell) = \sum_{\mathcal{T}^{\ell-1}} \Big[\Big(\prod_{j=1}^{P} \theta_{app_j}^{\ell}(\omega_j^{\ell-1}) \, \mathcal{N}(x_j^{\ell-1} - x^\ell \mid \mu_j^\ell, \Sigma_j^\ell) \cdot q(I, z_j^{\ell-1}) \Big) \cdot \alpha^{N_B} \prod_{k=P+1}^{|\mathcal{T}^{\ell-1}|} (1 - q(I, z_k^{\ell-1})) \Big] \tag{5.10}$$

To complete the recursion we only need to define the data term:

$$q(I, z^1) = p(I(x^1) \mid z^1) = p(\mathbf{f}(x) \mid z^1) = \mathcal{N}(\mathbf{f} \mid \mu^1, \Sigma^1) \tag{5.11}$$

$q(I, z^1)$ is a likelihood of a feature vector $\mathbf{f}$ (which is in our case a 6-dimensional vector of normalized Gabor filter outputs) to be generated with a model ω^1 for a particular edge orientation. It is modeled with a (6-dimensional) Gaussian distribution.

The likelihoods can thus be computed efficiently by dynamic programming, similarly as in the forward algorithm for Hierarchical HMM [57]. This procedure along with further speed-ups inspired by the coarse-to-fine search of Amit and Geman et al. [8, 61, 17] is described in the section on efficient computation.

Finding most probable hidden state activation

Given a model ω^ℓ we would additionally like to find the most probable tree of hidden states that generated an image patch $I(x^\ell)$:

$$\mathcal{T}^*(z^\ell) = \arg\max_{\mathcal{T}} p(\mathcal{T}|I(x^\ell); \omega^\ell) = \arg\max_{\mathcal{T}} p(I(x^\ell), \mathcal{T}|\omega^\ell) \tag{5.12}$$

Since $\mathcal{T}^*$ depends on both x^ℓ and ω^ℓ, which make a state $z^\ell = (\omega^\ell, x^\ell)$, we use the notation $\mathcal{T}^*(z^\ell)$. By defining $\delta(I, z) = \max_{\mathcal{T}} p(I(x), \mathcal{T}|\omega)$, we can write a recursion for $\delta(I, z^\ell)$ similar to that in (5.10), however, with max instead of the sum:

$$\delta(I, z^\ell) = \max_{\mathcal{T}^{\ell-1}} \Big[\Big(\prod_{j=1}^{P} \theta_{app_j}^{\ell}(\omega_j^{\ell-1}) \, \mathcal{N}(x_j^{\ell-1} - x^\ell \mid \mu_j^\ell, \Sigma_j^\ell) \cdot \delta(I, z_j^{\ell-1}) \Big) \cdot \alpha^{N_B} \prod_{k=P+1}^{|\mathcal{T}^{\ell-1}|} (1 - \delta(I, z_k^{\ell-1})) \Big] \tag{5.13}$$

Here $\delta(I, z^1) = \mathbf{f}(x^1)$ completes the recursion. The tree $\mathcal{T}$ can then be found by dynamic programming, similarly as in the Viterbi recurrence for the Hierarchical HMM [57]: we compute $\delta(I, z^\ell)$ sequentially, from bottom layer to top and for each hidden state keep back-pointers to the states at the previous layer that produced the maximum. The complete tree $\mathcal{T}^*$ can then be found by tracing back the back-pointers in the resulting graph.

We will refer to the leaves of the tree $\mathcal{T}^*(z^\ell)$ as the *support* of a hidden state z^ℓ, and denote it with $\text{supp}(z^\ell)$.

Efficient computation

This section explains how to calculate the likelihoods and the most probable trees efficiently. We will employ the following approach:

- *Recursive bottom-up computation*: Compute the likelihoods of the hidden states in a recursive bottom-up fashion, where the intermediate computations are shared among higher layer hidden states.

- *Coarse-to-fine computation*: Prior to computing the likelihood of a state we will first perform a *learned* computationally inexpensive *test* which will decide whether the likelihood of that state will be computed or whether the state will be *pruned*.

Since ultimately, we want to recognize and detect objects in images, our goal is to compute the image likelihoods under each of the object-layer compositions $\{\omega_i^{\mathcal{O}}\}$ and do so for all image patches $\{I(x_j^{\mathcal{O}})\}$ (we will show how the classes of the objects can be inferred based on these likelihoods in Section 5.2.3). In order to compute the likelihood for a particular state $z_{ij}^{\mathcal{O}} = (\omega_i^{\mathcal{O}}, x_j^{\mathcal{O}})$, the recursive formulation in (5.10) implies a bottom-up computation: start by computing the likelihoods of the hidden states at the first layer and continuing up. Note that we do not need to compute the likelihoods of the intermediate hidden states for each $z_{ij}^{\mathcal{O}}$ separately, but can compute each of them only once (different states defined over overlapping patches can re-use the intermediate hidden states).

Notice, however, that this entails computing the likelihoods $q(I, z^\ell)$, where $z^\ell = (\omega^\ell, x^\ell)$, for all possible locations x^ℓ over the image and all compositions ω^ℓ from the vocabulary, and for each of the layers $\ell = 1, \ldots, \mathcal{O}$, which would still cause inference to run slow. To make the computations efficient, we will exploit the fact that most hidden states will be highly unlikely. We will make use of an idea of *coarse-to-fine* search proposed by Amit and Geman et al. [8, 61, 17]. The strategy will be to *prune* unpromising hidden states based on computationally inexpensive *tests*. In the case the test returns

a positive answer the likelihood for a particular state will be computed, while otherwise the state will be pruned. Note that due to pruning, the resulting likelihoods in the hierarchy will not be exact. However, since only the very unlikely states will be pruned, the computed likelihoods will represent a sufficiently good approximation. We explain the complete procedure next.

Let $\mathcal{G} = \mathcal{G}(I) = (Z, E)$ denote an *inference graph*, where the nodes Z of the graph are the ("promising") hidden states z^ℓ. With each state z^ℓ we also associate two values: the likelihood $q(I, z^\ell)$ and the probability of the most probable hidden tree $\delta(I, z^\ell)$. Graph $\mathcal{G}$ has a hierarchical structure where the vertices Z are partitioned into vertex layers 1 to $\mathcal{O}$, that is, $Z = Z^0 \cup Z^1 \cup \cdots \cup Z^{\mathcal{O}}$. The vertex layer Z^ℓ contains the hidden states z^ℓ from layer ℓ, while Z^0 contains the observations $(\mathbf{F}, \mathbf{X})$ (each $(\mathbf{f}, x) \in (\mathbf{F}, \mathbf{X})$ is a node in Z^0).

Graph $\mathcal{G}$ is built from bottom (layer 1) to top (layer $\mathcal{O}$). For a hidden state $z^\ell = (\omega^\ell, x^\ell)$ we first perform a simple learned test $T_{\omega^\ell}(I, z^\ell)$ that has the following form:

$$T_{\omega^\ell}(I, z^\ell) = \mathbf{1}\left(\delta(I, z^\ell) > \tau_{\omega^\ell}\right), \tag{5.14}$$

where $\mathbf{1}$ is an indicator function and τ_{ω^ℓ} is a learned threshold. A notation τ_{ω^ℓ} is used, indicating that each composition ω^ℓ in the vocabulary can have its own threshold. If T returns a positive answer then we add the hidden state z^ℓ to Z^ℓ and calculate its likelihood $q(I, z^\ell)$. In the case when $T = 0$, the hidden state z^ℓ is pruned, i.e., its likelihood is not calculated and the state is not added to the inference graph. Intuitively, a hidden state will be pruned if only if the most probable hidden subtree for that state has a very small probability. During the process of training (until the hierarchy is built up to the level of objects), these thresholds are fixed and set to very small values (in this thesis we use $\tau = 0.05$). Their only role is to prune the very unlikely hypotheses and thus both, speed-up computation as well as minimize the memory/storage requirements (the size of $\mathcal{G}$). After the class models (the final, layer Ω^C of the vocabulary) are learned we can also learn the thresholds in a way to optimize the speed of inference while retaining the detection accuracy, as will be explained in Section 5.3.3.

For each state added to Z^ℓ we additionally form edges in E from the state node z^ℓ to the states $\mathcal{T}_*^{\ell-1}(z^\ell) = [z_j^{\ell-1}] \in Z^{\ell-1}$ that produced the value $\delta(I, z^\ell)$.

Reduction in spatial resolution. After each layer Z^ℓ is built, we perform *spatial contraction*, where the locations x^ℓ of the hidden states z^ℓ are downsampled by a factor $\rho^\ell < 1$. Among the states that code the same composition ω^ℓ and for which the locations (taken to have integer values) become equal, only the state that has the highest likelihood is kept. We use $\rho^1 = 1$ and $\rho^\ell = 1/2$, $\ell > 1$. This step is mainly meant to reduce the computational load: by reducing the spatial resolution at

each layer, we bring the far-away (location-wise) hidden states closer. This, indirectly (through learning), keeps the scales of the Gaussians approximately the same for all compositions in the vocabulary and makes inference faster.

5.2.3 Object class detection and recognition

This section explains how to perform

1. **multi-class object recognition and detection**,
2. **segmentation and parsing of the detected objects**

once we have obtained (by using the procedures described in the previous sections) the inference graph $\mathcal{G}$ built to hidden layer $Z^{\mathcal{O}}$.

We first show how to compute the state likelihoods under the class models from the vocabulary (layer Ω^C) given $\mathcal{G}$. Based on these likelihoods we make a decision on the presence of an object at a certain location. An object detection will, in this thesis, taken to be a hidden state $z^C = (c, x^C)$ with likelihood $q(I, z^C)$ higher than a class-specific threshold, that is, a threshold is used to decide whether a certain state belongs to the foreground (object) or background. Once this simple decision is made we could additionally use a more sophisticated verification step, e.g., employing SVM over the features in the detected region as in [110], however, we have not employed one in this thesis.

Let Z^C be a "class" hidden layer of $\mathcal{G}$ and let $z^C = (c, x^C) \in Z^C$ be a hidden state that represents the presence of an object class c at location x^C. Following (5.10), we can compute its likelihood $q(I, z^C)$ as follows:

$$\begin{aligned} q(I, z^C) = p(I \mid z^C) &= \sum_{z_i^{\mathcal{O}} = (\omega_i^{\mathcal{O}}, x^C)} p(z_i^{\mathcal{O}} \mid z^C)\, q(I, z_i^{\mathcal{O}}) \\ &= \sum_{z_i^{\mathcal{O}}} \theta_{app_c}{}^C(\omega_i^{\mathcal{O}})\, q(I, z_i^{\mathcal{O}}) \end{aligned} \tag{5.15}$$

Note that the sum runs over all possible states $z_i^{\mathcal{O}}$ at location x^C. However, since $\theta_{app_c}{}^C$ is zero for all but a subset of layer $\mathcal{O}$ compositions, i.e., $\{\omega_{i_c}^{\mathcal{O}}\}$, we only need to sum over the corresponding hidden states $z_{i_c}^{\mathcal{O}}$.

For object detection, we will only keep those states z^C that have likelihoods higher than a threshold, $q(I, z^C) > \tau_c$. This simple decision is used to separate foreground (object) from the background. Note that τ_c is a class-specific threshold, meaning that different decisions are made for different classes. The hidden states not satisfying the decision inequality are removed from Z^C.

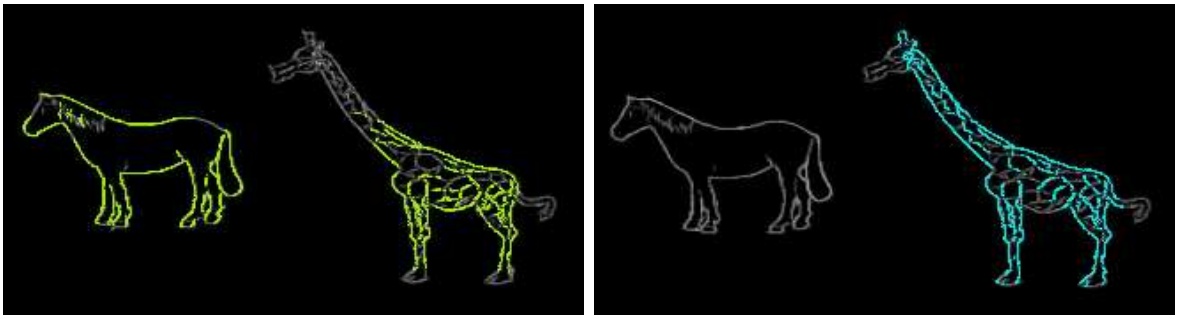


Figure 5.6: **Left**: The supports of object hypotheses for class *horse*. **Right**: Same for *giraffe*. See text for details.

However, all nodes from Z^C do not yet represent the final object detections, but define a set of *object hypotheses*. Since different patches in an image were assumed independent during inference, we can potentially have multiple different hypotheses in Z^C explaining partly the same image area, and thus a selection between the hypotheses is necessary. This is illustrated in Figure 5.6: the left side shows the supports of candidate detections for the horse class and the right side for the giraffe class. Ultimately, we would like to select the giraffe detection over the horse one, since it explains a larger portion of the image, even though its likelihood might be somewhat lower than that under the horse hypothesis. For this reason, we will perform competition among overlapping hypotheses as explained below.

Competition between overlapping hypotheses. Let z_i^C and z_j^C denote two object hypotheses with overlapping supports. To choose between them, we evaluate both hypotheses on the *union* of their supports, i.e., $\text{supp}(z_i^C) \cup \text{supp}(z_j^C)$. The unexplained features under each of the hypotheses count as background. We define the cost of a hypothesis with respect to the other as

$$\begin{aligned} \mathcal{C}(z_i^C | z_j^C) = & |\operatorname{supp}(z_i^C)| \cdot \log q(I, z_i^C) + \\ & |\operatorname{supp}(z_j^C) \setminus \operatorname{supp}(z_i^C)| \cdot \log \alpha. \end{aligned} \tag{5.16}$$

The cost weights each feature in the support of a hypothesis z_i^C by the log-likelihood $q(I, z_i^C)$ and the background features by $\log \alpha$. The hypothesis with the lower cost is selected as an object detection. The selection process is run in a greedy fashion until all overlapping hypotheses are accounted for.

A *parse tree* (most probable tree of hidden states) for each of the object detections z^C is obtained by tracing back the edges in $\mathcal{G}$, starting from the root node z^C down to the leaves. A few examples of parse trees are shown in Figure 6.15 in Section 6.2.2. The *segmentation* of the object is defined by the leaves of the tree, i.e., the support of the state z^C.

5.3 Learning

This section addresses the problem of learning the representation, which entails the following:

1. **Learning the vocabulary of compositions.** Finding an appropriate set of compositions to represent our data and learning the structure of each of the compositions (the number of parts and a (rough) initialization of the parameters).

2. **Learning the parameters of the representation.** Finding parameters of geometry and appearance for each composition as well as the thresholds to be used in the approximate inference algorithm.

The objective of learning will be to maximize the likelihood of the data. Since during recognition we are interested in finding the compositions that best explain the image in terms of likelihood, this objective is well suited for learning. Not to overfit the data, we will instead use a likelihood with a penalty for the complexity of the representation. Further, due to the exponential complexity of the unsupervised structure learning problem, our strategy will be to learn one layer at a time, in a bottom-up fashion. Specifically, at each step, we will assume that layers from 1 to $\ell-1$ have been learned and are held fixed during learning of layer ℓ.

The problem thus reduces to learning a vocabulary at layer ℓ by maximizing the penalized log-likelihood:

$$\begin{aligned} L(D;\Omega^\ell,\Theta^\ell) &= \sum_{k=1}^{N}\sum_{i} \log p(I_k(x_i^\ell) \mid \Omega^\ell,\Theta^\ell) - \lambda^\ell \left|\Omega^\ell\right| \\ &= \sum_{k=1}^{N}\sum_{i} \log \sum_{\mathcal{T}_{ki},\omega_{ki}^\ell} p(I_k(x_i^\ell),\mathcal{T}_{ki},\omega_{ki}^\ell|\Theta^\ell) - \lambda^\ell \left|\Omega^\ell\right|, \end{aligned} \tag{5.17}$$

where λ^ℓ stands for the amount of penalty for the complexity of the vocabulary and is layer-specific. The first sum runs over all N training images (which are the training data D), while the second sum runs over all patches in an image. From here on, we will omit the second sum for simplicity and assume that each training sample is an image patch of appropriate size. The second term is a penalty term, which corresponds to the sum of the number of parts across the compositions at layer ℓ: $|\Omega^\ell| = \sum_i P_i$. Here the sum runs over all $\omega_i^\ell \in \Omega^\ell$ and P_i denotes the number of parts of a composition ω_i^ℓ. We will thus prefer structurally simpler compositions with high repeatability and descriptive power.

We will learn the vocabulary with a MCMC (Monte Chain Monte Carlo) approach. Specifically, the objective in (5.17) will be used as a *scoring* function to compare

different *proposals* for the vocabularies. At each step of the MCMC iteration, we will learn a proposal vocabulary at layer ℓ, use EM (the Expectation Maximization algorithm) to learn the parameters and calculate the score of the learned vocabulary. Finally, the vocabulary with the highest score will be selected.

We will first assume that the structure of the representation is already known and show how to estimate the parameters using the EM algorithm. In Section 5.3.2 we propose how to learn also the structure of the vocabulary.

5.3.1 Learning the parameters

Let us for now assume that the structure (the number and the structure of the compositions) up to layer ℓ is given. We can employ the EM algorithm to estimate the parameters for each of the compositions at layer ℓ of the vocabulary. Since the model from one layer to the next is similar to the constellation model [157], we will adopt some of their derivations.

Instead of directly maximizing the likelihood in (5.17), EM iteratively re-estimates the parameters by maximizing the cost functions:

$$Q(\hat{\Theta}^\ell|\Theta^\ell) = \sum_k E\big[\log p(I_k(x_k^\ell), \mathcal{T}_k, \omega_k^\ell|\hat{\Theta}^\ell)\big] \tag{5.18}$$

Following [157], differentiating Q with respect to each of the parameters and setting the derivatives to 0 gives the following updates for the parameters, which constitute the M step of the EM algorithm:

$$\begin{aligned}
\hat{\mu}_{\omega^\ell,j} &= \frac{\sum_k p(\omega^\ell|I_k)\, E_{\omega^\ell,j}[q_k]}{\sum_k p(\omega^\ell|I_k)} \\
\hat{\Sigma}_{\omega^\ell,j} &= \frac{\sum_k p(\omega^\ell|I_k)\, E_{\omega^\ell,j}\big[q_k\, q_k^T\big]}{\sum_k p(\omega^\ell|I_k)} - \hat{\mu}_{\omega^\ell,j}\, \hat{\mu}_{\omega^\ell,j}^T \\
\hat{p}(\omega^\ell) &= \frac{1}{N} \sum_k p(\omega^\ell|I_k)
\end{aligned}$$

where $q_k = x_{kj}^\ell - x_k^\ell$. The expectation $E_{\omega^\ell}[\cdot]$ is taken with respect to the posterior density $p(\mathcal{T}_k|I_k, \omega^\ell, \Theta^\ell)$.

In the E-step we need to compute the sufficient statistics $E_{\omega^\ell,j}[q]$, $E_{\omega^\ell,j}[q\, q^T]$ and the posterior density

$$p(\omega^\ell|I_k) = \frac{p(I_k|\omega^\ell)p(\omega^\ell)}{\sum_i p(I_k|\omega_i^\ell)p(\omega_i^\ell)}, \tag{5.19}$$

where the likelihood $p(I_k|\omega^\ell)$ is calculated as described in Section 5.2.2.

We only still need to infer the parameters for the first layer models in the vocabulary. This is done by estimating the parameters (μ_i^1, Σ_i^1) of a multivariate Gaussian

distribution for each model ω_i^1: Each Gabor feature vector $\mathbf{f}$ is first normalized to have the dominant orientation equal to 1. All features $\mathbf{f}$ that have the i-th dimension equal to 1 are then used as the training examples for estimating the parameters (μ_i^1, Σ_i^1).

5.3.2 Learning the structure

Here we explain how we learn the proposal set of compositions, their structure and rough initializations of the parameters (which is important for initializing EM). We will assume that for each training image I we have the inference graph $\mathcal{G}_I = (Z^1 \cup \cdots \cup Z^{\ell-1}, E)$ built up to layer $\ell - 1$, which can be obtained as described in Section 5.2.2.

Our learning strategy will be the following:

1. First learn the geometry distributions between all possible pairs of compositions from layer $\ell - 1$.

2. Detect the modes in these distributions which will define two-part compositions called *duplets*.

3. Find a set of compositions, where each composition is a set of (frequently) co-occurring duplets.

4. Greedy selection of compositions from the set obtained in (3) and re-learn the parameters of the obtained selection using the EM algorithm. Use the MCMC approach to stochastically search for the vocabulary with the optimal score (5.17).

The overall learning procedure is illustrated in Figure 5.7.

Learning spatial correlations between parts

We commence by learning the spatial constraints among the *pairs* of compositions $(\omega_i^{\ell-1}, \omega_j^{\ell-1}) \in \Omega^{\ell-1} \times \Omega^{\ell-1}$. The first composition $\omega_i^{\ell-1}$ in the pair plays the role of a *reference*. This means that the spatial constraints will be learned relative to it. We further limit the relative distances between the composition pairs to be at most r^ℓ (the choice of r^ℓ is described below). We learn the spatial constraints by means of two-dimensional histograms $h_{ij}^\ell : [-r^\ell, r^\ell]^2 \to \mathbb{R}$.

During training, each histogram h_{ij}^ℓ is updated at $\bar{x}^{\ell-1} - x^{\ell-1}$ for each pair of hidden states $(z^{\ell-1}, \bar{z}^{\ell-1})$, where $z^{\ell-1} = (\omega_i^{\ell-1}, x^{\ell-1})$ and $\bar{z}^{\ell-1} = (\omega_j^{\ell-1}, \bar{x}^{\ell-1})$, satisfying:

1. $|x^{\ell-1} - \bar{x}^{\ell-1}| \leq r^\ell$, that is, $\bar{z}^{\ell-1}$ lies in the circular area around z with radius r^ℓ.

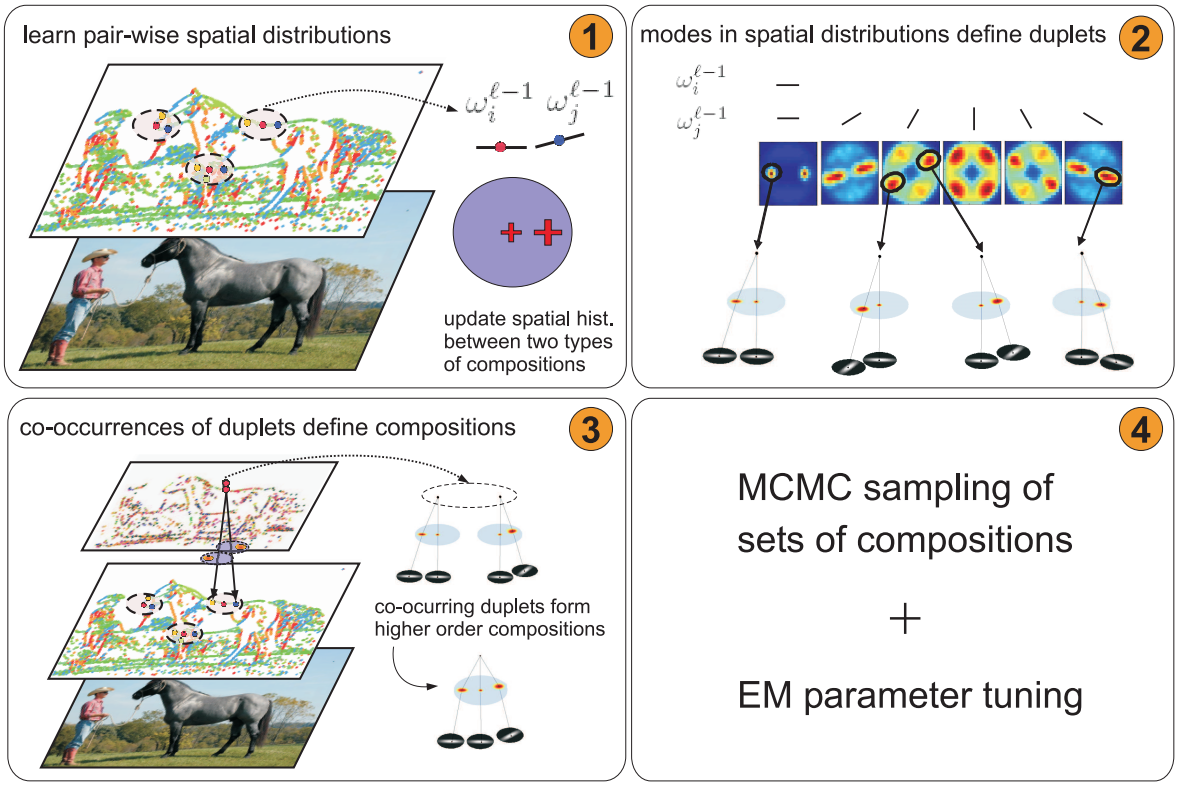


Figure 5.7: Approach to learning the structure and the parameters of the vocabulary at a particular layer.

2. $z^{\ell-1}$ and $\bar{z}^{\ell-1}$ have sufficiently disjoint supports. This is due to our assumption that the compositions have a tree topology, i.e. none of the parts in each composition spatially overlap (or rather, the probability that two parts overlap is small). The overlap of the supports is calculated as $o(z^{\ell-1}, \bar{z}^{\ell-1}) = \frac{|\text{supp}(z^{\ell-1}) \cap \text{supp}(\bar{z}^{\ell-1})|}{|\text{supp}(z^{\ell-1}) \cup \text{supp}(\bar{z}^{\ell-1})|}$. We allow for a small overlap of the parts, $o(z^{\ell-1}, \bar{z}^{\ell-1}) < 0.2$.

The histograms are updated for all inference graphs $\mathcal{G}_I$ and all the admissible pairs of hypotheses at layer $Z^{\ell-1}$.

The choice of r^ℓ. The correlation between the relative positions of the hypotheses is the highest at relatively small distances as depicted in Figure 5.9. Depending on the factors of the spatial contraction, the radii r^ℓ in this thesis are set to 8 for layer 2, 12 for the higher layers, and 15 for the final, object layer. Note that the circular regions are defined at spatial resolution of layer $\ell - 1$ which means that the area, if projected back to the original resolution, covers a much larger area.

Learning compositions of parts

The learned pair-wise spatial distributions are then *clustered* into *duplets* via the distribution modes. Figure 5.8 illustrates the clustering procedure. For each mode we estimate the mean μ^ℓ and variance Σ^ℓ by fitting a Gaussian distribution around it. Each of these modes thus results in a two-part composition with initial parameters: $P = 2$, ${\theta_{app}}_1^\ell(\omega_i^\ell) = 1$, ${\theta_{app}}_2^\ell(\omega_j^\ell) = 1$ and $\theta_{g2}^\ell = (\mu^\ell, \Sigma^\ell)$ (the first, reference part, in a composition is always assigned the default parameters as explained in Section 5.1).

We can now perform inference with the complete set of duplets. Define a duplet co-occurrence for a particular training image patch I_k as a set of duplets that best explain I_k (the likelihood of I_k defined by (5.10) is the highest for this particular set of duplets) and forms a tree. A duplet co-occurrence forms a composition and we will denote it by $\hat{\omega}^\ell$. During training, a list Ω_*^ℓ of all possible co-occurrences of duplets is kept. Note that for each composition in Ω_*^ℓ we also have the (rough) parameters of the distribution (previous paragraph), Θ_*^ℓ. Each $\hat{\omega}^\ell$ is additionally represented by a "count" value $f(\hat{\omega}^\ell)$ which is taken to be the value of the log-likelihood. Thus, in training, for each patch I_k the count of the corresponding $\hat{\omega}^\ell$ is updated accordingly, $f(\hat{\omega}^\ell) = f(\hat{\omega}^\ell) + \log q(I_k, \hat{\omega}^\ell)$.

Among Ω_*^ℓ we need to select a set of compositions (a vocabulary) yielding the highest score in (5.17). We adopt a greedy approach and further refine it by the MCMC algorithm. Let Ω_g^ℓ denote the current set of compositions selected by the greedy algorithm. At each step of the iteration, we do the following: for each patch I_k we update the count for the composition $\hat{\omega}^\ell \in \Omega_*^\ell \setminus \Omega_g^\ell$ that is most likely to have generated the patch (has the highest likelihood), by the difference in its likelihood and the likelihood under the compositions from $\hat{\omega}^\ell \in \Omega_*^\ell \setminus \Omega_g^\ell$: $f(\hat{\omega}^\ell) = f(\hat{\omega}^\ell) + \log q(I_k, \hat{\omega}^\ell) - \max_{\hat{\omega}_i^\ell \in \Omega_g^\ell} \log q(I_k, \hat{\omega}_i^\ell)$. After accounting for all the patches, we select the composition with the highest count.

We next employ a stochastic MCMC algorithm to get the final vocabulary Ω^ℓ. The first state of the Markov chain is the vocabulary Ω_g^ℓ obtained with the greedy selection. Let Ω_t^ℓ denote the vocabulary at the current state of the chain. At each step, we run the EM to also get the parameters Θ_t^ℓ, where Θ_*^ℓ is always used to initialize EM. We either exchange/add/remove one composition from Ω_t^ℓ with another one from $\Omega_*^\ell \setminus \Omega_t^\ell$ to get the vocabulary Ω_{t+1}^ℓ. The vocabulary Ω_{t+1}^ℓ is accepted as the next state of the Markov chain with probability

$$\min(1,\ \beta^{L(D;\Omega_{t+1}^\ell,\Theta_{t+1}^\ell) - L(D;\Omega_t^\ell,\Theta_t^\ell)}), \quad \beta > 1 \tag{5.20}$$

according to the Metropolis-Hastings algorithm.

The vocabulary at layer ℓ is finally defined as the Ω^ℓ with maximal value $L(D; \Omega^\ell, \Theta^\ell)$, after running several iterations of the M-H algorithm. We usually perform 100–200

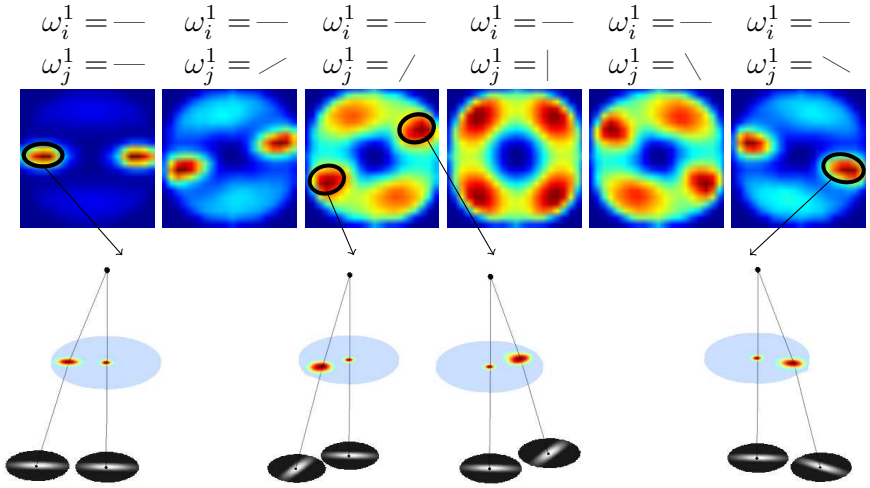


Figure 5.8: **Top**: Examples of learned histograms $h_{ij}^{\ell=2}$. **Bottom**: Examples of duplets defined by modes in the spatial histograms.

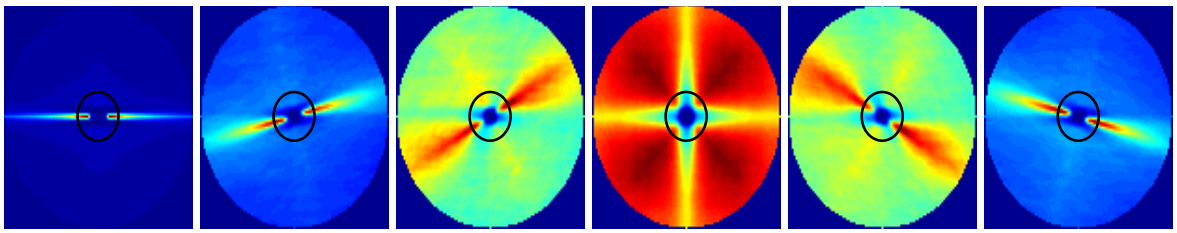


Figure 5.9: Examples of learned histograms $h_{ij}^{\ell=2}$ for very large radius, $r^\ell = 50$. The circle shows the chosen radius, $r^\ell = 8$.

steps.

To train the top, object-layer of compositions, we make use of the available supervision, similarly as in the constellation model [157]. During the greedy selection of the compositions, we always select the one with the highest detection accuracy (we use the F-measure).

5.3.3 Learning a multi-class vocabulary

We learn multiple object classes with the following strategy. Learning of the lower layers which are more generic is performed over the images of all classes (thus maximizing sharing between them). Moreover, we prefer to learn the lower layers from a set of natural images, which avoids knowing the classes in advance and results in a vocabulary which is more in tune to the statistics of natural images. Because the general, class-independent statistics of shapes becomes very sparse in the higher layers, we learn layers 4 and higher sequentially — by updating the vocabulary with the compositions

learned for one class after another. When learning each class we use the vocabulary learned so far and only *add* those compositions to each of the layers which are needed to represent the class sufficiently well.

When training the higher layers we use the bounding box information of the objects. We additionally scale the images so that the diagonal of the bounding box is approximately 250 pixels. The size of the training images has a direct influence on the number of layers being learned (due to compositionality, the number of layers needed to represent the whole shapes of the objects is logarithmic in the number of extracted edge features in the image). In order to learn a 6-layer hierarchy, the 250 pixel diagonal constraint has proven to be a good choice.

Once we have learned the multi-class vocabulary, we could, in principle, run the EM algorithm to re-learn the parameters over the complete hierarchy in a similar fashion as in the Hierarchical HMMs [57]. This way, the higher layer structure would constrain also the lower layer compositions to be learned mainly on the objects and much less on the noisy input. However, we have not done so in this thesis.

Learning the thresholds for the *tests*

Given that the object class representation is known, we can learn the thresholds $\tau(\omega^\ell)$ to be used in our (approximate) inference algorithm (Section 5.2.2).

We use a similar idea to that of Amit et al. [8] and Fleuret and Geman [61]. The main goal is to learn the thresholds in way that nothing is lost with respect to the accuracy of detection, while at the same time optimizing for the efficiency of inference.

Specifically, by running the inference algorithm on the set of class training images $\{I_k\}$, we obtain the object likelihoods $\{q(I_k, c_k)\}$, the corresponding most probable hidden trees of states $\{\mathcal{T}(I_k, c_k)\}$ and their probabilities $\{\delta(I_k, c_k)\}$. For each composition ω^ℓ in the vocabulary, we then define $\tau(\omega^\ell)$ as the minimal probability obtained in the training set:

$$\tau(\omega^\ell) = \min_k \min_{z^\ell=(\omega^\ell,x^\ell)\in\mathcal{T}(I_k,c_k)} \delta(I_k(x^\ell), \omega^\ell) \tag{5.21}$$

For a better generalization, however, one can rather take a certain percentage of the value on the right in (5.21), the idea also suggested by Fleuret and Geman [61].

5.3.4 Bottom-up and top-down optimization

The problem when optimizing the vocabulary at each layer separately (the *bottom-up phase*), is that the local solution does not necessarily lead to the globally optimal solution. A vocabulary that is well expressive but very small at one layer, may, when further combined, result in a vocabulary of lower expressiveness in the layers above,

and thus give us poor top-level models of object class shapes. We thus introduce the *top-down phase* of learning, in which we revise the vocabulary at each layer as to maximally improve the expressiveness of vocabulary at the layer above.

Top-down learning phase

The idea behind the top-down phase is to improve the vocabulary at layer ℓ in order to increase the likelihood of the data under layer $\ell+1$. This procedure is summarized as follows. *Step 1.* We first find the critical data points under the current layer $\ell+1$. *Step 2.* We re-learn layer ℓ based on the critical points. *Step 3.* We re-learn layer $\ell+1$ and repeat the process.

Step 1. We first define the critical points in each image under the current layer $\ell+1$. These are the points at which the ratio Δ of the likelihoods

$$\Delta(\mathbf{f},x)=\frac{\max_{z^\ell:(\mathbf{f},x)\in\mathrm{supp}(z^\ell)}q(I(x^\ell)|z^\ell)}{\max_{z^{\ell-1}:(\mathbf{f},x)\in\mathrm{supp}(z^{\ell-1})}q(I(x^{\ell-1})|z^{\ell-1})} \tag{5.22}$$

is high. This ratio tells us how well an observation $(\mathbf{f},x)$ is explained under the vocabulary $\Omega^{\ell+1}$ relative to how well it is explained under Ω^ℓ. Note that $\Delta\in[0,1]$. From $(\mathbf{F},\mathbf{X})$ we select the points according to the distribution $1-\Delta$, i.e. each point $(\mathbf{f},x)\in(\mathbf{F},\mathbf{X})$ is sampled with probability $1-\Delta(\mathbf{f},x)$. This is a heuristics that has worked well in our experiments. The selected points are used as the data D' for learning new shapes at layer ℓ. If $\Delta(\mathbf{f},x)=0$, it means that $(\mathbf{f},x)$ is not explained under the current vocabulary at layer $\ell+1$ and will be added to D' with probability 1. On the other hand, $\Delta(\mathbf{f},x)=1$ means that $(\mathbf{f},x)$ is explained equally well by layer ℓ and $\ell+1$, and this point will not be used in the re-learning process.

Step 2. In this step, we learn a new set of compositions Ω^ℓ_* that maximizes the likelihood function over the new data D'. We further perform the greedy and the stochastic selection (as described in the section 5.3.2) on the joint vocabulary $\Omega^\ell\cup\Omega^\ell_*$ over the original data D. When running the selection algorithm, we slightly decrease the value of λ^ℓ (only when evaluating layer ℓ) to allow the algorithm to select the shapes also from Ω^ℓ_*. This means that the final vocabulary at layer ℓ may become somewhat redundant — the complexity term in (5.17) at layer $\ell+1$ will increase, however, the likelihood term in (5.17) will also increase, potentially improving the score L of layer $\ell+1$.

Step 3. Based on the new layer ℓ we re-learn the vocabulary at layer $\ell+1$. The complete top-down phase is repeated, stochastically, several times and the best scoring vocabulary $\Omega^{\ell:\ell+1}$ is finally chosen.

In principle, we could optimize more than two layers simultaneously, however, the two-consecutive-layer optimization has turned out to be sufficient in our experiments.

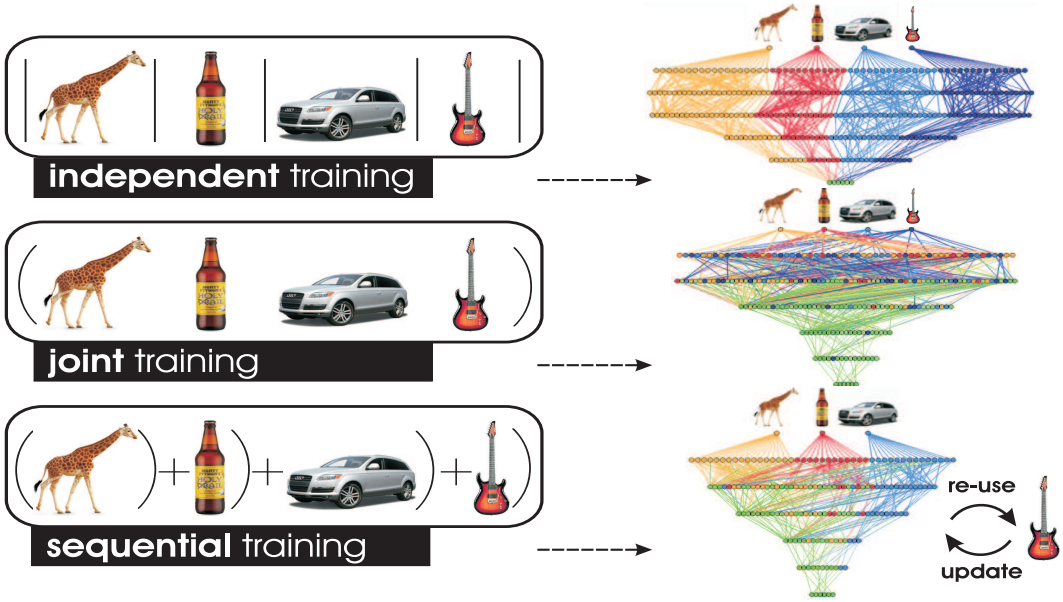


Figure 5.10: Learning strategies in our hierarchical framework. The green nodes on the right denote 1.) shareable features in joint training, and 2.) re-used features in sequential training.

5.4 Multi-class learning strategies

To learn and represent multiple object classes there have mainly been two strategies: the detectors for each class have either been trained in isolation, or trained on all the classes simultaneously. Both exert certain advantages and disadvantages. Training independently allows us to apply complex probabilistic models that use a significant amount of class specific features and allows us to tune the parameters for each class separately. For object class detection, these approaches had notable success [88]. However, representing multiple classes in this way, means stacking together specific class representations. This, on the one hand, implies that each novel class can be added in constant time, however, the representation grows clearly linearly with the number of classes and is thus also linear in inference. On the other hand, joint representations enlarge sublinearly by virtue of sharing the features among several object classes [146, 111]. This means sharing common computations and increasing the speed of the joint detector. Training, however, is usually quadratic in the number of classes. Furthermore, adding just one more class forces us to re-train the representation altogether.

Receiving somewhat less attention, the strategy to learn the classes sequentially (but not independently) potentially enjoys the traits of both learning types [111, 41, 84]. By learning one class after another, we can transfer the knowledge acquired so far to novel classes and thus likely achieve both, sublinearity in inference and cut down training

time.

In literature, the approaches have mainly used one of these three learning strategies in isolation. To the best of our knowledge, little research has been done on analyzing and comparing them with respect to one another. This is important because it allows us to point to losses and gains of each particular learning setting, which could focus further research and improve the performance. This is exactly what we are set to do — we present a hierarchical framework within which **all** of the aforementioned learning strategies can be unbiasedly evaluated and put into perspective.

Prominent work on these issues has been done in the domain of flat representations [111, 146], where each class is modeled as an immediate aggregate of local features. However, there is an increasing literature consensus, that hierarchies provide a more suitable form of multi-class representation [145, 166, 119, 153, 135]. Hierarchies not only share complex object parts among similar classes, but can re-use features at several levels of granularity also for dissimilar objects.

In this thesis, we provide a rigorous experimental evaluation of several important multi-class learning strategies for object detection within a *generative, hierarchical framework*. We build on the unsupervised structure learning approach by [55]. Here we propose and evaluate three types of multi-class learning: 1.) independent training of individual categories, 2.) joint training, and 3.) sequential learning of classes. Several issues were evaluated on multiple object classes: 1.) growth of representation, 2.) training and 3.) inference time, 4.) degree of feature sharing and re-use at each level of the hierarchy, 5.) influence of class ordering in sequential learning, and 6.) detection performance, all as a function of the number of classes learned. We demonstrate a highly logarithmic computational behavior of both, joint and sequential class training strategies that significantly outperforms previously reported results in the domain of flat representations [111].

5.4.1 Independent training of individual object classes

In independent training, a class specific vocabulary $\mathcal{V}_c$ is learned using the training images of each particular class $C = c$. We learn $\mathcal{V}_c$ by maximizing 5.17 over the data $D = \{(\mathbf{F}_n, \mathbf{X}_n, C = c)\}$. For the negative images in the validation step, we randomly sample images from other classes. The joint multi-class representation $\mathcal{V}$ is then obtained by stacking the class specific vocabularies $\mathcal{V}_c$ together, $\mathcal{V}^\ell = \cup_c \mathcal{V}_c^\ell$ (the edges E are added accordingly). Note that $\mathcal{V}_c^1$ is the only layer common to all classes (6 oriented contour fragments), thus $\mathcal{V}^1 = \mathcal{V}_c^1$. Even if some classes use similar or even structurally identical features in the higher layers, they are stored as separate nodes in the vocabulary.

5.4.2 Joint training of object classes

In joint training, the learning phase has two steps. In the first step, the training data D for all the classes is presented to the algorithm simultaneously, and is treated as unlabeled. The spatial parameters Θ of the compositions at each layer are then inferred from images of all classes, and will code "average" spatial part dispositions. The joint statistics also influences the structure of the models by preferring those that are most repeatable over the classes. This way, the jointly learned vocabulary $\mathcal{V}$ will be the best trade-off between the likelihood L and the complexity T over *all* the classes in the dataset. However, the final, top-level likelihood for each particular class could be low because the more discriminative class-specific information has been lost. Thus, we employ a second step which revisits each class separately. Here, we use the joint vocabulary $\mathcal{V}$ and *add* new compositions ω^ℓ to each layer ℓ if they further increase the score in 5.17 for each particular class. This procedure is similar to that used in sequential training and will be explained in more detail in the following subsection. Object layer $\mathcal{V}^\mathcal{O}$ is consequently learned and added to $\mathcal{V}$ for each class. We validate the object models after all classes have been trained. A similarity measure is used to compare every two classes based on the degree of feature sharing between them. In validation, we choose the negative images by sampling the images of the classes according to the distribution defined by the similarity measure. This way, we discard the models that poorly discriminate between the similar classes.

5.4.3 Sequential training of object classes

When training the classes sequentially, we train on each class separately, however, our aim is to 1.) maximize the re-use of compositions learned for the previous classes, and 2.) add those missing (class-specific) compositions that are needed to represent class k sufficiently well. Let $\mathcal{V}_{1:k-1}$ denote the vocabulary learned for classes 1 to $k-1$. To learn a novel class k, for each layer ℓ we seek a new set of compositions that maximizes the score function in 5.17 over the data $D = \{(\mathbf{F}_n, \mathbf{X}_n, C = k)\}$ conditionally on the already learned vocabulary $\mathcal{V}^\ell_{1:k-1}$. This is done by treating the hypotheses inferred with respect to $\mathcal{V}^\ell_{1:k-1}$ as fixed, which gives us a starting value of the score function 5.17. Each new composition ω^ℓ is then evaluated and selected conditionally on this value, i.e. such that the difference $L(D; \Omega^\ell \cup \omega^\ell, \Theta^\ell \cup \theta^\ell) - L(D; \Omega^\ell, \Theta^\ell)$ is maximized.

5.5 Using the learned intermediate features for object classification

We propose to create similarity connections between the compositions *within layers* to achieve invariance for high variability in object shape and draw similarities *across layers* to achieve a proper scale normalization of features. We show how a layer-independent description of objects defined by the so-called shape-terminals, i.e. *shapinals*, can be passed to a classifier to recognize objects based on the intermediate features (compositions of intermediate complexity).

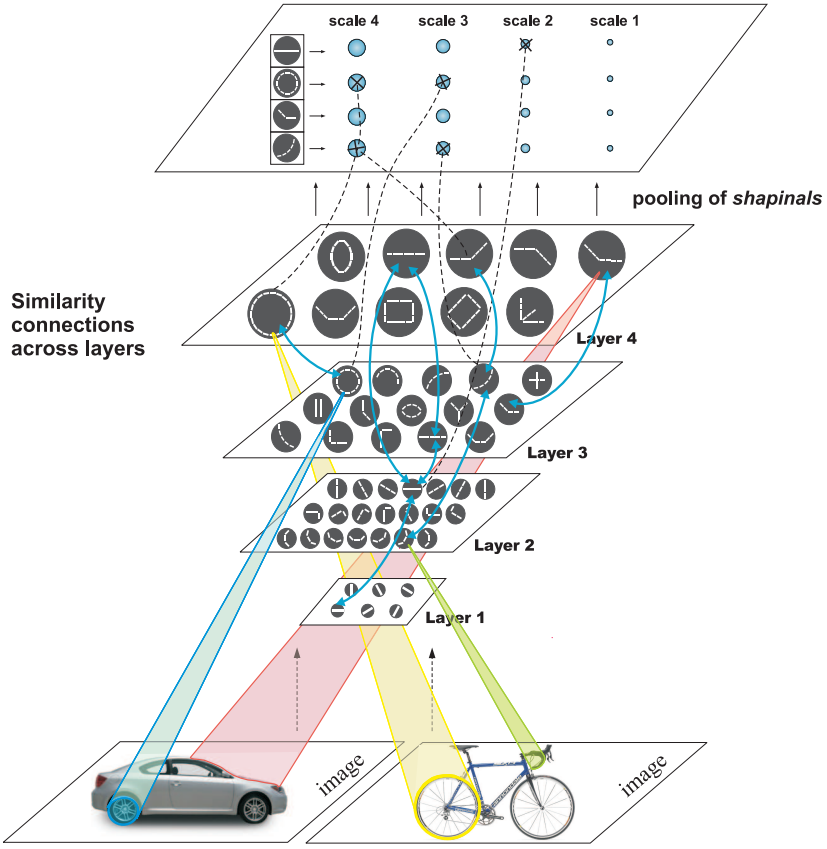


Figure 5.11: Cross-layered, scale independent representation.

5.5.1 Similarity between compositions within layers

The compositions can be learned without supervision from a set of images as described in Section 5.3. The drawback, however, is the fact that they are realized as discrete labels (compositions types) without a proper geometrical parametrization that would enable a comparison between them. Consequently, two visually similar curvatures

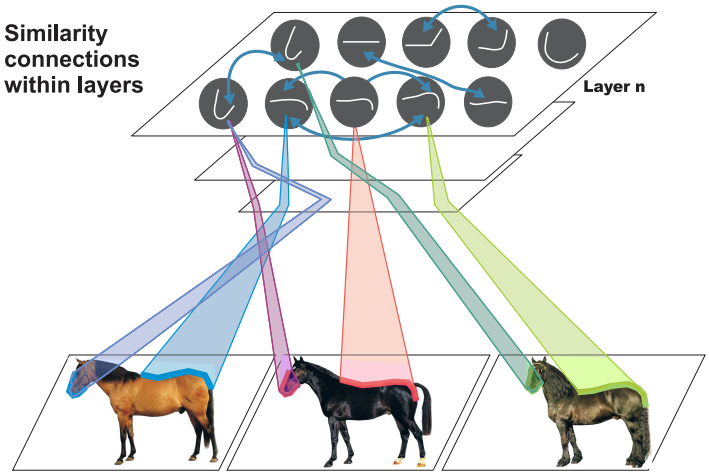


Figure 5.12: For greater generalization we establish similarities between compositions within a particular layer.

are likely to be encoded in two different compositions. Grouping (OR-ing) by co-occurrence [55] only partially solves this problem: since perceptibly equal structures can be composed in different ways, parts formed as different compositions will highly co-occur. However, two visual shapes that are only similar to a certain extent are likely to have a small, random co-occurrence. It is thus crucial to have the means of comparing two different compositions in a geometrical sense.

In a strictly mathematical sense, the optimal similarity between two hierarchically formed shapes encoding statistically learned spatial variations, would be to draw samples from the distribution of spatial variance for each part, with the process repeated down to the original Layer 1. Each sampled composition would thus take the form of a number of shape-forming points. Here-on, two compositions could be compared using standard shape similarity techniques. The final similarity value would be the average similarity calculated over a number of sampled versions of two compared compositions. However, due to a larger number of compositions (in the order of a few hundred) in the higher hierarchical layers, this method would be computationally unfeasible.

Since each composition is defined as a recursive geometric combination of simpler parts, a comparison should be performed in a similar manner. We consider two compositions to be perceptually similar if both have a similar spatial configuration of parts. However, the compatibility of spatial variations of parts within two compositions that come directly from the adjacent lower layer should contribute more to the similarity function than the spatial variations encoded in their subparts which code much smaller and simpler shapes.

We thus define a *similarity* measure $\text{sim}_{n,\ell}(\omega_i^\ell, \omega_j^\ell)$ between compositions ω_i^ℓ and ω_j^ℓ,

both from Layer ℓ, with respect to Layer n recursively as follows.

$$\mathrm{sim}_{n,\ell}(\omega_i^\ell, \omega_j^\ell) = \min\big\{\mathrm{sim}'_{n,\ell}(\omega_i^\ell, \omega_j^\ell), \mathrm{sim}'_{n,\ell}(\omega_i^\ell, \omega_j^\ell)\big\}$$

for $1 < \ell \leq n$ and

$$\mathrm{sim}_{n,1}(\omega_i^1, \omega_j^1) = \rho\big(\mathcal{R}_{f_{n,1}}(\mathit{filter}_i), \mathcal{R}_{f_{n,1}}(\mathit{filter}_j)\big)$$

otherwise, where

$$\mathrm{sim}'_{n,\ell}(\omega_i^\ell, \omega_j^\ell) = M\big(\mathrm{psim}_{n,\ell}(\omega_{i_1}^{\ell-1}, \omega_j^\ell), \ldots, \mathrm{psim}_{n,\ell}(\omega_{i_m}^{\ell-1}, \omega_j^\ell)\big)\,,$$

and

$$\mathrm{psim}_{n,\ell}(\omega_{i_t}^{\ell-1}, \omega_j^\ell) = \max_{j_l}\big\{\mathrm{sim}_{n,\ell-1}(\omega_{i_t}^{\ell-1}, \omega_{j_l}^{\ell-1}) \cdot \rho\big(\mathcal{R}_{f_{n,\ell}}(\mathit{map}_{i_t}), \mathcal{R}_{f_{n,\ell}}(\mathit{map}_{j_l})\big)\big\}\,.$$

Here $\rho(A, B) = \frac{A \cdot B}{\|A\| \cdot \|B\|}$ is a measure for the similarity between maps (geometric distributions) A and B ($A \cdot B$ denotes a component-wise inner product of two matrices and $\|A\|$ is a norm induced by this product). Note that $\rho(A, A) = 1$ and $\rho(A, B) = 0$ when the supports of A and B are disjoint.

Further, $\mathcal{R}_{f_{n,\ell}}(\cdot)$ denotes resampling by a factor $f_{n,\ell}$, where $f_{n,\ell}$ refers to the quotient of the receptive field sizes of layers ℓ and n, respectively. Since the receptive field sizes of subparts relative to part from Layer n reduce by factor 2, we take $f_{n,\ell} = 0.5^{n-\ell}$. By resampling the spatial maps of lower layers, we are weighting down the influences that the lower layer subparts have on the final similarity calculation. Thus, from the perspective of Layer 4 for example, the different orientations of the filters that compose a certain part down at the base, Layer 1 level, become more alike and virtually an unimportant factor in the similarity calculation.

Next, we observe that $\mathrm{psim}_{n,\ell}(\omega_{i_t}^{\ell-1}, \omega_j^\ell)$ gives a similarity between $\omega_{i_t}^{\ell-1}$, a subpart of ω_i^ℓ, and the best matching subpart of ω_j^ℓ. For $\mathrm{sim}'_{n,\ell}(\omega_i^\ell, \omega_j^\ell)$ we would like to give an *average* similarity between the subparts of ω_i^ℓ and their best matched subparts of ω_j^ℓ. To do this, we take:

$$M_{\mathrm{avg}}(x_1, x_2, \ldots, x_m) = \frac{1}{m}\sum_{i=1}^{m} x_i,$$

$$M(x_1, x_2, \ldots, x_m) = \begin{cases} 0, \text{ if there exists } j \text{ s.t. } x_j < T, \\ M_{\mathrm{avg}}(x_1, x_2, \ldots, x_m), \text{ otherwise} \end{cases}$$

In the similarity function M as defined above, the similarity between two compositions, where some part cannot be matched to any part of the second part within a specified tolerance T, is set to zero. Finally, the similarity between two compositions ω_i^n and ω_j^n, both from Layer n, is obtained as $\mathrm{sim}_{n,n}(\omega_i^n, \omega_j^n)$.

The advantage of calculating the similarity in this way lies in its recursive formulation. When calculating similarities within Layer n, we can efficiently re-use the similarities calculated from the layers below. Since the majority of similarities are small on Layer $n-1$, the number of Layer n compositions that need to be compared also becomes very low.

We must emphasize, however, that all the similarity calculations are performed during an offline-stage and thereafter become a part of the hierarchical vocabulary. This information can then be used efficiently in online processing of images. Examples of repeatability of compositions across objects of the same class is depicted in Figure 5.13.

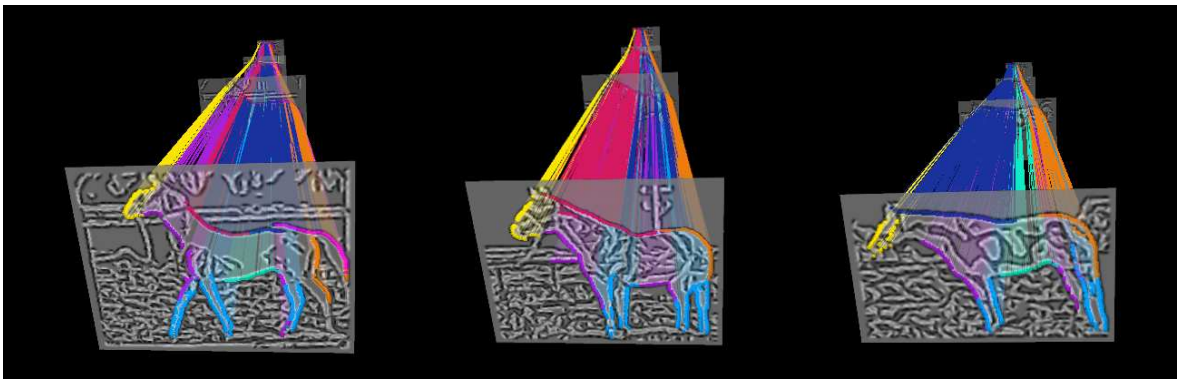


Figure 5.13: Repeatability of features at intermediate layers across a class. Different colors denote different types of compositions.

5.5.2 Similarity between compositions across layers

Comparing parts from different layers is somewhat harder, since their receptive fields are at least by a factor 2 apart. We propose to calculate the similarities in the following way. Let a similarity measure between the parts ω_i^ℓ and $\omega_j^{\ell'}$, on Layers ℓ and ℓ' with respect to Layer n, be

$$\mathrm{sim}_{n,\ell,\ell'}(\omega_i^\ell, \omega_j^{\ell'}) = M\big(\mathrm{psim}_{n,\ell,\ell'}(\omega_{i_1}^{\ell-1}, \omega_j^{\ell'}), \ldots, \mathrm{psim}_{n,\ell,\ell'}(\omega_{i_m}^{\ell-1}, \omega_j^{\ell'})\big) \tag{5.23}$$

for $1 < \ell' < \ell \leq n$, and

$$\mathrm{sim}_{n,\ell,1}(\omega_i^\ell, \omega_j^1) = \rho\big(\mathcal{CR}(\omega_i^\ell), \mathcal{R}_{f_{1,\ell}}(\mathit{filter}_j)\big)\,, \tag{5.24}$$

for $1 < \ell \leq n$, where $\mathcal{CR}(\omega_i^\ell)$ denotes the reconstruction of the composition ω_i^ℓ with filters on Layer 1 and

$$\mathrm{psim}_{n,\ell,\ell'}(\omega_{i_t}^{\ell-1}, \omega_j^{\ell'}) = \max_{j_l}\big\{\mathrm{sim}_{n,\ell-1,\ell'-1}(\omega_{i_t}^{\ell-1}, \omega_{j_l}^{\ell'-1}) \cdot \rho\big(\mathcal{R}_{f_{n,\ell}}(\mathit{map}_{i_t}), \mathcal{R}_{f_{n,\ell'}}(\mathit{map}_{j_l})\big)\big\}\,.$$

In (5.23) we recursively compare each part of ω_i^{ω} with parts of $\omega_j^{\ell'}$ and taking the *average* given by a suitable function M (see Section 5.5.1). The second case (5.24) covers the calculation of similarities when we compare a composition ω_i^{ℓ} on Layer ℓ to a composition ω_j^1 on Layer 1. Here we choose to reconstruct the part ω_i^{ℓ} from the filters on Layer 1 (by recursively using relative centers of parts, (x_i, y_i)), resize the filter corresponding to ω_j^1, and calculate the correlation between the two maps.

Note that this measure, in contrast to the measure within layers defined in the previous section, is not symmetric. It does not take much effort to make it symmetric, but it would further complicate the definition.

5.5.3 Creating layer-independent labels

For each composition ω we can now make connections to all the compositions in the hierarchical vocabulary that have a similarity above a chosen threshold (we take it to be 0.5). This can be seen as organizing the compositions topologically; the connections and their strength define a similarity neighborhood of each composition. Such organization of compositions offers numerous advantages and brings higher robustness into the representation: whenever a certain composition is being matched in the detection stage, any composition from its defined similarity neighborhood may contribute to the final hypothesis.

Next, the layer-independent set of labels is created with a simple greedy algorithm. Firstly, each composition in the hierarchical vocabulary is assigned into a separate group. At each step, two groups are joined (are assigned the same label), if their similarities are higher than a chosen threshold. The similarity between the joined group and the rest of the groups is set as the minimum of similarities of compositions forming the group with the compositions forming the remaining groups. When no two groups of compositions are above the threshold the process ends. Groups are labeled and a mapping from the composition in the hierarchical vocabulary into the obtained layer-independent set of labels is formed: $\mathit{Cross\_layer}(\omega_i^{\ell}) = q$, where q denotes the label of the group to which a composition ω_i^k belongs.

The layer-independent labels thus equalize compositions coding lines, circles, etc. across layers of the hierarchical vocabulary.

5.5.4 Shapinals: Obtaining a cross-layered object description

From the complete list of hidden states across all layers of the inference graph $\mathcal{G}$ and a small number of scales, we extract a set of *shapinals* as follows. At each step, we select a state z_i with the highest cardinality of $\mathrm{supp}(z_i)$, corresponding to the number of image points it describes. By performing local inhibition, all the states with smaller

supports are discarded from the list of hidden states. When adding the selected state z_i to the list of shapinals, we assign it a layer-independent label, $Cross\_layer(\omega)$ and a value of *lod*, i.e. *level-of-detail*. This value codes the approximate size of the state taking into account the scale and level of hierarchy at which it was inferred. Since the receptive fields of hierarchical layers increase by a certain factor (usually 2), the notion of size is thus also encoded within the hierarchy and can naturally be combined with image scales. By sampling 2 scales per octave, level-of-detail can be calculated as $lod = \text{round}(i_{layer} + 0.5 \cdot scale)$. To give an example, a line detected at layer 2 and scale 3 will be given the same value of *lod* as a line detected at layer 3 and scale 1.

How the extracted shapinals are used for object classification is described in the section on Experimental results.

Chapter 6

Experimental results

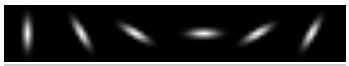
The evaluation is separated into two parts. We first show in Section 6.1 the capability of our method to learn generic shape structures from natural images and apply them to an object classification task. Second, we utilize the approach for multi-class object detection on 15 object classes in Section 6.2. We analyze the computational properties of our approach through several important aspects: time needed to train the representation, time needed to detect object classes in a given image, storage demands of the representation, ability to share and re-use features among classes at several layers of the hierarchy and detection accuracy. We additionally compare various multi-class training strategies. Preliminary results on 175 object classes are also reported.

All experiments were performed on a single core on a Intel Xeon-4 CPU 2.66 Ghz computer. The method is implemented in C++.

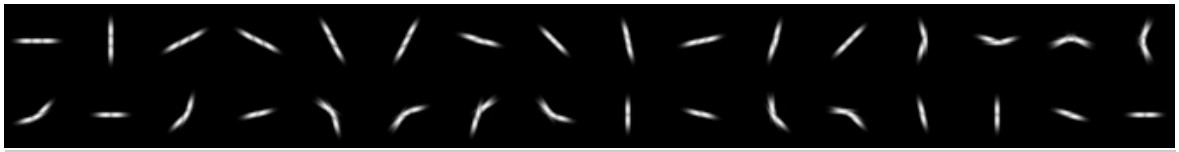
6.1 Natural statistics and object classification

We first applied our learning approach to a collection of 1500 natural images. Learning was performed only up to layer 3. The complete learning process took roughly 5 hours. The learned hierarchy consisted of 160 compositions on Layer 2 and 553 compositions on Layer 3. A few examples from both layers are depicted in Fig. 6.1 (note that the images shown are only the mean shapes generated by the generative models). The learned features include corners, end-stopped lines, various curvatures, T- and L-junctions. Interestingly, these structures resemble the ones predicted by the Gestalt theory of grouping [158]. Our statistical analysis could serve as an empirical proof to this classical theory.

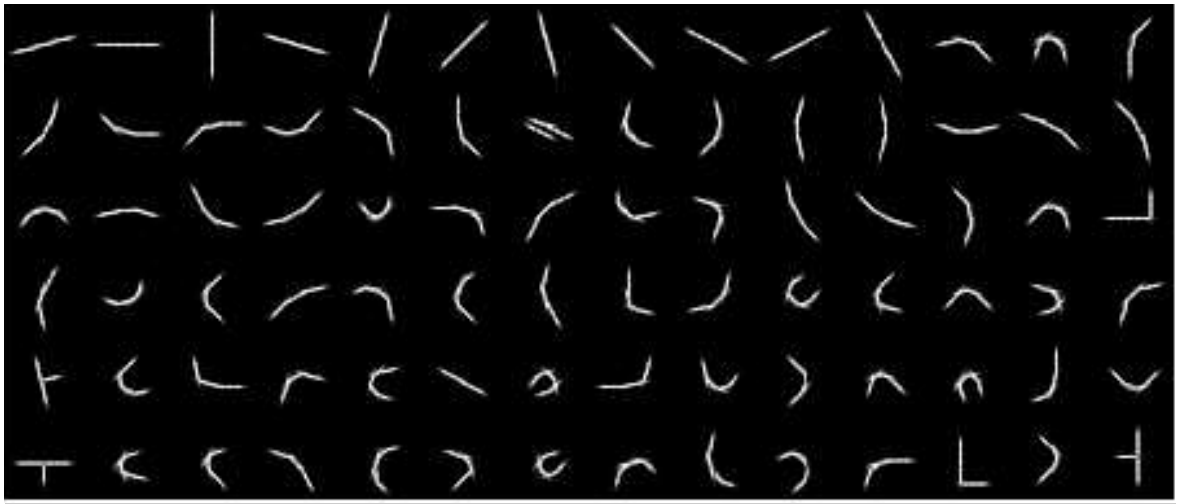
The tuning properties of a few learned compositions from Layer 2 of the vocabulary is shown in Figure 6.2.



Layer 1



Layer 2



Layer 3

Figure 6.1: A vocabulary learned on 1500 natural images (not all compositions are shown). The first layer is fixed (pre-designed). Only the mean of the compositions are depicted. The vocabulary has learned generic features such as various curvatures and corners, and L- and T- junctions.

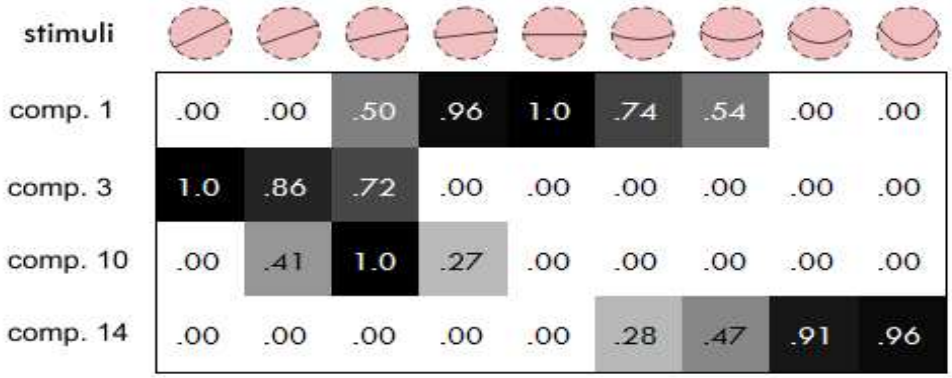


Figure 6.2: Tuning properties of compositions. A few images of curvature were generated (top row) of size equal to the size of the receptive fields of Layer 2 compositions. The Table shows the responses of the maximally active composition for a particular image.

Each image was processed at 3 scales spaced apart by $\sqrt{2}$. For the learned vocabulary features we calculated the similarities between them using the approach described in Section 5.5. Upon these similarities, the vocabulary features were grouped to produce 102 layer-independent features. Pooling of features across layers was performed as described in Section 5.5.4. These were consequently combined with a linear SVM for multi-class classification [2, 81]. The results, averaged over 8 random splits of train/test images, are reported in Table 6.1 with classification rates of other hierarchical approaches shown for comparison. For 15 and 30 training images we obtained a 60.5% and 66.5% accuracy, respectively, which is one of the best results reported by a hierarchical approach so-far. A better performance has been reported by Todorovic and Ahuja [5] (however using also color and region information) and Ahmed at al. [4] and Yu at al. [162] for the case of using 30 training examples.

We further tested the classification performance by varying the number of training examples. For testing, 50 examples were used for the classes where this was possible and less otherwise. The classification rate was normalized accordingly. In all cases, the result was averaged over 8 random splits of the data. The results are presented and compared with other classification methods in Figure 6.4. We only compare to approaches that use only the shape information. Overall, better performances have been obtained in [156, 20].

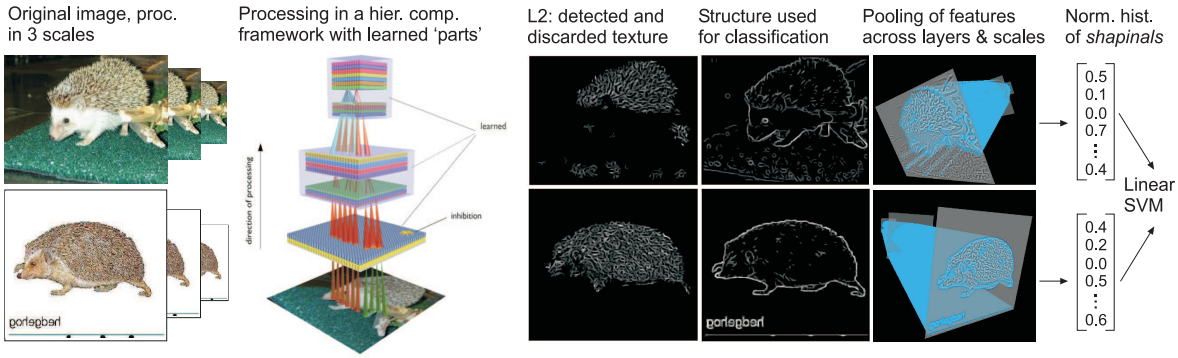


Figure 6.3: Pooling of features across layers for classification of objects in the Caltech-101 dataset.

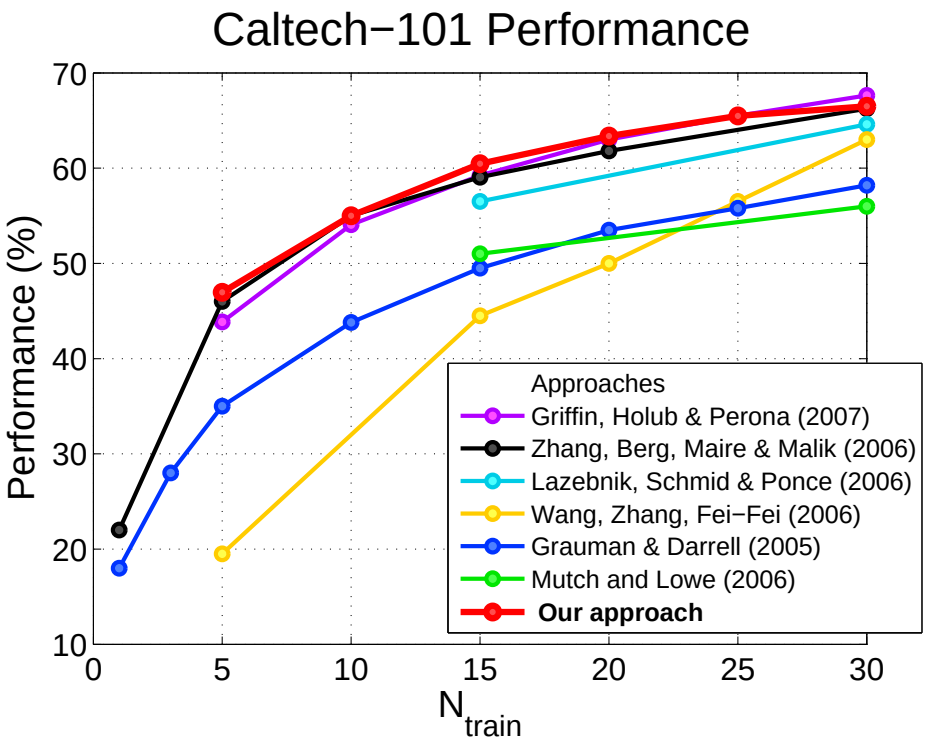
6.2 Object class detection

The approach was tested on 15 diverse object classes from standard recognition datasets. The classes used are the following: *Apple logo, bottle, giraffe, mug, swan* from the ETH dataset [45, 46], *car_side* from the multi-scale UIUC car dataset [3], *horse* from the multi-scale Weizmann dataset (adapted by Shotton et al. [133]), *motorbike* from the

Table 6.1: Average classification rate (in %) on Caltech 101.

	$N_{train} = 15$	$N_{train} = 30$
Serre et al. [130]	44	/
Mutch et al. [103]	51	56
Ranzato et al. [119]	/	54
Ommer et al. [109]	/	61.3
Wolf et al. [159]	51.18	/
Ahmed at al. [4]	58.1	67.2
Yu at al. [162]	59.2	**67.4**
Lee at al. [73]	57.7	65.4
Jarrett at al. [79]	/	65.5
our approach	**60.5**	66.5

Figure 6.4: Classification performance on Caltech-101.



TUD dataset [102, 88], and *bicycle_side, car_front, car_rear, cup, face, person* from the GRAZ dataset [111]. The basic information with respect to the number of training/test images is given in Table 6.2. These datasets are known to be challenging because they contain a high amount of clutter, multiple objects per image, great scale differences of objects and exhibit a significant intra class variability. Note, however, that most of the

objects in a class are viewed in a canonical view and there is little variation in 3D pose. Articulated objects such as horses, cows and people appear in multiple articulations.

In this, and all of the experiments to follow, we learn the complete, 6-layer object representation and do not use linear classifiers (as in the previous experiment) in any stage of processing.

Table 6.2: Information on the datasets used to evaluate the detection performance.

dataset	ETH shape					UIUC	Weizmann	INRIA	TUD
class	Apple logo	bottle	giraffe	mug	swan	car_side (m. scale)	horse (m. scale)	horse	motorbike
# train im.	19	21	30	24	15	40	20	50	50
# test im.	21	27	57	24	17	108	228	290	115

dataset	GRAZ										
class	bicycle side	bottle	car front	car rear	cow	cup	face	horse side	motorbike	mug	person
# train im.	45	/	20	50	20	16	50	30	50	/	19
# test im.	53	64	20	400	65	16	217	96	400	15	19

For training we used the bounding box information of objects or the masks if they were available. Each object was resized so that its diagonal in an image was approximately 250 pixels. Examples of training images for a few classes are shown in Figure 6.5.

For testing, we resized each test image up to a factor 3 and used from 4 to 6 scales for object detection. Two scales per octave were used. Examples of test images are shown in Figure 6.6. The test images (before scaling) are quite large, roughly of size 400×600. Multiple objects can appear in one image (and can also be of different sizes), although never from different classes. When evaluating the detection performance, a detection is counted as correct, if the predicted bounding box b_{fg} coincides with the ground truth b_{gt} more than 50% (the PASCAL criterion [39]):

$$\frac{area(b_{fg} \cap b_{gt})}{area(b_{fg} \cup b_{gt})} > 0.5.$$

On the ETH dataset and INRIA horses this threshold is lowered to 0.3 to enable a fair comparison with the related work by Ferrari et al. [45].

The performance is given either with recall at equal error rate (EER) or positive detection rate at low FPPI (false positives per image), depending on the type of results reported on these datasets thus-far. Recall is defined as the number of correctly detected objects divided by all objects in the test images. Recall at EER means that the parameters of the model are varied until the number of false positives (incorrect

detections) equals to the number of false negatives (missed detections of objects). This measure is standardly used to evaluate the detection performance, although it has the drawback of being dependent on the number of negative images (images without the object class in question): if the number of negative images in the dataset increases, the false positive rate usually increases as well and thus the overall performance decreases. As a consequence, Ferrari et al. [45, 46] introduced the measure which evaluates the detection rate at a fixed false positive rate per image, i.e. a 0.4 FPPI means that roughly every second image can have a falsely predicted object. Detection rate is the same as recall.

In test time we form the bounding box around the predicted object in the following way. For a found detection (an active hypothesis at the top, object, layer in the inference graph) we find its support (supp(z^c), i.e., all first layer hypotheses (contour fragments) which are in the inferred subgraph of the object hypothesis. A bounding box is then formed around these features.

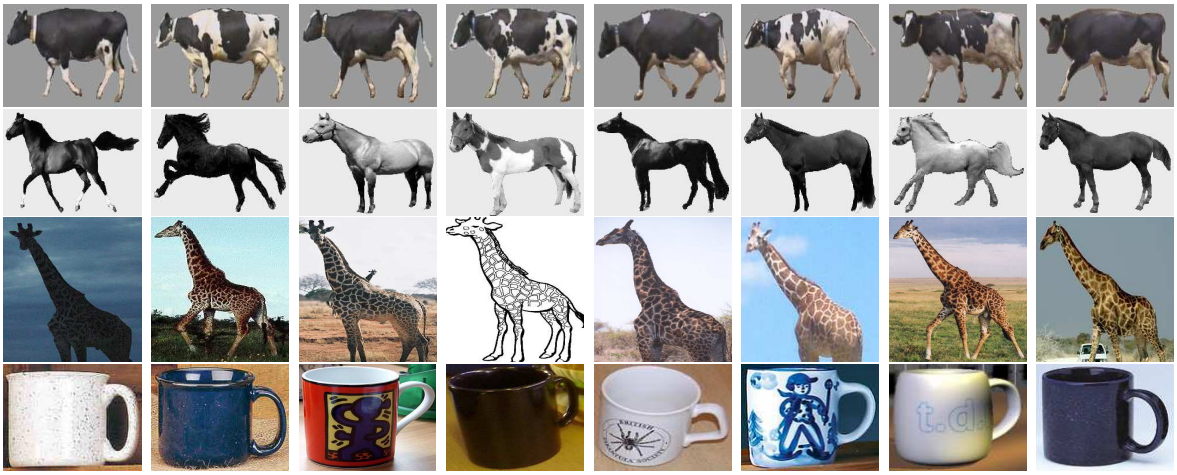


Figure 6.5: Examples of class training images (*cow, horse, giraffe, mug*). For datasets where the segmentation masks are available we use only the object pixels for training, otherwise we use the image inside the bounding box.

6.2.1 Single class learning and performance

We first evaluated our approach to learn and detect each object class individually. We report the training and inference times, and the detection performance, which will then be compared to the multi-class case in Subsection 6.2.2 to test the scalability of the approach.

Training time. To train a class it takes on average $20 - 25$ minutes. This includes the time to fully optimize the vocabulary and to learn the thresholds in the model.

Figure 6.6: Examples of test images. Top row: ETH shape dataset. Middle row: GRAZ dataset. Bottom row: Weizman horse dataset.

For example, it takes 23 minutes to train on Apple logo, 31 for giraffe, 17 for swan, 25 for cow, and 35 for the horse class. Most of the related approaches do not give the training times, so we only compare to the results of Shotton et al. [133], where training on 50 horses takes approximately 2 hours. Their experiments were conducted on 2.2 GHz machine using a C# implementation, which is similar to our experimental setup with slightly older hardware.

Inference time. Object detection using a hierarchical vocabulary learned for each individual class takes from $2-4$ seconds per image, depending on the size of the image and the amount of texture it contains. For example, it takes 3.6 seconds to process Apple logos, 3.2 seconds for giraffes, and 1.9 seconds for swans. Other related detection approaches report approximately 5 to 10 times higher times (in seconds): [145]: $20-30$, [164]: 16.9, [63]: 20, [44]: $12-18$, however, at slightly older hardware.

Detection performance. The ETH experiments are performed in a 5-fold cross-validation obtained by sampling 5 subsets of half of the class images at random. The test set for evaluating detection consists of all the remaining images in the dataset. The detection performance is given as the detection rate at the rate of 0.4 false-positives per image (FPPI), averaged over the 5 trials as in [45]. We also report the actual FPPI rate for each class. The detection performance is reported in Table 6.3. Similarly, the results for the INRIA horses are given in a 5-fold cross-validation obtained by sampling 5 subsets of 50 class images at random and using the remaining 120 for testing. The test set also includes 170 negative images to allow for a higher FPPI rate [45]. With respect to [45], we achieve a better performance for all classes, most notably for giraffes (24.7%). Our method performs comparably to a discriminative framework by Fritz and Schiele [63].

Table 6.3: Detection results. On the ETH shape and INRIA horses we report the detection-rate (in %) at 0.4 FPPI averaged over five random splits train/test data. For all the other datasets the results are reported as recall at equal-error-rate (EER).

	class	**[45]**	**[63]**	**our approach**	
ETH shape	applelogo	83.2 (1.7)	89.9 (4.5)	87.3 (2.6)	0.32 FPPI
	bottle	83.2 (7.5)	76.8 (6.1)	**86.2** (**2.8**)	**0.36** FPPI
	giraffe	58.6 (14.6)	90.5 (5.4)	83.3 (4.3)	0.21 FPPI
	mug	83.6 (8.6)	82.7 (5.1)	**84.6** (**2.3**)	**0.27** FPPI
	swan	75.4 (13.4)	84.0(8.4)	78.2 (5.4)	0.26 FPPI
	average	76.8	84.8	83.7	0.28 FPPI
INRIA	horse	84.8(2.6)	/	**85.1**(**2.2**)	**0.37** FPPI

	class	**related work**		**our approach**
UIUC	car_side, multiscale	90.6 [103]	93.5 [5]	**93.5**
Weizmann	horse_multiscale	89.0 [133]	93.0 [132]	**94.3**
TUD	motorbike	87 [88]	88 [102]	83.2

	class	**[111]**	**[133]**	**our approach**
GRAZ	face	96.4	97.2	94
	bicycle_side	72	67.9	68.5
	bottle	91	90.6	89.1
	cow	100	98.5	96.9
	cup	81.2	85	**85**
	car_front	90	70.6	76.5
	car_rear	97.7	98.2	97.5
	horse_side	91.8	93.7	**93.7**
	motorbike	95.6	99.7	93.0
	mug	93.3	90	90
	person	52.6	52.4	**60.4**

For the experiments on the rest of the datasets we report the recall at EER. The results are given in Table 6.3. Our approach achieves competitive detection rates with respect to the state-of-the-art. Note also that Mikolajczyk et al. [102] and Leibe et al. [88] used 150 training examples of motorbikes, while we only used 50 (to enable a fair comparison with Opelt et al. [111] on GRAZ).

The reason that our approach does not achieve better detection rates might be two-fold: 1.) Only contour information is used. Some object also have distinctive texture (such as giraffes) and thus modeling also texture could give a boost in performance. Second, since test images in most cases contain significant texture with which our approach does not cope well (it can give rise to many spurious detections), the detection thresholds are usually set quite high at the expense of missing a few true positives. 2.) The approach is generative and the background model is currently very simplistic. If more sophisticated background models would be used, better discrimination could be achieve against noisy, textured regions.

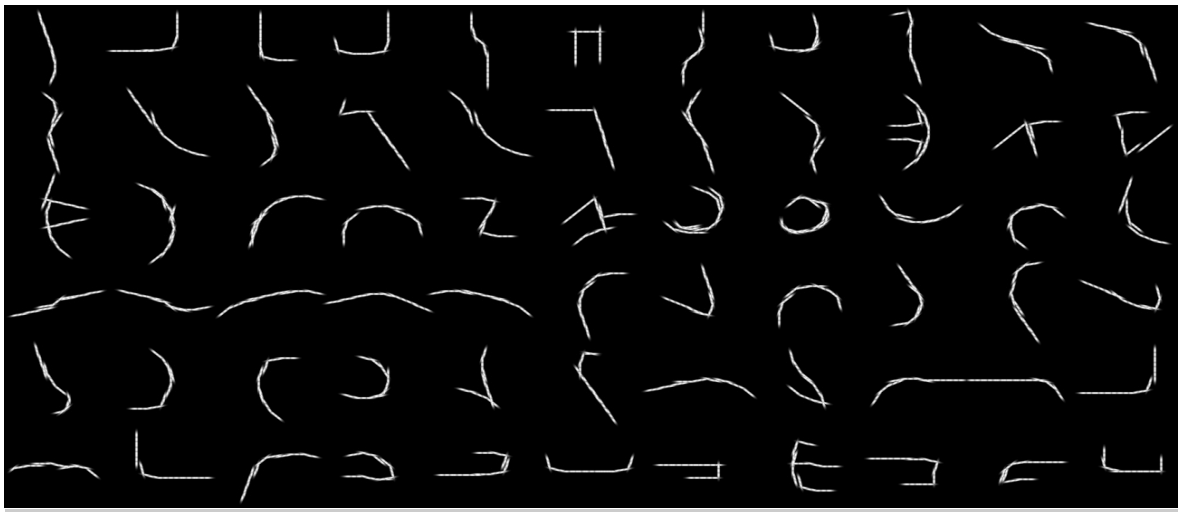
While the performance in a single-class case is comparable to the current state-of-the-art, the main advantage of our approach are its computational properties when the number of classes is higher, as demonstrated next.

6.2.2 Multi-class learning and performance

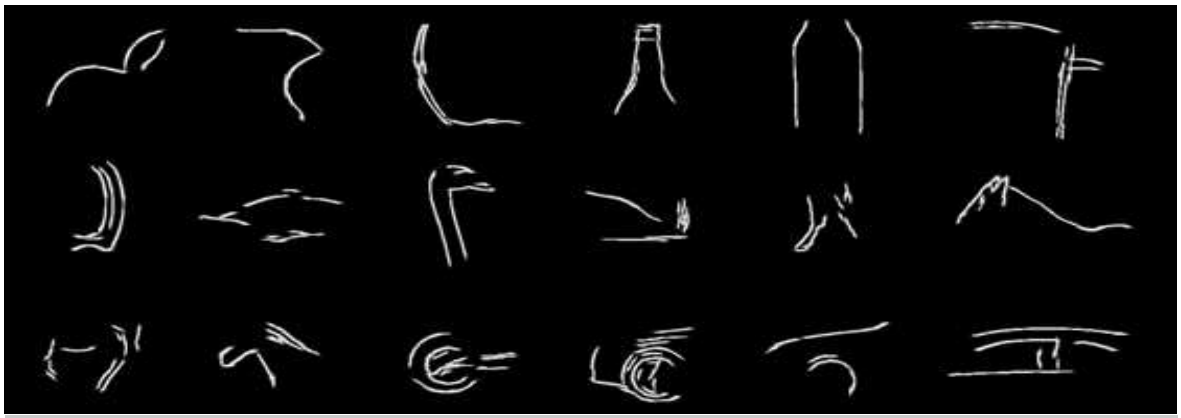
To evaluate the scaling behavior of our approach we have incrementally learned 15 classes one after another.

The learned vocabulary. A few examples of the learned shapes at layers 4 to 6 are shown in Fig.6.7 (only samples from the generative model are depicted). An example of a complete object-layer composition is depicted in Fig. 6.8. It can be seen that the approach has learned the essential structure of the class well, not missing any important shape information. However, some of the details that might be necessary to distinguish similar classes, may be missing. This is because the last layer (and each in between) is only connected to the previous one, while the smaller, more distinctive features should likely be collected from the lower layers. Incorporating connections across several layers will be part of future work.

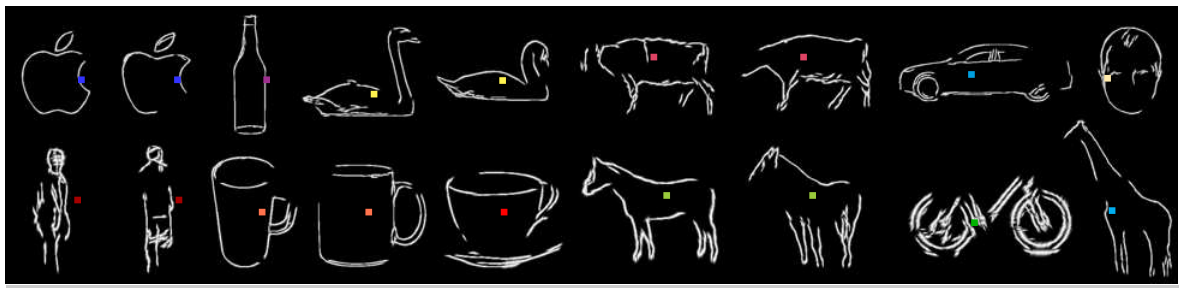
Degree of composition sharing among classes. To see to what extent are the compositions in the vocabulary shared between classes of different degrees of similarity, we depict the learned hierarchies for two visually similar classes (motorbike and bicycle), two semi-similar classes (giraffe and horse), and two dissimilar classes (swan and car_front) in Figure 6.9. The nodes denote the compositions and the links denote the edges of the vocabulary (the spatial relations between the compositions). The green nodes represent the compositions used by both classes, while the specific colors denote



Layer 4



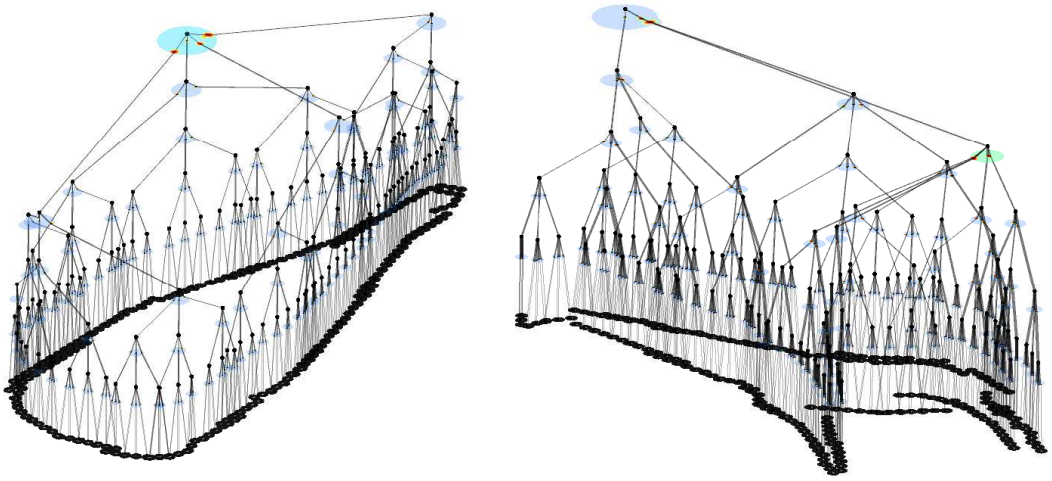
Layer 5



Layer $\mathcal{O} = 6$ - object layer

Apple logo | bottle | swan | cow | car | face
person | mug | cup | horse | bicycle | giraffe

Figure 6.7: Examples of compositions at layers 4, 5, and the final object layer learned for 15 classes for object detection.



(a) (b)

Figure 6.8: Example of one learned object-layer composition for (a) bottle and (b) giraffe, with the complete model structure shown. The blue patches denote the limiting circular regions, the red patches inside them depict the spatial relations (and their variance). The nodes are the nodes in the vocabulary, i.e., the compositions.

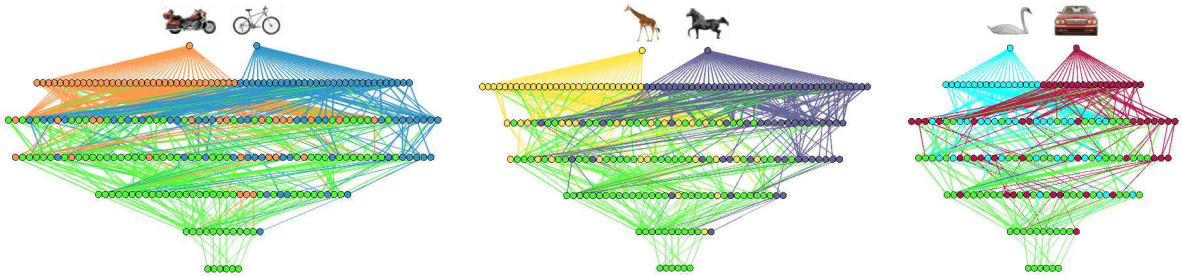
class specific compositions. If a spatial relation is shared as well, the edge is also colored green. The computational advantage of our hierarchical representation over flat vocabularies [111, 88, 133, 146] is that the compositions are highly shared at *several* layers of the vocabulary, which significantly reduces the overall complexity of inference. Each shared compositions means a shared computation between two distinct classes. Even for the visually dissimilar classes, which may not have any complex parts in common, our representation is highly shared at the lower layers of the hierarchy.

To numerically evaluate the shareability of the compositions among the classes, we use the following measure:

$$\text{deg\_share}(\ell) = \frac{1}{|V^\ell|} \sum_{\omega^\ell \in V^\ell} \frac{(\#\text{ of classes that use } \omega^\ell) - 1}{\#\text{ of all classes} - 1},$$

defined for each layer ℓ separately. By "ω^ℓ used by class c" it is meant that the probability of ω^ℓ under c is not equal to zero. To give some intuition behind the measure: $deg\_share = 0$ if no composition from layer ℓ is shared (each class uses its own set of compositions), and it is 1 if each composition is used by all the classes. Figure 6.11 plots the values for the learned 15−class representation. Beside the mean (which defines deg_share), the plots also show the standard deviation.

The highest sharing is between motorbike-bike and giraffe-horse. Sharing decreases in the higher layers where the features become more discriminative and class specific,



(a) (b) (c)

Figure 6.9: Sharing of compositions between classes. Each plot shows a *vocabulary* learned for two object classes. The bottom nodes in each plot represent the 6 edge orientations, one row up denotes the learned second layer compositions, etc. The sixth row (from bottom to top) are the whole-shape, object-layer compositions (i.e., different aspects of shapes in the classes), while the top layer is the class layer which pools the corresponding object compositions together. The green nodes denote the *shared* compositions, while specific colors show class-specific compositions. *From left to right:* Vocabularies for: **(a)** two visually similar classes (motorbike and bicycle), **(b)** two semi-similar object classes (giraffe and horse), **(c)** two visually dissimilar classes (swan and car_front). Notice that even for dissimilar object classes, the compositions from the lower layers are shared between them, thus speeding up the overall object detection.

however, the scores are still relatively high. This means that the approach learned the most sharable parts of the objects sufficiently well.

The size of the vocabulary. It is important to test how the size of the vocabulary (the number of compositions at each layer) scales with the number of classes, since this has a direct influence on both inference time and storage demands. We report the size as a function of the number of learned classes in Figure 6.10.

One can observe a highly logarithmic tendency especially in the lower layers of the hierarchy. This is particularly important because the complexity of inference is much higher for these layers (because they contain less discriminative shapes which are detected more numerously in images). Although the fifth layer contains highly class specific shapes, one can still observe a logarithmic increase in size. The final, object layer, is naturally linear in the number of classes, but it does, however, learn less object shapes than is the number of all training examples of objects in each class. Since the objects in the training examples vary quite substantially, it is hard to expect for the model to achieve a higher compression. As part of future work, we would like to use video examples of objects. This could give us the correspondences between (articulated) object parts across the video, which could, in turn, be used in the model

(distinct compositions could be associated with the same part inside an top-layer, object model).

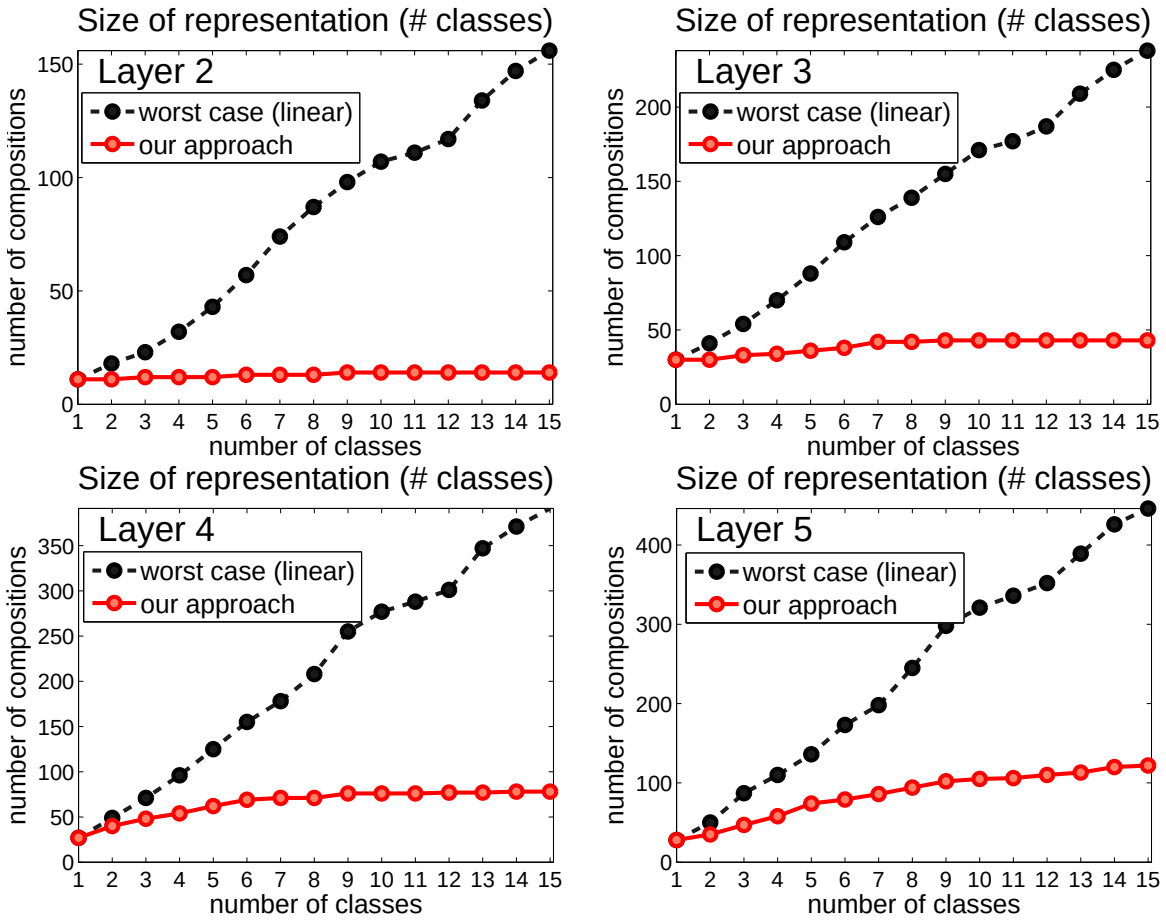


Figure 6.10: Size of representation (the number of compositions per layer) as a function of the number of learned classes. "Worst case" denotes the sum over the sizes of single-class vocabularies.

We further compare the scaling tendency of our approach with the one reported for a non-hierarchical representation by Opelt et al. [111]. The comparison is given in Figure 6.12 where the worst case is taken as in [111] (a 100 features per class). We compare the size of the class-specific vocabulary at layer 5, where the learned shapes are of approximately the same granularity as the features used in [111]. We additionally compare the overall size of the vocabulary (a sum of the number of features over all layers). Our approach achieves a substantially better scaling tendency, which is due to sharing at multiple layers of representation. This, on the one hand, compresses the overall representation, while it also attains a higher variability of the shapes in the higher layers and, consequently, a lower number of them are needed to represent the classes.

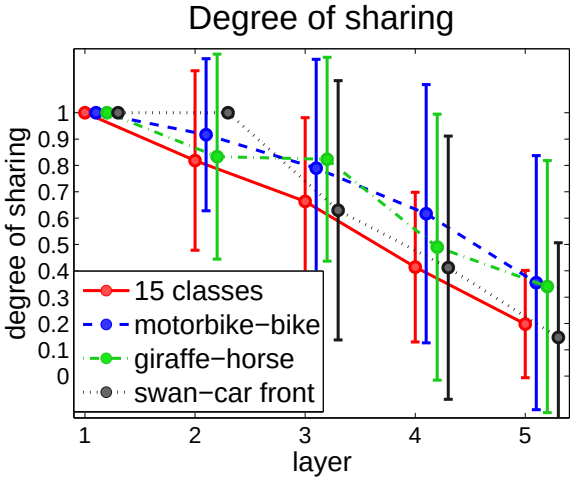


Figure 6.11: Degree of sharing of features (from different layer) among classes for the multi-class vocabulary. The highest sharing seems to be between motorbike-bike, followed by giraffe-horse, both pairs being relatively similar. Sharing decreases in the higher layers where the features become more discriminative and class specific.

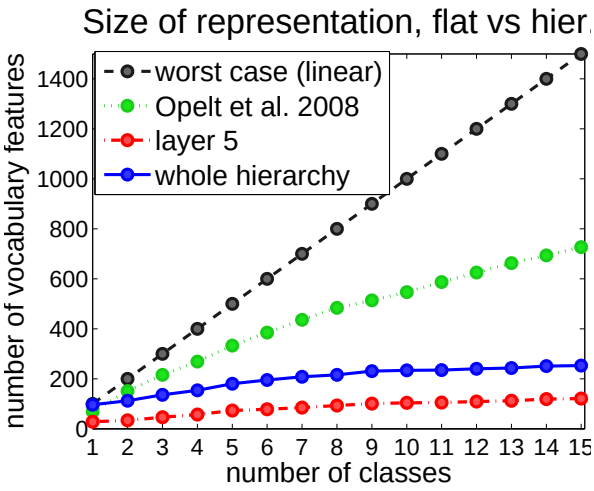


Figure 6.12: A comparison of scaling behaviors of the approach by Opelt et al. [111] (non-hierarchical) and our hierarchical approach. "layer 5" denotes the number of compositions on the fifth layer of the vocabulary, while "whole hierarchy" represents the number of all compositions in the vocabulary. "worst case" is the same as in [111], namely 100 features per class.

Storage. Figure 6.13 shows the storage demands as a function of the number of learned classes. This is the actual size (in Megabytes) of the vocabulary stored on a hard disk. The "worst case" denotes stacking separate vocabularies for each class together, while "our approach" correspond to the learned multi-class vocabulary. Both are linear in the number of classes (the last, object layer is obviously linear in the number of classes). It is worth noting that the 15-class hierarchy takes only $1.6MB$ on disk.

Inference time. We further test how the complexity of inference increases with each additional class learned. We randomly sample ten images per each class (altogether, 150 images are used in the experiment) and report the detection times averaged over all of the selected images. The times are reported as a function of the number of learned classes. The results are plotted in Figure 6.14. The tendency seems to be logarithmic, although, the number of classes is still too small to make an accurate observation. It is worth noting, that it takes only 16 seconds per image to apply a vocabulary of all 15 classes.

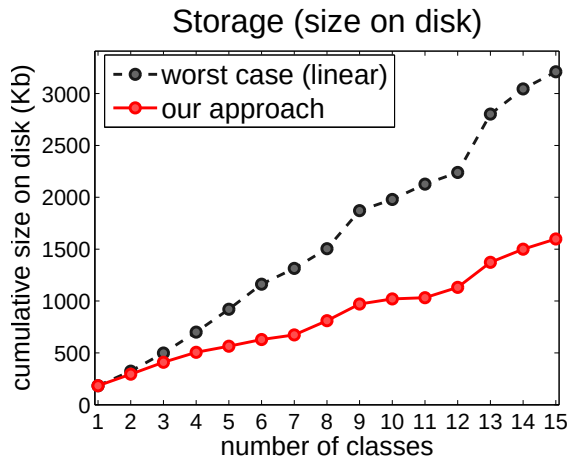


Figure 6.13: Storage demands (size of the hierarchy stored on a computer disk) as a function of the number of learned classes. "worst case" denotes stacking separate vocabularies for each class together, while "our approach" correspond to the learned multi-class vocabulary. Both are linear in the number of classes (the last, object layer is obviously linear in the number of classes). A vocabulary of 15 classes takes up only 1.6Mb of disk space.

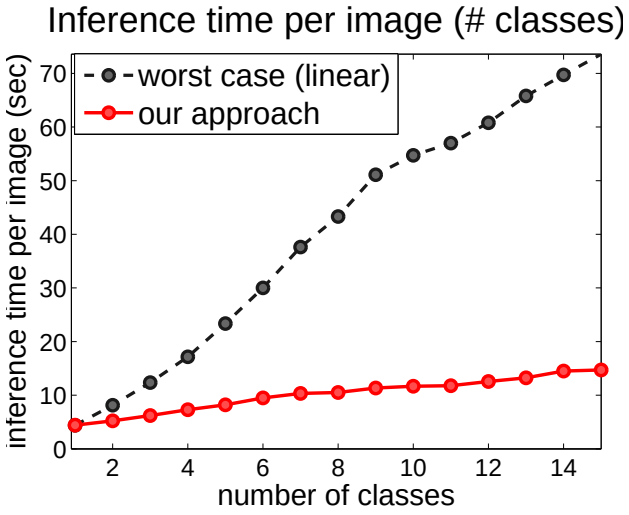


Figure 6.14: Average inference time per image as a function of the number of learned classes. "worst case" corresponds to learning a vocabulary for each class separately and doing detection with each of them, one after another. "our approach" corresponds to learning first three layers on general images, and the higher layers sequentially, one class after another.

Table 6.4: Comparison in detection accuracy for single- and multi-class object representations and detection procedures.

class	Apple	bottle	giraffe	mug	swan	horse	cow	mbike	bike	car_f.	car_s.	car_r.	face	person	cup
measure	det. rate (in %) at 0.4FPPI					recall (in %) at EER									
single	88.6	85.5	83.5	84.9	75.8	84.5	96.9	83.2	68.5	76.5	97.5	93.5	94.0	60.4	85.0
multi	84.1	80.0	80.2	80.3	72.7	82.3	95.4	83.2	64.8	82.4	93.8	92.0	93.0	62.5	85.0

Detection performance. We additionally test the multiclass detection performance. The results are presented in Table 6.4 and compared with the performance of the single class vocabularies. The evaluation was the following. For the case of a single class, a separate vocabulary was trained on each class and evaluated for detection independently of other classes. For the multi-class case, a joint multi-class vocabulary was learned (as explained in Sec. 6.2.2) and detection was performed as proposed in Sec. 5.3.3, that is, we allowed for competition among hypotheses explaining partly the same regions in an image.

A few example detections of object are presented in Figure 6.15. The colored links denote the inference subgraphs corresponding to a particular object class detection.

175 classes. To increase the scale of the experiments we show the performance of our approach on 175 classes from LabelMe [141]. The training and test sets were collected in the following way. Among 318, 407 available object class annotations from LabelMe we selected those that: 1.) the size of the object was at least 200 pixels in diagonal, 2.) was not labeled as occluded, 3.) the class contained at least 30 labeled objects. This narrowed the database to approximately 500 classes among which we manually selected those that are mainly defined by shape, e.g. we discarded the classes like "beach", "mountains", etc. The final dataset contained 175 object classes. For each class we randomly selected 30 annotations. Each training example is thus an image of an object class extracted from the original image with respect to its annotated bounding box. A few class training examples are depicted in Figure 6.16.

Figure 6.17 shows how the number of compositions per each layer grows with the number of learned classes. It is evident that the lower layers converge very quickly (after 5 classes no more new compositions are added to the vocabulary), while Layer 5 grows logarithmically with the number of object classes. Such tendency has been previously reported in non-hierarchical approaches by Krempp et al. [84], Torralba et al. [146], and Opelt et al. [111], where the classifiers were trained to share features. The last, sixth layer, is of course linear (depicted in Figure 6.18), however, the number of Layer 6 compositions for each class is much lower than the number of training examples. This means that our proposed approach is able to achieve good compression of similar objects in a given class.

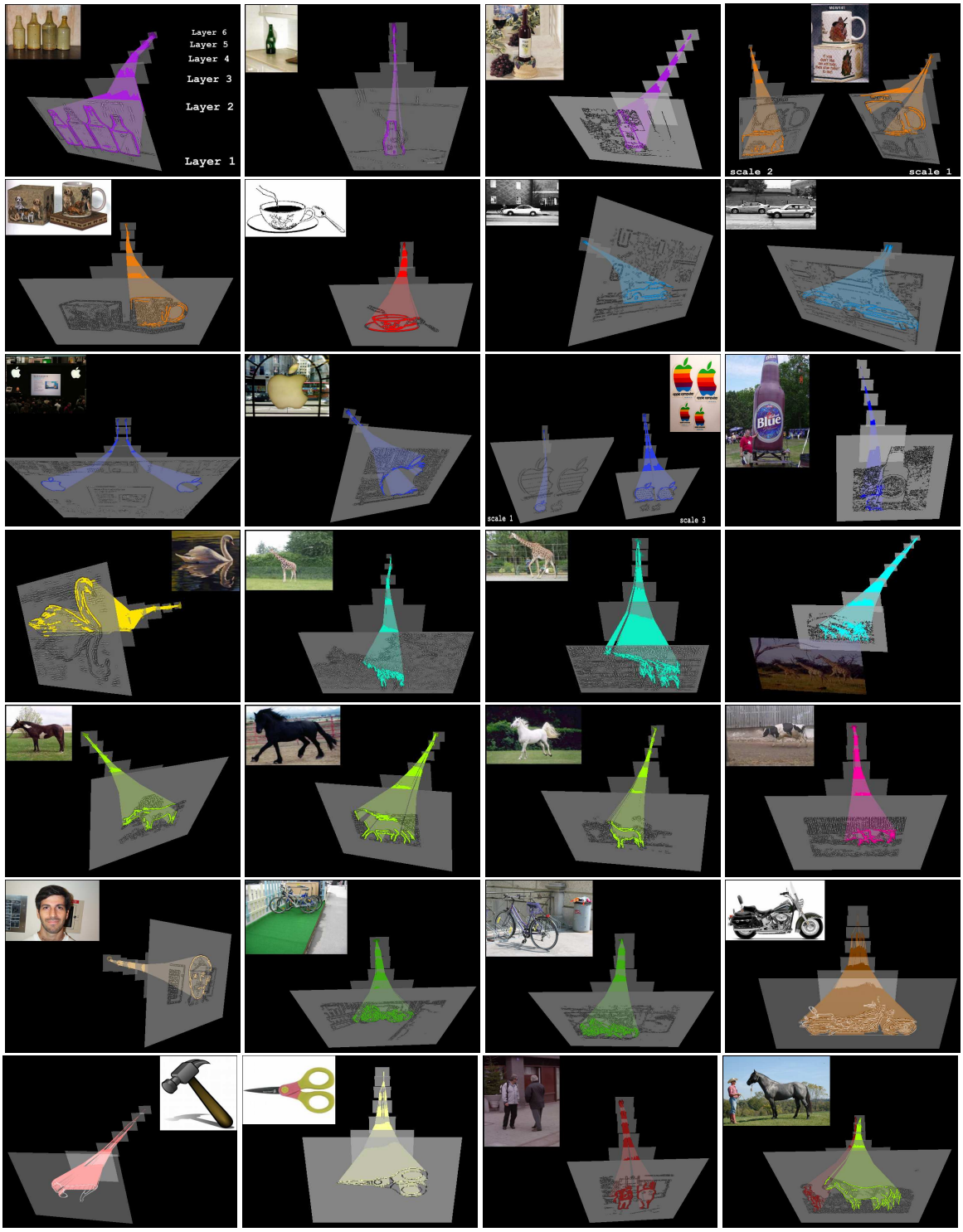


Figure 6.15: Examples of detections. The links show the most probable tree activations for a particular (top-layer) class detection. The links are color-coded to denote different classes. From top left: *bottle*, *bottle*, *bottle*, *mug*, *mug*, *cup*, *car*, *car*, *apple*, *apple*, *apple*, *apple* (false positive detected in clutter), *swan*, *giraffe*, *giraffe*, *giraffe*, *horse*, *horse*, *horse*, *cow*, *face*, *bicycle*, *bicycle*, *motorbike*, *hammer*, *scissors*, *person*, *person* and *horse*.

The results for inference are presented in Figure 6.19. For "independent" in the inference time plot we used the time needed to detect one class, which we then linearly extrapolated with the number of classes. We can observe that our approach achieves much lower inference times than the worst case (no sharing of features among classes), although it is clear that for a higher number of classes more research is needed to cut down the inference times to a practical value. Detection time also appears to be linear in the number of classes, although it is significantly faster than for the independent case. Linearity might also be due to the complexity of test images, where tens of classes can appear in one single image. By increasing the number of classes modeled in the vocabulary, we are thus also increasing the number of classes the approach recognizes in an image, and this dependence could be linear.

Figure 6.20 shows the degree of transfer as a function of the number of learned classes. Notice that even for the more discriminative Layer 5 the majority of features gets re-used as quickly as 75 object classes are learned.

We additionally show the re-usability of features between the 175 object classes in Figure 6.21. A colored point means that a particular class uses a particular feature. Different colors correspond to features at different layers of the hierarchy. We can observe that the features a highly re-used in the lower layers and less in the higher, more discriminative layers.

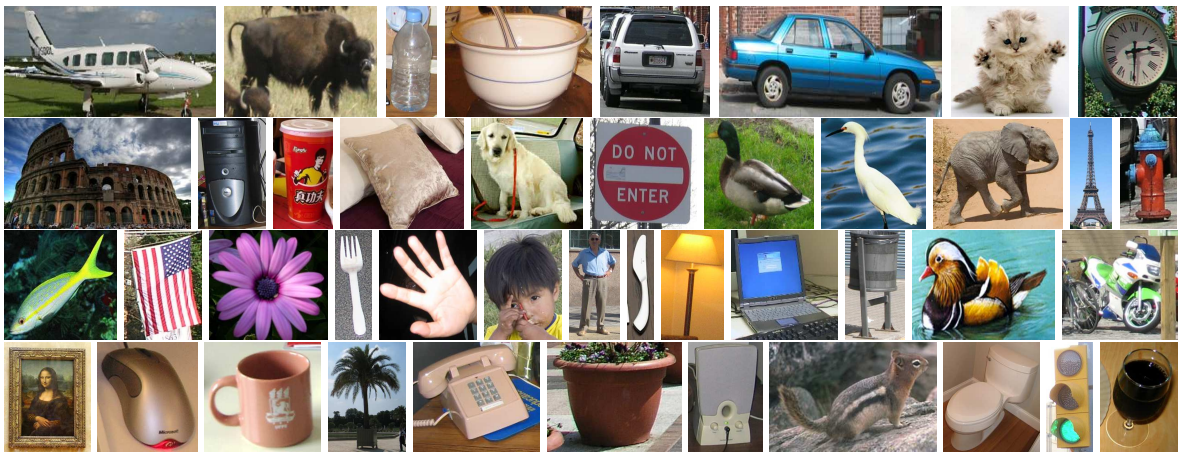


Figure 6.16: Examples of class training examples from the LabelMe dataset [141]. From top left to bottom right: *airplane, bison, bottle, bowl, car az90, car az180, cat, clock, Colosseum, cpu, cup, cushion, dog, do-not-enter sign, duck, egret, elephant, Eiffel Tower, fire hydrant, fish, flag, flower, fork, hand, head, human, knife, lamp, laptop, litter bin, mandarin, motorbike, Mona Lisa, mouse, mug, palm tree, phone, pot, speaker, squirrel, toilet, traffic light, wine glass.*

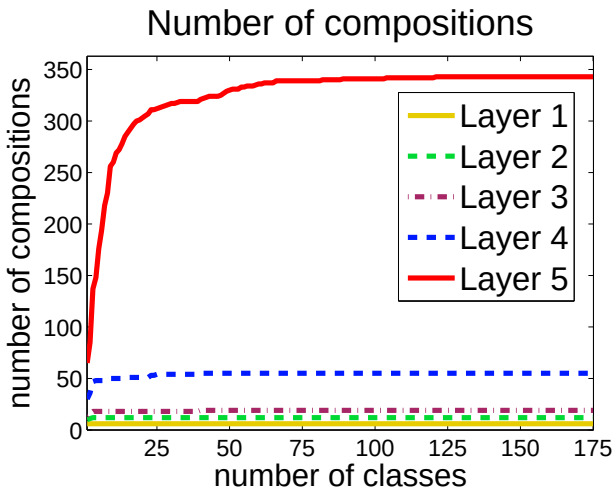


Figure 6.17: Results for 175 object classes from LabelMe [141]. Size of representation (number of compositions per layer) as a function of the number of learned classes. Lower layers stop growing (no more compositions are formed) after a few learned classes, while layer 5 seems to be logarithmic.

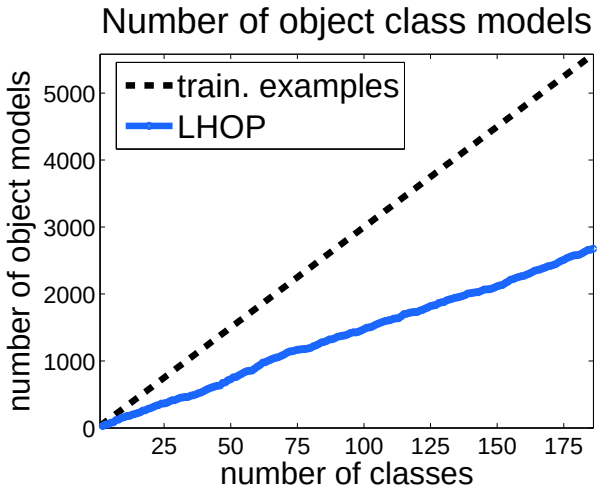


Figure 6.18: Results for 175 object classes from LabelMe [141]. The number of compositions at Layer 6 as a function of the number of learned classes. Since Layer 6 is the object layer, this function is obviously linear. The number of compositions representing each class is much lower than the number of training examples.

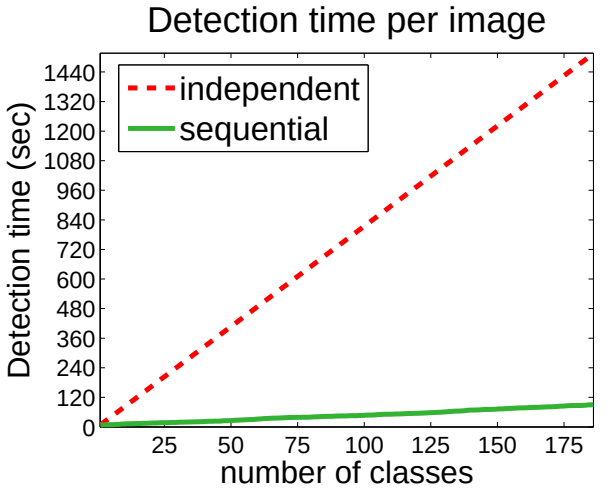


Figure 6.19: Results for 175 object classes from LabelMe [141]. Detection time as a function of the number of learned classes. Detection time appears to be linear in the number of classes, although much faster than for the independent case. Linearity might also be due to the complexity of test images, where tens of classes can appear in one single image.

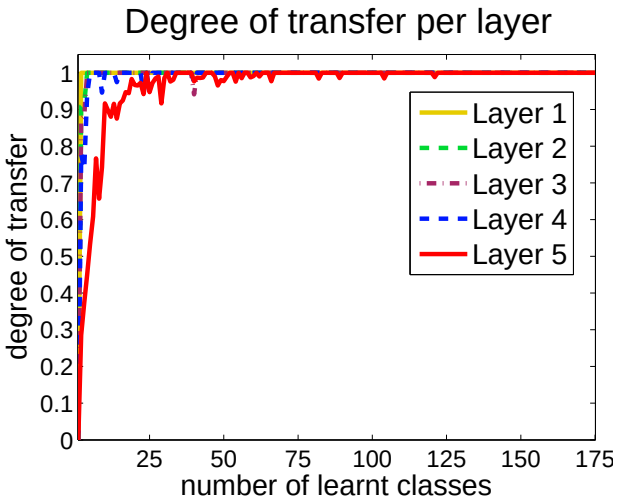


Figure 6.20: Results for 175 object classes from LabelMe [141]. deg_transfer as a function of the number of learned classes. Most features get re-used when learning a novel object class already after having learned a small set of classes.

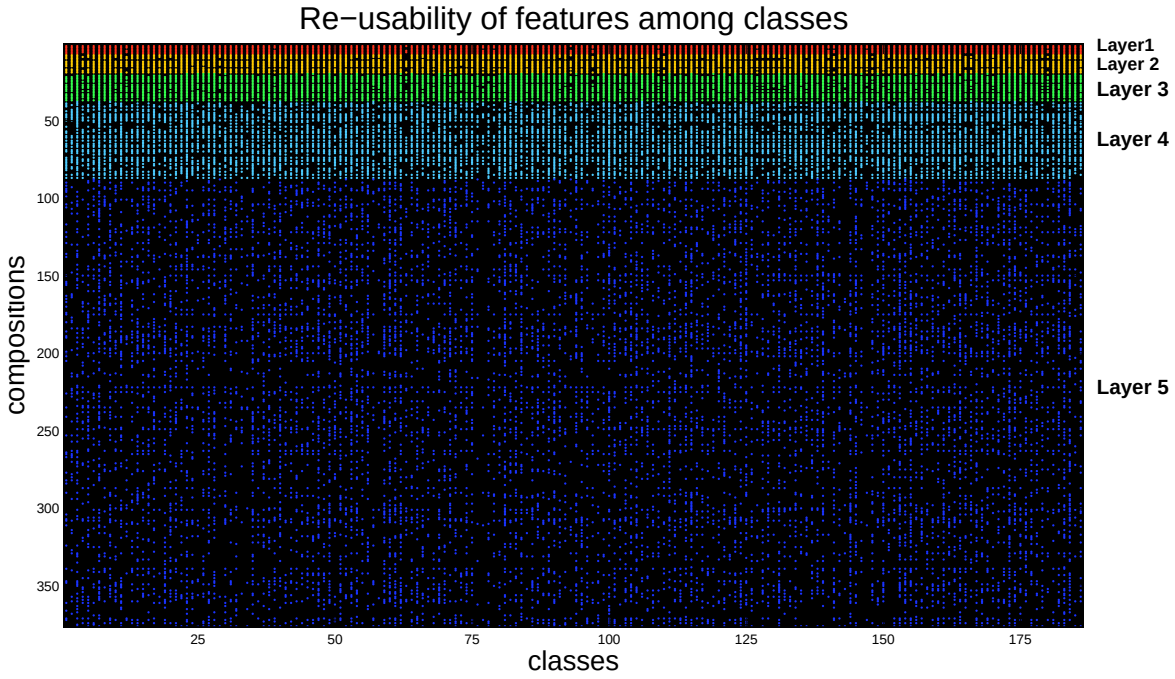


Figure 6.21: Results for 175 object classes from LabelMe [141]. Reusability of features (vertical axis) between 175 object classes (horizontal axis) for different layers. A colored point means that a particular class uses a particular feature. Different colors correspond to features at different layers of the hierarchy. We can observe that the features a highly re-used in the lower layers and less in the higher, more discriminative layers.

6.3 Evaluating multi-class training strategies

We have evaluated our approach for different multi-class learning strategies on several object classes. Specifically, we used: UIUC multi-scale cars [3], GRAZ [111] cows and persons, Weizmann multi-scale horses (adapted by Shotton et al. [133]), all five classes from the ETH dataset [45], and all ten classes from TUD *shape*2 [139]. Basic information is given in Table 6.2. A 6-layer vocabulary was learned.

To evaluate the detection performance we use the measures described in the previous Sections. Apart from those measure, we will also report classification-by-detection (on TUD *shape*2).

To evaluate the shareability of compositions between the classes, we will use the measure previously defined in (6.2.2). In sequential training, we can additionally evaluate the degree of re-use when learning each novel class. Higher re-use means lower training time and a more compact representation. We expect a tendency of higher re-use as the number n of classes grows, thus we define it with respect to the number of learned classes:

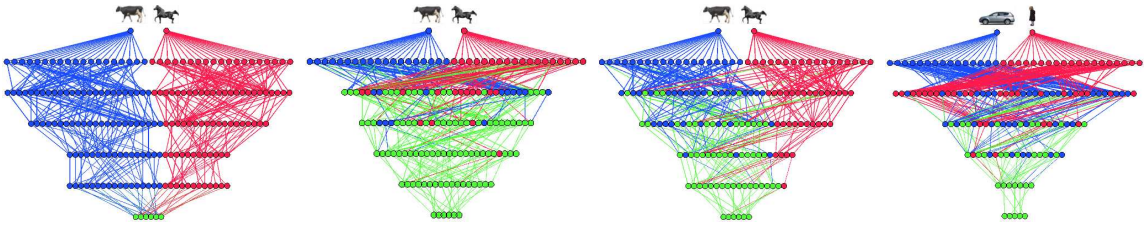
$$\text{deg\_transfer}(n,\ell) = \frac{\#\text{ of } \omega^\ell \in \Omega^\ell_{1:n-1} \text{ used by } c_n}{\#\text{ of all } \omega^\ell \in \Omega^\ell_{1:n} \text{ used by } c_n} \tag{6.1}$$

Here c_n denotes the n-th class.

We have tested the approach by progressively increasing the number of object classes (from 2 to 10). The individual training will be denoted by I, joint by J, and sequential by S.

Two classes. We tested all three learning strategies on two visually very similar classes (cow, horse), and two dissimilar classes (person, car). Table 6.6 gives information on 1.) size (the number of compositions at each layer), 2.) training and 3.) inference time, 4.) recall at EER. In sequential training, both possible orders were used (denoted with $S1$ and $S2$) to see whether different learning orders (of classes) affect the performance. The first two rows show the results for each class individually, while the last row contains information with respect to the conjoined representations.

Degree of sharing. The hierarchies learned in I, J, and S on cows and horses, and J for car-person are shown in Figure 6.22 in a respective order from left to right. The red nodes depict cow/car and blue horse/person compositions. The green nodes depict the shared compositions. We can observe a slightly lower number of shareable nodes for S compared to J, yet still the lower layers for cow-horse are almost completely re-used. What is interesting is that joint J training for cow and horse has found more compositions in common of the two classes in the higher layers (Layer 5), while this was not the case for S. Even for the visually dissimilar classes (car-person) sharing is present at lower layers. Numerically, the degrees of sharing and transfer are plotted in

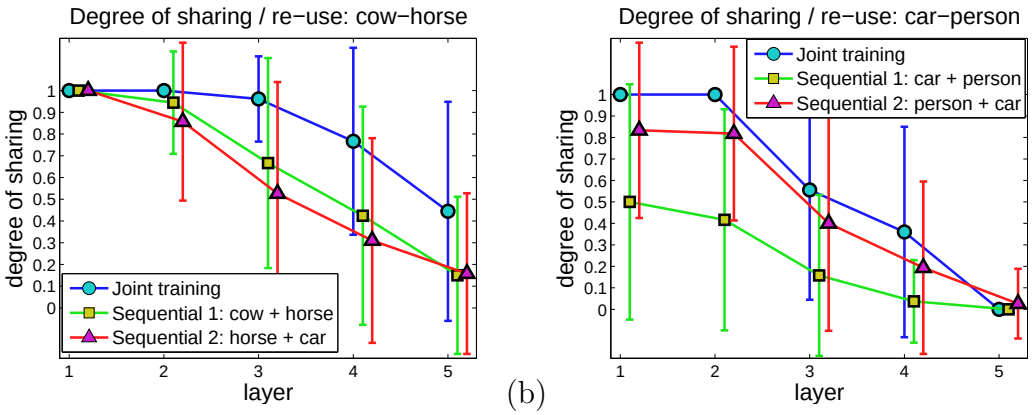


(a) (b) (c) (d)

Figure 6.22: Learned 2-class vocabularies for different learning strategies (the nodes depict the compositions, the links represent the compositional relations between them). From left to right: cow-horse hierarchy for (a) I (two vocabularies, one learned on cows only, one on horse only, stacked together), (b) J, (c) $S1$ (vocabulary trained on cow first, then horse), and (d) car-person J. Green nodes denoted shared compositions. The joint J vocabulary for cow and horse has found more compositions in common of the two classes in the higher layers (Layer 5), while the lower layers are equally sharable for both J and S. For person-car there are no common complex parts (Layer 5 compositions), but there is significant still sharing at the lower layers.

Figure 6.23.

Detection rate. The recall values for each class are reported in Table 6.6. Interestingly, "knowing" horses improved the performance for cows. For car-person, individual training produced the best result, while training person before car turned out to be a better strategy for S. Figure 6.24 shows the detection rates for cows and horses on the **joint** test set (the strongest class hypothesis is evaluated), which allows for a much higher false-positive rate. We evaluate it with F-measure (to account for FP). A higher performance for all joint representations over the independent one can be observed. This is due to the high degree of sharing in J and S, which puts similar



(a) (b)

Figure 6.23: *Degree of sharing* for a) cow-horse, b) car-person vocabularies.

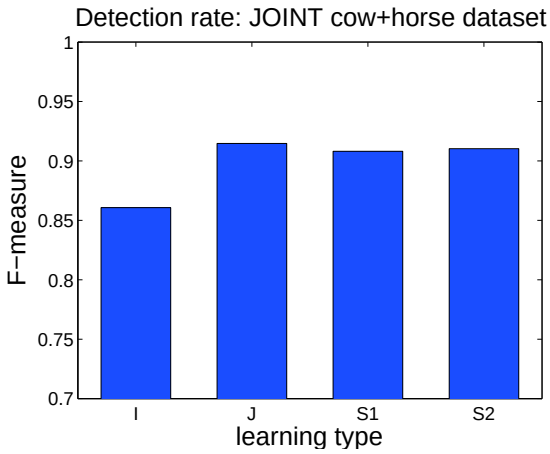


Figure 6.24: Detection rate (F measure) for independent, joint, sequential S_1 (cow + horse) and sequential S_2 (horse + cow) training on the joint cow-horse test set. A joint vocabulary of cow and horse is used for detection and the highest response in an image counts as a detection.

hypotheses in perspective and thus discriminates between them better.

Five classes. The results for ETH-5 are reported in Table 6.7. We used half of the images for training, and the other half for testing. The split was random, but the same for I, J, and S. We also test whether different orders in S affect performance (we report an average over 3 random S runs). Ordering does slightly affect performance, which means we may try to find an optimal order of classes in training. We can also observe that the number of compositions at each layer is higher for S as for J (both being much smaller than I), but this only slightly showed in inference times.

Ten classes. The results on TUD-10 are presented in Table 6.8. A few examples of the learned shapes for S are shown in Figure 6.25. Due to the high training complexity of J, we have only ran J for 2, 5 and 10 classes. We report classification-by-detection (the strongest class hypothesis in an image must overlap with ground truth more than 50%). To demonstrate the strength of our representation, we have also ran a linear SVM with hypotheses from Layers $1-3$, and compared the performances. This was done similarly as proposed in Section 6.1, however, no pooling over different layers was used: for each image a vector of a dimension equal to the number of compositions at the particular layer was formed. Each component (which corresponds to a particular type of a composition, for example an L-junction) of the vector was obtained by summing over all the responses in an image belonging to this particular type of composition. To increase the discriminative information, radial sampling was performed: each image was split into 5 orientation bins (one central bin and 4 orientation bins in the outer circle) in which separate vectors were formed (each dimension in the vector represented a sum over the likelihoods for a particular composition in the vocabulary). These were consequently stacked together into one, high-dimensional vector. We report the performance for each layer.

Already here, Layer 3 + SVM outperforms prior work [139] by 10%. Figure 6.27 shows the classification rates as a number of learned classes. In the case when the

number of classes is small, all 3 approaches perform equally well (or comparably). When the number of classes grows, our approach ("sequential") outperforms both versions of SVM.

Vocabulary size. The top row in Figure 6.26 shows representation size for I, J and S as a function of learned classes. With respect to worst case (I), both J and S have a highly sublinear growth. Moreover, in layers 2 and 3, where the burden on inference is the highest (the highest number of inferred hypotheses), an almost constant tendency can be seen. We also compare the curves with those reported for a flat approach by Opelt et al. [111] in Figure 6.26 (last plot). We plot the number of models at Layer 5 which are approximately of the same granularity as the learned *boundary fragments* in [111]. Both, J and S hierarchical learning types show a significantly better tendency as in Opelt et al. [111].

Inference time. Figure 6.28 shows inference times for the independently, jointly and sequentially trained hierarchies. Jointly trained hierarchy attained the lowest time for inference, but sequentially trained hierarchy performed only slightly slower. Both, joint and sequential, hierarchies are significantly faster than the one produced by independent training, with sequentially learned hierarchy being only slightly slower than the jointly learned one. This means that even though in sequential training we do not have all the data available at once, it is possible to find sharable features among the classes sufficiently well.

Training time. In Figure 6.29 the cumulative training time is depicted for independent and sequential training. Due to the re-use of features, sequential training learns novel classes increasingly faster (up to some constant time) than independent training.

Storage. Figure 6.30 shows the size of the hierarchy file stored on disk. Since the classes have relatively simple shapes with little texture and most of them are not articulated, their representation is consequently very compact.

Sharing and transfer. Figure 6.31 depicts the degree of sharing for the 10-class representation. Notice that while the sharing of features is similar in the lower layers for both, joint and sequential training, the jointly trained vocabulary attains better sharing of features in the higher layers of the hierarchy. Figure 6.32 depicts the degree of transfer as a function of the number of learned classes. As the number of classes grows, more and more features get re-used (also in the more discriminative, higher layers of the hierarchy.

Discussion. Three strategies have been thoroughly evaluated and compared through several important computational aspects as well as by detection performance. We conclude that:

1. Both joint and sequential training strategies exert sublinear growth in vocabulary size, Figure 6.26 (more evidently so in the lower layers) and, consequently,

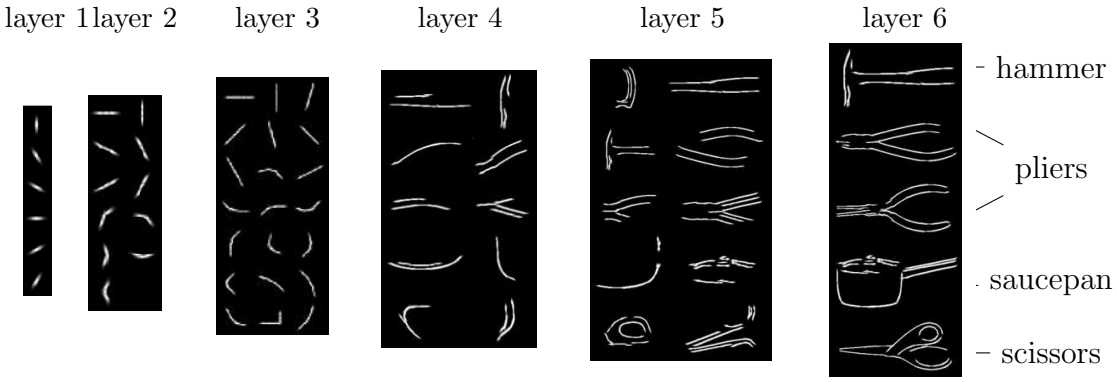


Figure 6.25: A few examples from the learned hierarchical shape vocabulary for S on TUD-10. Each shape in the hierarchy is a composition of shapes from the layer below. Only the *mean* of each shape is shown.

method	size on disk	**classf. rate**						
Stark et al.[139]	/	44%						
Level 1 + SVM	206 Kb	32%						
Level 2 + SVM	3,913 Kb	44%	tr. time	infer. time	num. of. comp.			
Level 3 + SVM	34,508 Kb	54%			L2	L3	L4	L5
Independent	1,249 Kb	71%	207 min	12.2 sec	74	96	159	181
Joint	408 Kb	69%	752 min	2.0 sec	14	23	39	59
Sequential	490 Kb	71%	151 min	2.4 sec	9	21	49	76

Table 6.8: Results on the TUD-10.

Table 6.5: Comparison of detection rates with related work. Left: Average detection-rate (in %) at 0.4 FPPI for the ETH shape database. Right: Recall at EER for various classes.

	applelogo	bottle	giraffe	mug	swan	avg.
[45]	83.2(1.7)	83.2(7.5)	58.6(14.6)	83.6(8.6)	75.4(13.4)	76.8
[63]	89.9(4.5)	76.8(6.1)	90.5(5.4)	82.7(5.1)	84.0(8.4)	84.8
our appr.	87.3(2.6)	**86.2**(**2.8**)	83.3(4.3)	**84.6**(**2.3**)	78.2(5.4)	83.7
(ind.tr.)	0.32 FPPI	**0.36** FPPI	0.21 FPPI	**0.27** FPPI	0.26 FPPI	

	cow	horse	car_scale	person
Related work	100. [111]	89.0 [133]	90.6 [103]	52.6 [111]
	100. [145]	93.0 [132]	93.5 [5]	52.4 [133]
	98.5 [133]	/	92.1 [55]	50.0 [132]
our appr.	98.5	**94.3**	**93.5**	**60.4**

Table 6.6: Results for different learning types on the cow-horse, and car-person classes.

class	size of representation (num. of compositions per layer)																tr.time (min)				infer.time(sec)				**rec. at EER** (%)			
	I				J				S_1 $(1+2)$				S_2 $(2+1)$				I	J	S_1	S_2	I	J	S_1	S_2	I	J	S_1	S_2
	L2	L3	L4	L5	L2	L3	L4	L5	L2	L3	L4	L5	L2	L3	L4	L5												
cow (1)	17	17	23	25	17	25	27	25	17	17	23	25	14	17	20	20	25	/	25	19	1.9	2.0	2.1	2.0	96.9	96.9	96.9	98.5
horse (2)	12	12	18	24	17	26	26	27	18	18	24	21	12	12	18	24	20	/	17	20	2.3	2.6	2.7	2.7	94.3	92.5	93.4	94.3
cow+hrs.	29	29	41	49	17	26	30	36	18	21	33	40	14	19	29	38	45	65	42	39	4.3	2.5	2.6	2.5	95.6	94.7	95.6	**96.4**
car (1)	6	10	13	16	7	16	20	20	6	10	13	16	11	19	18	18	35	/	35	33	3.4	5.2	5.3	5.4	93.5	91.7	93.5	92.4
person (2)	9	16	19	21	7	12	14	22	11	12	15	23	9	16	19	21	17	/	15	17	2.3	2.6	2.8	3.0	60.4	52.1	56.3	60.4
car+prsn.	15	26	32	37	7	18	25	42	12	19	27	39	11	25	31	38	52	85	50	50	6.3	4.8	4.9	5.0	**77.0**	71.9	74.9	76.4

Table 6.7: Results for different learning strategies on the 5−class ETH shape dataset.

class	size of representation												tr.time(min)			inf.time(sec)			**det.rate**(%)			FPPI		
	I				J				S, mean (std) - over 3 runs				I	J	S	I	J	S	I	J	S	I	J	S
	L2	L3	L4	L5	L2	L3	L4	L5	L2	L3	L4	L5												
applelogo	11	30	27	28	14	15	21	28	10(0.6)	25(1.7)	27(12.7)	23(7.5)	23	/	23	3.6	11.1	12.1	88.6	84.1	86.4	0.34	0.27	0.28
bottle	7	11	22	22	13	16	21	22	9(0.6)	22(8.1)	28(2)	22(3.6)	25	/	21	3.4	11.1	12.1	85.5	80.0	80.0	0.4	0.34	0.32
giraffe	5	13	22	37	15	19	26	26	10(0.6)	28(5.9)	35(1.7)	30(1.2)	31	/	26	3.2	11.1	12.1	82.4	81.3	84.6	0.19	0.16	0.18
mug	9	16	25	23	16	19	25	34	10(0.6)	25(7.9)	30(4.7)	29(4.9)	31	/	18	3.6	11.1	12.1	84.9	80.3	83.3	0.31	0.22	0.22
swan	11	18	29	26	16	20	26	27	10(0)	23(6.4)	30(4.0)	27(1.5)	17	/	12	2.8	11.1	12.1	75.8	69.7	72.7	0.28	0.22	0.21
all	43	88	125	136	16	22	32	55	11(0.6)	33(2.5)	61(9.5)	79(13.7)	127	235	100	16.6	11.1	12.1	**83.4**	79.1	81.4	0.30	**0.24**	**0.24**

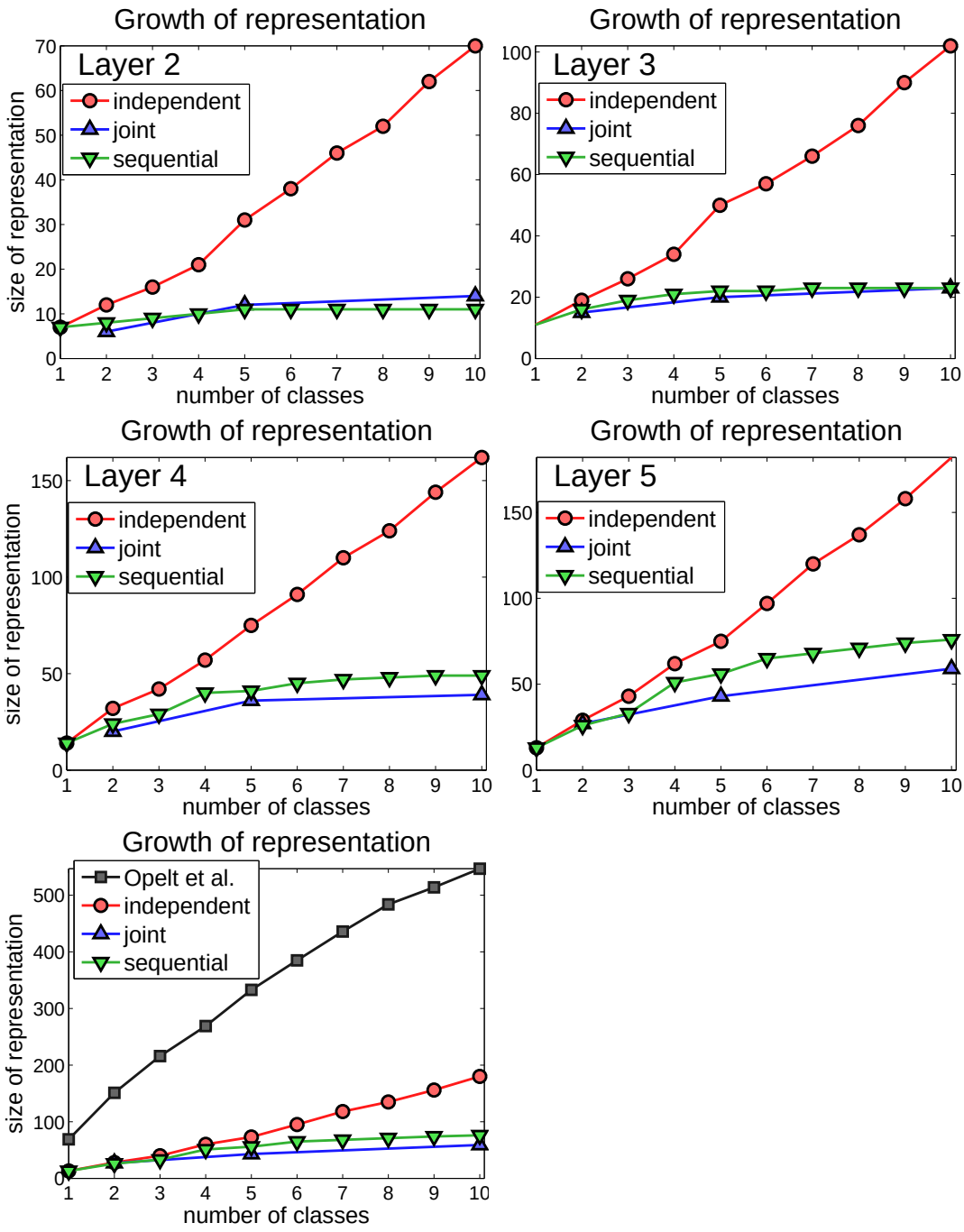


Figure 6.26: Results on TUD-10. (1-4) representation size (number of compositions at a particular layer of the vocabulary) as a function of the number of learned classes; 5.) representation size compared to Opelt et al. [111].

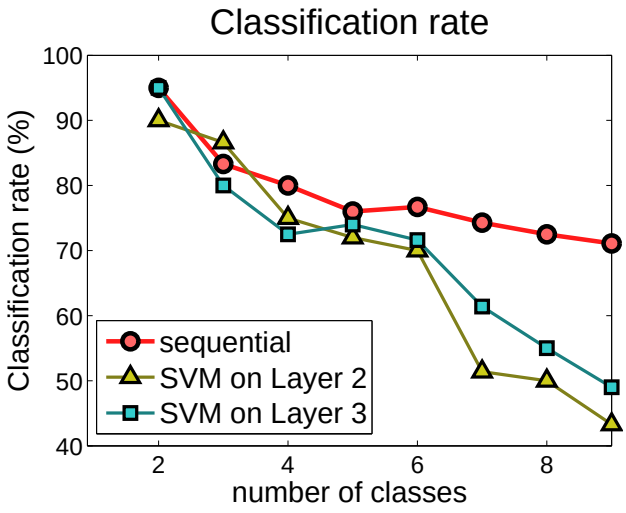


Figure 6.27: Classification rates for TUD-10. Reported results are for sequentially trained 6-layer vocabulary and for Layer 2+SVM and Layer 3+SVM.

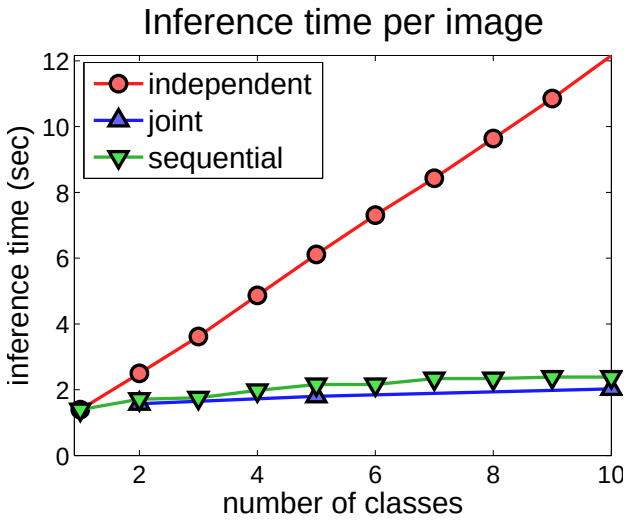


Figure 6.28: Average inference time per image for TUD-10. Jointly trained hierarchy attained the lowest time for inference, but sequentially trained hierarchy performed only slightly slower. Both, joint and sequential, hierarchies are significantly faster than the one produced by independent training.

Figure 6.29: Cumulative training time for TUD-10. It takes approximately $20-30$ minutes to learn a vocabulary for one object class in independent training. Training time reduces when learning each additional object class in sequential training.

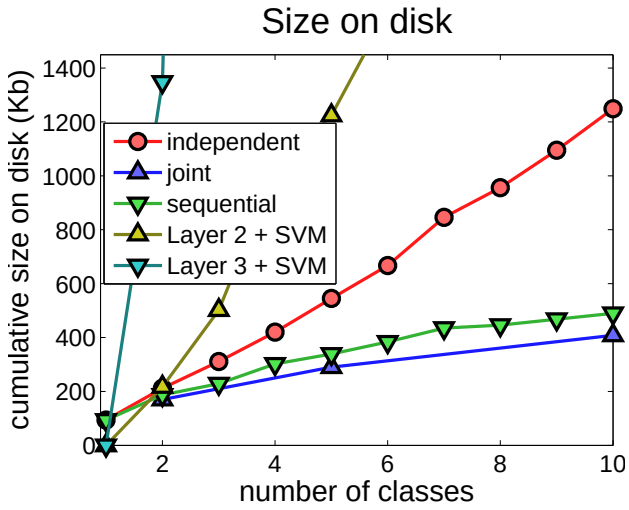


Figure 6.30: Size of the multi-class hierarchical vocabulary on disk for TUD-10. A comparison in size is given with respect to the SVM representation. Jointly trained hierarchy is the most compact and takes less than 400Kb.

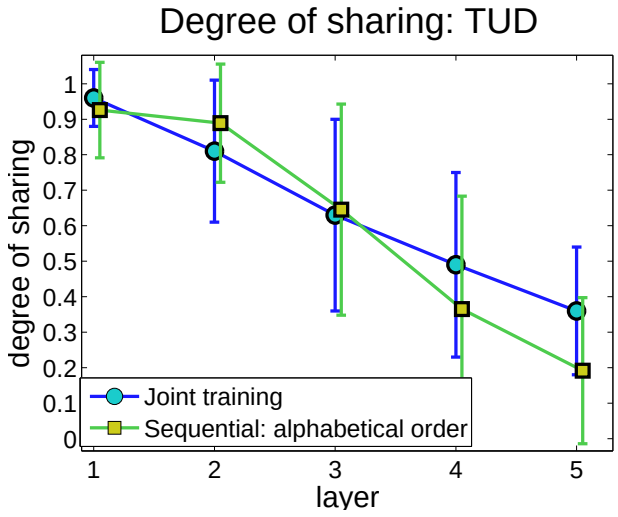


Figure 6.31: Degree of sharing for TUD-10. Jointly trained vocabulary attains better sharing of features in the higher layers of the hierarchy.

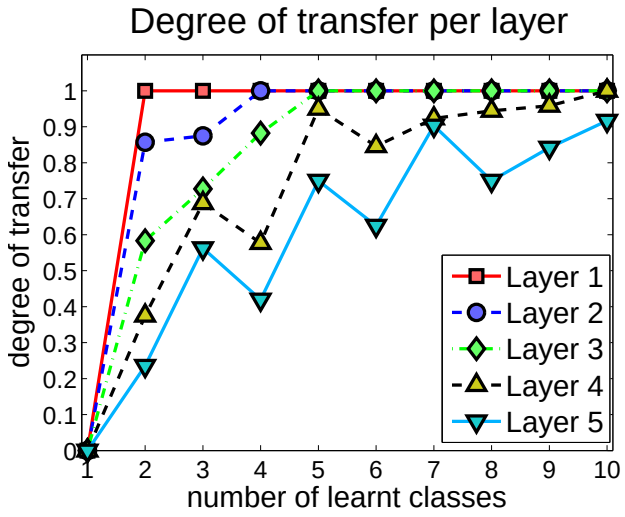


Figure 6.32: Degree of transfer for TUD-10. As the number of classes grows, more and more features are reused to represent novel classes (also in the more discriminative, higher layers of the hierarchy).

sublinear inference time. This is due to a high degree of sharing (Figure 6.31) and transfer (Figure 6.32) within the resulting vocabularies. The hierarchy obtained by sequential training grows somewhat faster, but not significantly so.

2. Training time was expectedly worst for joint training, while training time even reduced with each additional class during sequential training (Table 6.8 and Figure 6.29.

3. Different training orders of classes did perform somewhat differently, Table 6.6 (some had higher re-use than others), but not significantly so — this means we might try to find an "optimal" order of learning. As part of future work, we could explore *curriculum learning* (starting with simple objects and gradually increasing their complexity) as the strategy to optimally learn class representations [13].

4. Training independently has mostly yielded the best detection rates, Tables 6.6 and 6.7, but the discrepancy with the other two strategies was low. For similar classes (cow-horse), sequential learning even improved the detection performance, and was consistently above the joint's performance. By training sequentially, we can learn class specific features (yet still have a high degree of sharing) which boost performance.

Most importantly, sequential training has achieved the best trade-off between detection performance, re-usability, inference and training time. Since, ultimately, the goal is to learn a higher number of object classes, sequential learning seems to be intuitively the most appealing. We have experimentally shown it to be effective both computationally and in terms of detection performance.

6.4 Using different bottom-level features

We have also tested our learning approach in the case when different base features are used for Layer 1. In the first experiment we used polarity edges (which code orientation as well as the polarity of an edge, i.e., dark/white or white/dark are two different features). To detect features in an image that correspond to Layer 1 we used only the odd (sine) Gabor filters. Specifically, we used 8 filters (4 orientations and two polarities). For the total energy in (5.1), we used only the positive output values:

$$\mathcal{E}(x, y, \psi) = r_{-\pi/2}(x, y, \psi) \cdot \mathbf{1}_{r_{-\pi/2}(x,y,\psi) \geq 0}, \tag{6.2}$$

where $r_{-\pi/2}(x, y, \psi)$ denote the outputs of odd Gabor filters at location (x, y) and orientation ψ, respectively.

Figure 6.33 shows the vocabulary learned for three object classes, namely, cars, faces and mugs. Five layers were learned, i.e. the fifth layer corresponded to the *object layer* which codes the whole shape of the objects. With respect to the six layers used in the previous experiments, this only means that images of smaller size were used to train the representation. In the second row, the spatial distributions of the parts are also shown for each of the Layer 2 compositions. Figure 6.34 depicts the connections in the vocabulary for the three classes. In Figure 6.35, a few part-level (Layer 4) and object layer (Layer 5) detections are shown. It is interesting to see that the face detector works on real faces, hand-drawn faces and also on a monkey face.

In the second experiment even simpler feature were used to define Layer 1. Specifically, we used the Difference-of-Gaussians (DoG) filters, defined as:

$$g_{\sigma_1,\sigma_2}(x,y) = \frac{1}{2\pi\sigma_1^2}\exp\Big(-\frac{x^2+y^2}{2\sigma_1^2}\Big) - \frac{1}{2\pi\sigma_2^2}\exp\Big(-\frac{x^2+y^2}{2\sigma_2^2}\Big),$$

where (x, y) represents the center of the filter's domain, and the parameters are set to $(\sigma_1, \sigma_2) = (\sqrt{2}, 2)$.

Figure 6.36 depicts the vocabulary trained on 500 natural images. It can be seen that the second layer has learned oriented edges as well as "end-stop" orientation detectors (the detectors for the ends of oriented lines). Layer 3 has picked up corner and curvature like structures as previously learned with the oriented base features.

6.5 Discussion

We experimentally demonstrated the following important issues:

1. Applied to a collection of natural images, the approach *learns* shape structures that resemble those emphasized by the Gestalt theory [158];

2. We show that these generic compositions can be effectively used for object classification.

3. For object detection we demonstrated a competitive speed of detection with respect to the related approaches already for a single class;

4. For multi-class detection we achieved a sub-linear growth in size of the hierarchical vocabulary. However, the complexity of inference seems to be linear in the number of classes, at least on very complex scenes from the LabelMe dataset. As part of future work, we may want to also build class hierarchies, i.e. organize the object class representation in a taxonomic way.

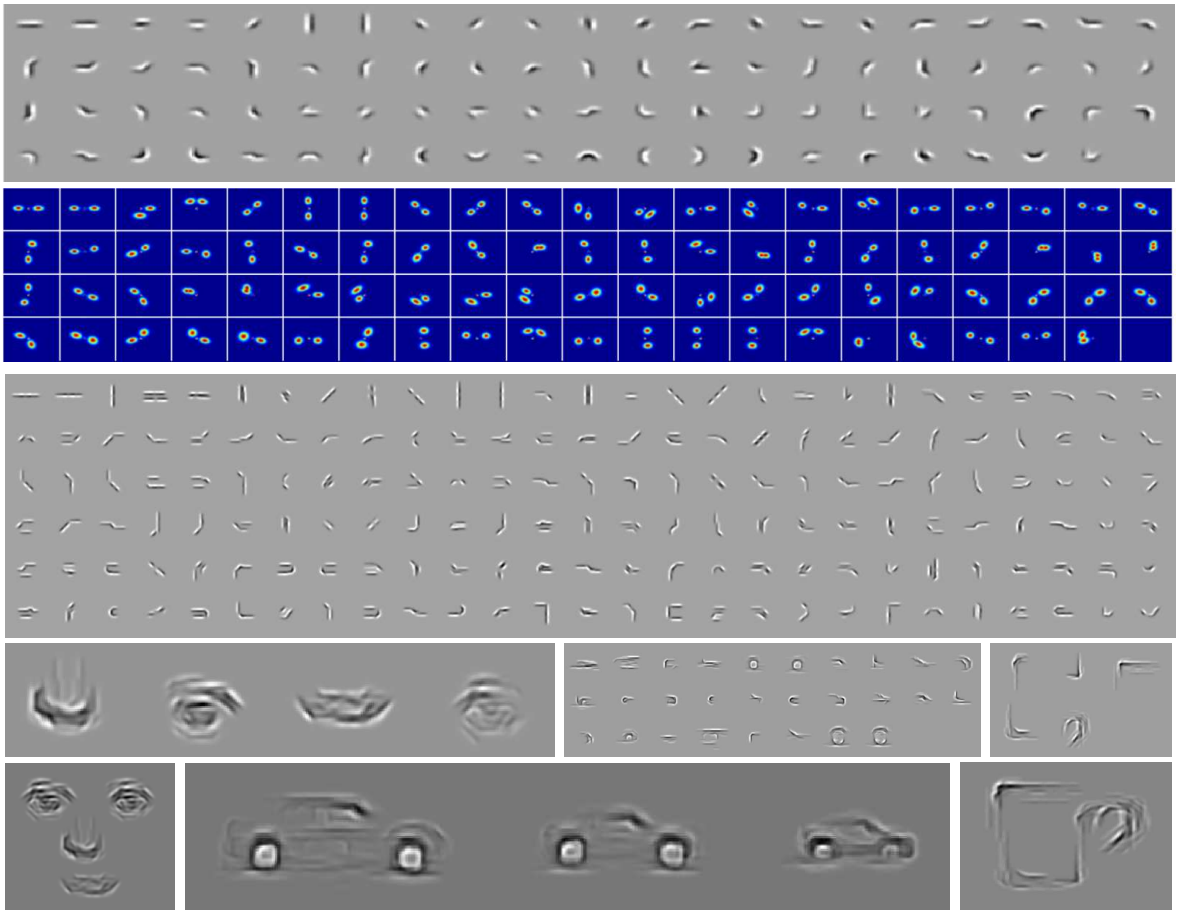


Figure 6.33: Examples of the learned compositions (only the mean shapes are shown). **1st row and 3rd row**: Layer 2, Layer 3, respectively, **2nd row**: Learned spatial flexibility modeled in Layer 2 parts, **4th row**: Layer 4 compositions for faces, cars, and mugs, **5th row**: Layer 5 compositions for faces, cars (obtained on 3 different scales), and mugs.

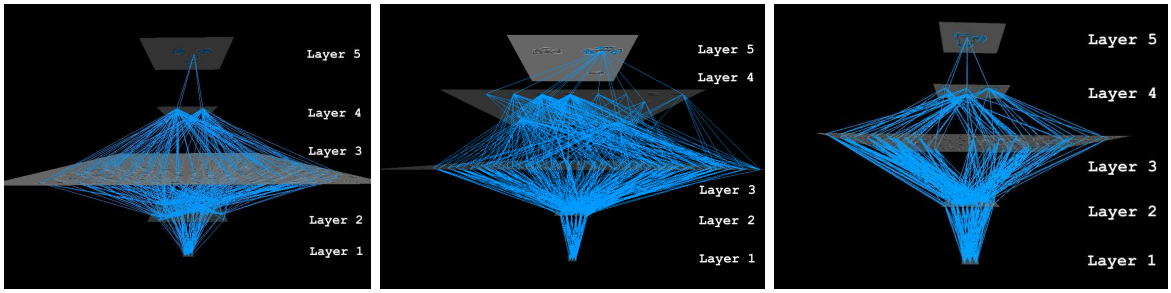


Figure 6.34: Learned vocabularies for faces, cars and mugs (Layers 1 to 3 are the same for all object classes). The connections show which features are used for each of the object classes.

5. We demonstrated a competitive detection performance with respect to the current state-of-the-art.

6. The hierarchical vocabulary representing 15 object classes is very compact; it uses only 1.6MB on disk.

7. We compared three training strategies (independent, joint and sequential). It was shown that sequential training achieves the best trade-off between detection performance, re-usability, inference and training time.

8. It was shown that the approach is able to learn the representation when using different base features at Layer 1. Most notably, we are able to learn generic structures such as corners and curvatures and other higher-order geometric structures out of features as simple as On-Off/Off-On detectors. This might have implications that the biological visual system could learn the selectivities in the higher layers of cortical processing by adjusting to the statistics of the visual environment.

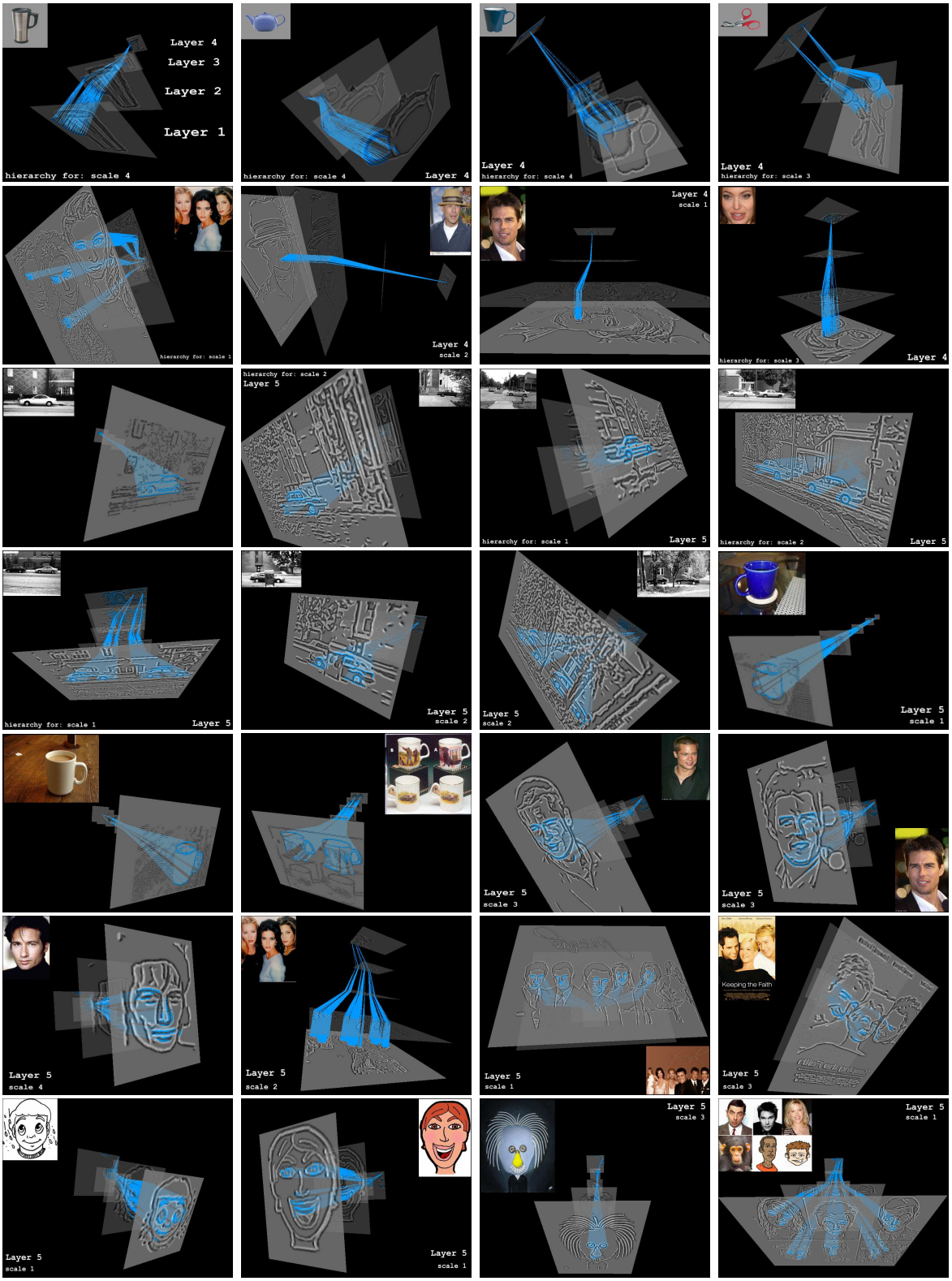


Figure 6.35: Examples of detections for Layer 4 (part-level) (first two rows) and Layer 5 (here object layer) compositions. The links denote the most probable tree activations (parse trees).

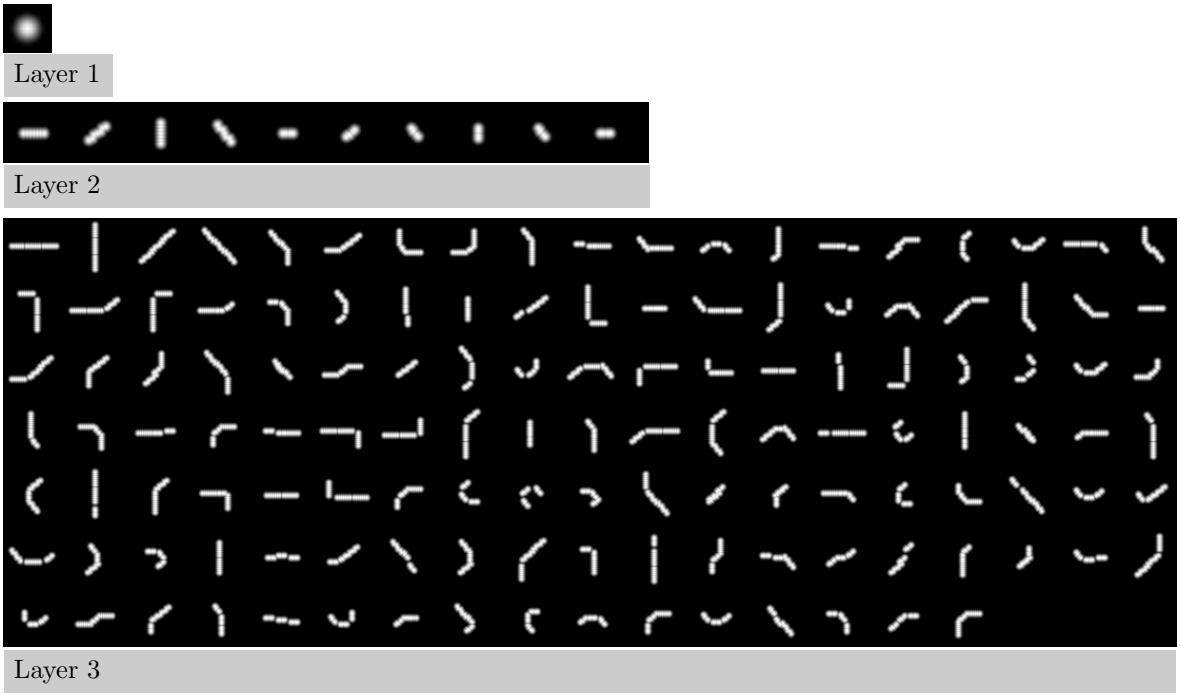


Figure 6.36: Examples of the learned vocabulary compositions when the Difference-of-Gaussians (DoG) filters are used at Layer 1. Only the mean of the compositions are depicted.

Chapter 7

Summary and Conclusions

We proposed a novel approach which learns a hierarchical compositional shape vocabulary to represent multiple object classes in an unsupervised manner. Learning is performed bottom-up, from small oriented contour fragments to whole-object class shapes. The vocabulary is learned recursively, where the shapes at each layer are combined via spatial relations to form larger and more complex shape compositions.

Experimental evaluation was two-fold: one that shows the capability of the method to learn generic shape structures from natural images and uses them for object classification, and another one that utilizes the approach in multi-class object detection. We have demonstrated a competitive classification and detection performance, fast inference even for a single class, and most importantly, a highly logarithmic growth in the size of the vocabulary and, consequently, a logarithmic inference complexity as the number of modeled classes increases. The observed scaling tendency of our hierarchical framework goes well beyond that of a flat approach [111]. This provides an important showcase that highlights learned hierarchical compositional vocabularies as a suitable form of representing a higher number of object classes.

7.1 Future work

There are numerous directions for future work. One important aspect would be to include multiple modalities in the representation. In the current implementation, we only use 2D shape (contours) to represent the objects. Since many object classes have distinctive textures and color, adding this information to our model would increase the range of classes that the method could be applied to. Additionally, modeling texture could also boost the performance of our current model: since textured regions in an image usually have a lot of noisy Layer 1 feature detection, they are currently more susceptible to false positive object detections. Having a model of texture could be used

to remove such regions from the shape processing model.

Another possible direction would be to use video to train the more articulated objects such as people, horses, etc. Motion information could be used to track different shape compositions across frames and establish correspondences between them. For example, the legs of a horse can take different visual appearances as the horse moves and learning their correspondence as being one "leg" part could further improve and compress the object class representation.

An interesting extension would also be to apply the approach to stereo images of objects to learn 3D object representations. This would be especially important for robotic applications. Learning a set of generic 3D structures could, for example, be used for the tasks of robot grasping and manipulation.

Part of our ongoing work is to make the approach scalable to a large number of object classes. To achieve this, several improvements and extensions are still needed. We will need to incorporate the discriminative information into the model (to distinguish better between the similar classes such as cow, horse, zebra, etc), make use of contextual information to speed-up detection and improve performance in the case of ambiguous information (small objects, large occlusion, etc) and use attention mechanisms to speed-up detection in large complex images. Furthermore, a taxonomic organization of object classes could further improve the speed of detection and, possibly, also the recognition rates of the approach. However, this would only be worth exploring when the number of classes is significantly large.

7.2 Possible relations to biological systems

We conclude the thesis with some parallels between the proposed approach and the biological systems.

The visual information in the primate brain is processed in two visual pathways. The *dorsal* pathway mostly processes the motion information and is involved in recognizing movements/actions. The *ventral* pathway is mainly involved in processing 2D and 3D shape. While clearly there must be interaction between these two areas, the details of them are not yet well understood. It is, however, well known, that the organization of both pathways is hierarchical [122]: in the ventral pathway the information is sent from the retina to the Lateral geniculate nucleus (LGN), where there is evidence that the cells have ON-OFF/OFF-ON receptive fields (similar to Difference-of-Gaussians filters). From LGN the information is sent to the V1 of the visual cortex, where the neurons respond to oriented bars of different widths and lengths [122]. V1 has a retinotopic organization, i.e. neighboring cells map to neighboring cells in the retina (the topological structure of an incoming image is retained). The outputs of

V1 cells are processed by V2 cells, which have been claimed to detect more complex features such as corner-like structures [78, 9] which is similar to the second layer of our model. These responses are then passed to the area V4 where the receptive fields are approximately 4 times larger than that of the cells in V1 and respond to complex features such as various curvatures and corner combinations [113, 22]. In our model, layer 3 contains similar feature detectors. The connections between these areas are not as simple as described here; there also exist by-pass connections between V1 and V4 and most importantly, all connections are also reciprocal — with the feedback connections from V4 to V2, V2 to V1, V4 to V1 and V1 to LGN even significantly outnumbering the feed-forward connections. These backward connections could have the role of feedback of higher layer responses to the lower layers where the information is much more ambiguous. Such connections could also be implemented in our model, but are currently only used to track the object responses back to the image to obtain the segmentation of the objects.

From V4 the information is passed to the Inferotemporal cortex (IT), which is again organized into several hierarchical regions. The first one is TE/TEO which is claimed to be the last retinotopically organized area. There is evidence that the cells correspond to complex object parts [143, 150]. Much less is known about the higher object areas and how exactly the object representations are formed [120]. In this respect, our model could be used to predict the responses of the high-level visual neurons and facilitate neurophysiological experiments.

While there is still no consensus on how much is pre-determined genetically and how much is learned, there is increasing evidence that even the receptive fields in lower layers of the hierarchical visual pathway are learned [128, 65, 161, 83, 82]. In our model we show that unsupervised learning and exploiting the statistics of natural images can lead to reliable and generic feature detectors that can be used to represent the 2D object shape.

Bibliography

[1] Cognitive systems for cognitive assistants - cosy. *EU FP6 IST Cognitive Systems Integrated project*.

[2] On the algorithmic implementation of multi-class svms. *Journal of Machine Learning Research*, 2:265–292, 2002.

[3] S. Agarwal, A. Awan, and D. Roth. Learning to detect objects in images via a sparse, part-based representation. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 26(11):1475–1490, 2004.

[4] A. Ahmed, K. Yu, W. Xu, Y. Gong, and E. P. Xing. Training hierarchical feed-forward visual recognition models using transfer learning from pseudo tasks. In *European Conference on Computer Vision*, volume III, pages 69–82, 2009.

[5] N. Ahuja and S. Todorovic. Connected segmentation tree — a joint representation of region layout and hierarchy. *IEEE Computer Vision and Pattern Recognition*, 2008.

[6] Y. Amit. *2d Object Detection and Recognition: Models, Algorithms and Networks*. MIT Press, Cambridge, 2002.

[7] Y. Amit and D. Geman. A computational model for visual selection. *Neural Computation*, 11(7):1691–1715, 1999.

[8] Y. Amit, D. Geman, and X. Fan. A coarse-to-fine strategy for multiclass shape detection. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 26(12):1606–1621, 2004.

[9] A. Anzai, X. Peng, and D. C. Van Essen. Neurons in monkey visual areaa v2 encode combinations of orientations. *Nature Neuroscience*, 10(10):1313–1321, 2007.

[10] H. B. Barlow. Cerebral cortex as a model builder. In D. Rose and V. Dobson, editors, *Models of the Visual Cortex*, pages 37–46. John Wiley: Chichester, 1985.

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

[12] A. J. Bell and T. J. Sejnowski. An information maximisation approach to blind separation and blind deconvolution. *Neural Computation*, 7(6):1129–1159, 1995.

[13] Y. Bengio, J. Louradour, R. Collobert, and J. Weston. Curriculum learning. In *ICML '09: Proceedings of the 26th Annual International Conference on Machine Learning*, pages 41–48, New York, NY, USA, 2009. ACM.

[14] E. Bienenstock. Composition. In A. Aertsen and V. Braitenberg, editors, *Brain Theory - Biological Basis and Computational Theory of Vision*, pages 269–300. Elsevier, 1996.

[15] E. Bienenstock and S. Geman. Compositionality in neural systems. In M. Arbib, editor, *The Handbook of Brain Theory and Neural Networks*, pages 223–226. MIT Press, 1995.

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

[17] G. Blanchard and D. Geman. Hierarchical testing designs for pattern recognition. *Annals of Statistics*, 33(3):1155–1202, 2005.

[18] M. Borga. *Learning Multidimensional Signal Processing*. PhD thesis, Linköping University, Sweden, 1998.

[19] M. Borga and H. Knutsson. Canonical correlation analysis in early vision processing. In *Proceedings of ESANN*, 2001.

[20] A. Bosch, A. Zisserman, and X. Muoz. Image classification using random forests and ferns. In *IEEE International Conference on Computer Vision*, 2007.

[21] G. Bouchard and B. Triggs. Hierarchical part-based visual object categorization. In *IEEE Computer Vision and Pattern Recognition*, pages 710–715, 2005.

[22] S. Brincat and C. Connor. Dynamic shape synthesis in posterior inferotemporal cortex. *Neuron*, 49(1):17–24, 2006.

[23] A. Califano and R. Mohan. Multidimensional indexing for recognizing visual shapes. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 16(4):373–392, 1994.

[24] Y. Chen, L. Zhu, C. Lin, A. Yuille, and H. Zhang. Rapid inference on a novel and/or graph for object detection, segmentation and parsing. In *Neural Information Processing Systems*, pages 2153–2160, 2007.

[25] C. V. Chork and P. J. Rousseeuw. Integrating a high-breakdown option into discriminant analysis in exploration geochemistry. *Journal of Geochemical Exploration*, 43(2):191–203, 1992.

[26] P. Comon. Independent component analysis - a new concept? *Signal Processing*, 36:287–314, 1994.

[27] C. E. Connor, S. L. Brincat, and A. Pasupathy. Transformation of shape information in the ventral pathway. *Current Opinion in Neurobiology*, 17(2):140–147, 2007.

[28] J. C. Corbeil and A. Archambault. *The Firefly Visual Dictionary.* Toronto : Firefly Books, 2002.

[29] C. Croux and C. Dehon. Robust linear discrimination analysis using s-estimators. *The Canadian Journal of Statistics*, 29:473–492, 2001.

[30] F. De la Torre and M. J. Black. A framework for robust subspace learning. *International Journal of Computer Vision*, 54(1):117–142, 2003.

[31] C. Dehon, P. Filzmoser, and C. Croux. Robust methods for canonical correlation analysis. In *Data Analysis, Classification and Related Methods*, volume B, pages 321–326. Berlin: Springer-Verlag, 2000.

[32] S. Dickinson. The evolution of object categorization and the challenge of image abstraction. In S. Dickinson, A. Leonardis, B. Schiele, and M. J. Tarr, editors, *Object Categorization: Computer and Human Vision Perspectives*. Cambridge University Press, 2009.

[33] S. Dickinson, A. Pentland, and A. Rosenfeld. 3-D shape recovery using distributed aspect matching. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 14(2):174–198, 1992.

[34] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification.* Wiley-Interscience, 2nd edition, 2000.

[35] S. Edelman and N. Intrator. Unsupervised statistical learning in vision: computational principles, biological evidence. In *extended abstract of invited talk at the ECCV-2004 Workshop on Statistical Learning in Computer Vision*, 2004.

[36] S. Edelman, N. Intrator, and J. S. Jacobson. Unsupervised learning of visual structure. In *Biologically Motivated Computer Vision*, pages 629–642, 2002.

[37] B. Epshtein and S. Ullman. Semantic hierarchies for recognizing objects and parts. In *IEEE Computer Vision and Pattern Recognition*, 2007.

[38] G. J. Ettinger. Hierarchical object recognition using libraries of parameterized model sub-parts. Technical report, MIT, 1987.

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

[40] L. F. Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *Computer Vision and Image Understanding*, 106(1):59–70, April 2007.

[41] L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: an incremental bayesian approach tested on 101 object categories. In *IEEE CVPR'04, Workshop on Generative-Model Based Vision*, 2004.

[42] L. Fei-Fei, R. Fergus, and A. Torralba. Recognizing and learning object categories. In *http://people.csail.mit.edu/torralba/shortCourseRLOC/*, 2009.

[43] R. Fergus, P. Perona, and A. Zisserman. A sparse object category model for efficient learning and exhaustive recognition. In *IEEE Computer Vision and Pattern Recognition*, volume 1, pages 380–397, 2005.

[44] R. Fergus, P. Perona, and A. Zisserman. Weakly supervised scale-invariant learning of models for visual recognition. *International Journal of Computer Vision*, 71(3):273–303, March 2007.

[45] V. Ferrari, L. Fevrier, F. Jurie, and C. Schmid. Accurate object detection with deformable shape models learnt from images. In *IEEE Computer Vision and Pattern Recognition*, 2007.

[46] V. Ferrari, L. Fevrier, F. Jurie, and C. Schmid. Groups of adjacent contour segments for object detection. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 30(1):36–51, 2008.

[47] S. Fidler, G. Berginc, and A. Leonardis. Hierarchical statistical learning of generic parts of object structure. In *IEEE Computer Vision and Pattern Recognition*, pages 182–189, 2006.

[48] S. Fidler, M. Boben, and A. Leonardis. Similarity-based cross-layered hierarchical representation for object categorization. In *IEEE Computer Vision and Pattern Recognition*, 2008.

[49] S. Fidler, M. Boben, and A. Leonardis. Evaluating multi-class learning strategies in a generative hierarchical framework for object detection. In *Neural Information Processing Systems*, 2009.

[50] S. Fidler, M. Boben, and A. Leonardis. Learning a hierarchical compositional shape vocabulary for multi-class object representation. *Submitted.*, 2009.

[51] S. Fidler, M. Boben, and A. Leonardis. Learning hierarchical compositional representations of object structure. In S. Dickinson, A. Leonardis, B. Schiele, and M. J. Tarr, editors, *Object Categorization: Computer and Human Vision Perspectives*. Cambridge University Press, 2009.

[52] S. Fidler, M. Boben, and A. Leonardis. Optimization framework for learning a hierarchical shape vocabulary for object class detection. In *British Machine Vision Conference*, 2009.

[53] S. Fidler and A. Leonardis. Robust lda classification. In *Vision in a dynamic world: 27th workshop of the Austrian Association for Pattern Recognition*, pages 119–126, 2003.

[54] S. Fidler and A. Leonardis. Robust lda classification by subsampling. In *Workshop in Statistical Analysis in Computer Vision in conjunction with CVPR*, 2003.

[55] S. Fidler and A. Leonardis. Towards scalable representations of visual categories: Learning a hierarchy of parts. In *IEEE Computer Vision and Pattern Recognition*, 2007.

[56] S. Fidler, D. Skočaj, and A. Leonardis. Combining reconstructive and discriminative subspace methods for robust classification and regression by subsampling. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 28(3):337–350, 2006.

[57] S. Fine, Y. Singer, and N. Tishby. The hierarchical hidden markov model: Analysis and applications. *Machine Learning*, 32(1):41–62, 1998.

[58] M. Fischler and R. Elschlager. The representation and matching of pictorial structures. *IEEE Transactions on Computers*, (C-22):67–92, 1973.

[59] J. Fiser and R. N. Aslin. Statistical learning of new visual feature combinations by infants. *Proc Natl Acad Sci U S A*, 99(24):15822–15826, 2002.

[60] R. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7:179–188, 1936.

[61] F. Fleuret and D. Geman. Coarse-to-fine face detection. *International Journal of Computer Vision*, 41(1/2):85–107, 2001.

[62] A. R. J. Francois and G. G. Medioni. Generic shape learning and recognition. In *Object Representation in Computer Vision*, pages 287–320, 1996.

[63] M. Fritz and B. Schiele. Decomposition, discovery and detection of visual categories using topic models. In *IEEE Computer Vision and Pattern Recognition*, 2008.

[64] K. Fukushima, S. Miyake, and T. Ito. Neocognitron: a neural network model for a mechanism of visual pattern recognition. *IEEE Systems, Man and Cybernetics*, 13(3):826–834, 1983.

[65] C. Furmanski, D. Schluppeck, and S. Engel. Learning strengthens the response of primary visual cortex to simple patterns. *Current Biology*, 14(7):573–578, 2003.

[66] S. Geman, D. Potter, and Z. Chi. Composition systems. *Quarterly of Applied Mathematics*, 60(4):707–736, 2002.

[67] W. E. L. Grimson and T. Lozano-Perez. Localizing overlapping parts by searching the interpretation tree. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 9(4):469–482, 1987.

[68] R. Gross, I. Matthews, and S. Baker. Fisher light-fields for face recognition across pose and illumination. In *German Symposium on Pattern Recognition (DAGM)*, 2002.

[69] D. M. Hawkins and G. J. McLachlan. High-breakdown linear discriminant analysis. *Journal of the American Statistical Association*, 92:136–143, 1997.

[70] J. Hawkins and S. Blakeslee. *On Intelligence*. Times Books, 2004.

[71] X. He and W. K. Fung. High breakdown estimation for multiple populations with applications to discriminant analysis. *Journal of Multivariate Analysis*, 72(2):151–162, 2000.

[72] G. E. Hinton. Learning multiple layers of representation. *Trends in Cognitive Sciences*, 11(10):428–434, 2007.

[73] R. R. Honglak Lee, Roger Grosse and A. Y. Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. pages 609–616, 2009.

[74] F.-J. Huang and Y. LeCun. Large-scale learning with svm and convolutional nets for generic object categorization. In *IEEE Computer Vision and Pattern Recognition*, pages 284–291, 2006.

[75] M. Hubert and K. V. Driessen. Fast and robust discriminant analysis. *Computational Statistics and Data Analysis*, 45:301–320, 2003.

[76] J. Hup, A. James, B. Payne, S. Lomber, P. Girard, and J. Bullier. Cortical feedback improves discrimination between figure and background by v1, v2 and v3 neurons. *Nature*, 394:784–787, 1998.

[77] D. Huttenlocher and P. Felzenszwalb. Pictorial structures for object recognition. *International Journal of Computer Vision*, 61(1):55–79, 2005.

[78] M. Ito and H. Komatsu. Representation of angles embedded within contour stimuli in area v2 of macaque monkeys. *The Journal of Neuroscience*, 24(13):3313–3324, 2004.

[79] K. Jarrett, K. Kavukcuoglu, M. A. Ranzato, and Y. LeCun. What is the best multi-stage architecture for object recognition? In *IEEE International Conference on Computer Vision*, 2009.

[80] Y. Jin and S. Geman. Context and hierarchy in a probabilistic image model. In *IEEE Computer Vision and Pattern Recognition*, volume 2, pages 2145–2152, 2006.

[81] T. Joachims. Making large-scale support vector machine learning practical. pages 169–184, 1999.

[82] Z. Kourtzi, L. R. Betts, P. Sarkheil, and A. E. Welchman. Distributed neural plasticity for shape learning in the human visual cortex. *PLoS Biology*, 3(7), July 2005.

[83] Z. Kourtzi, A. S. Tolias, C. F. Altmann, M. Augath, and N. K. Logothetis. Integration of local features into global shapes: monkey and human fmri studies. *Neuron*, 37(2):333–346, January 2003.

[84] S. Krempp, D. Geman, and Y. Amit. Sequential learning of reusable parts for object detection. Technical report, CS Johns Hopkins, 2002.

[85] C. H. Lampert, M. B. Blaschko, and T. Hofmann. Beyond sliding windows: Object localization by efficient subwindow search. In *IEEE Computer Vision and Pattern Recognition*, 2008.

[86] D. D. Lee and H. S. Seung. Algorithms for non-negative matrix factorization. In *Neural Information Processing Systems*, volume 13, pages 556–562, 2001.

[87] T. S. Lee and D. Mumford. Hierarchical bayesian inference in the visual cortex. *J Opt Soc Am A Opt Image Sci Vis*, 20(7):1434–1448, 2003.

[88] B. Leibe, A. Leonardis, and B. Schiele. Robust object detection with interleaved categorization and segmentation. *International Journal of Computer Vision*, 77(1-3):259–289, 2008.

[89] A. Leonardis and H. Bischof. Robust recognition using eigenimages. *Computer Vision and Image Understanding*, 78(1):99–118, 2000.

[90] A. Leonardis and S. Fidler. Learning hierarchical representations of object categories for robot vision. In *ISRR 2007 : 13th International Symposium of Robotics Research*, pages 125–136, 2007.

[91] M. Leordeanu, R. Sukhtankar, and M. Hebert. Object and category recognition from pairwise relations between features. In *IEEE Computer Vision and Pattern Recognition*, 2007.

[92] J. Lindgren and A. Hyvrinen. Emergence of conjunctive visual features by quadratic independent component analysis. In *Neural Information Processing Systems*, 2006.

[93] X. Lu, Y. Wang, and A. K. Jain. Combining classifiers for face recognition. In *IEEE International Conference on Multimedia and Expo (ICME)*, volume III, 2003.

[94] O. L. Mangasarian and D. R. Musicant. Robust linear and support vector regression. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 22(2):950–955, 2000.

[95] G. L. Marcialis and F. Roli. Fusion of pca and lda for face verification. In *Proceedings of the ECCV Workshop on Biometric Authentication (BIOMET*, pages 30–37, 2002.

[96] M. Marszałek and C. Schmid. Constructing category hierarchies for visual recognition. In *European Conference on Computer Vision*, volume IV of *LNCS*, pages 479–491. Springer, 2008.

[97] A. M. Martinez and A. C. Kak. Pca versus lda. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 23(2):228–233, 2001.

[98] T. Masquelier and S. J. Thorpe. Unsupervised learning of visual features through spike timing dependent plasticity. *PLoS Computational Biology*, 3(2):832–838, 2007.

[99] G. Medioni, T. Fan, and R. Nevatia. Recognizing 3-d objects using surface descriptions. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 2(11):1140–1157, 1989.

[100] B. W. Mel and J. Fiser. Minimizing binding errors using learned conjunctive features. *Neural Computation*, 12(4):731–762, 2000.

[101] T. Melzer, M. Reiter, and H. Bischof. Appearance models based on kernel canonical correlation analysis. *Pattern Recognition*, 36(9):1961–1973, 2003.

[102] K. Mikolajczyk, B. Leibe, and B. Schiele. Multiple object class detection with a generative model. In *IEEE Computer Vision and Pattern Recognition*, pages 26–36, 2006.

[103] J. Mutch and D. G. Lowe. Multiclass object recognition with sparse, localized features. In *IEEE Computer Vision and Pattern Recognition*, pages 11–18, 2006.

[104] S. K. Nayar, H. Murase, and S. A. Nene. Parametric appearance representation. In *Early Visual Learning*, pages 131–160. Oxford University Press, 1996.

[105] R. C. Nelson and A. Selinger. Large-scale tests of a keyed, appearance-based 3-d object recognition system. *Vision Research, Special issue on computational vision*, 38(15-16):2469–2488, 1998.

[106] S. A. Nene, S. K. Nayar, and H. Murase. Columbia object image library (coil-20). Technical report, 1996.

[107] D. Nistér and H. Stewénius. Scalable recognition with a vocabulary tree. In *IEEE Computer Vision and Pattern Recognition*, volume 2, pages 2161–2168, 2006.

[108] B. Ommer and J. M. Buhmann. Learning compositional categorization models. In *European Conference on Computer Vision*, pages 316–329, 2006.

[109] B. Ommer and J. M. Buhmann. Learning the compositional nature of visual objects. In *IEEE Computer Vision and Pattern Recognition*, 2007.

[110] B. Ommer and J. Malik. Multi-scale object detection by clustering lines. In *IEEE International Conference on Computer Vision*, 2009.

[111] A. Opelt, A. Pinz, and A. Zisserman. Learning an alphabet of shape and appearance for multi-class object detection. *International Journal of Computer Vision*, 80(1):16–44, 2008.

[112] A. Pasupathy and C. Connor. Responses to contour features in macaque area v4. *Journal of Neurophysiology*, 82(5):2490–2502, 1999.

[113] A. Pasupathy and C. E. Connor. Population coding of shape in area v4. *Nature Neuroscience*, 5(12):1332–1338, 2002.

[114] N. Petkov. Biologically motivated computationally intensive approaches to image pattern recognition. *Future Generation Computer Systems*, 11(4-5):451–465, 1995.

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

[116] N. Pinto, D. Cox, and J. DiCarlo. Why is real-world visual object recognition hard? *PLoS Computational Biology*, 4(1):151–156, 2008.

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

[118] A. M. Pires. Robust linear discriminant analysis and the projection pursuit approach. In *Proceedings of ICORS*, 2001.

[119] M. A. Ranzato, F.-J. Huang, Y.-L. Boureau, and Y. LeCun. Unsupervised learning of invariant feature hierarchies with applications to object recognition. In *IEEE Computer Vision and Pattern Recognition*, 2007.

[120] L. Reddy and N. Kanwisher. Coding of visual objects in the ventral stream. *Current Opinion in Neurobiology*, 16(4):408–414, 2006.

[121] M. Riesenhuber and T. Poggio. Hierarchical models of object recognition in cortex. *Nature Neuroscience*, 2(11):1019–1025, 1999.

[122] E. T. Rolls and G. Deco. *Computational Neuroscience of Vision.* Oxford Univ. Press, 2002.

[123] H. Rom and G. G. Medioni. Hierarchical decomposition and axial shape description. *IEEE Trans. Pattern Analysis and Machine Intelligence*, (10):973–981, 1993.

[124] P. J. Rousseeuw. Multivariate estimation with high breakdown point. In *Mathematical Statistics and Applications*, volume B, pages 283–297. Reidel, Dordrecht, 1985.

[125] F. Samaria and A. Harter. Parameterisation of a stochastic model for human face identification. In *2nd IEEE Workshop on Applications of Computer Vision*, 1994.

[126] S. Sarkar and K. L. Boyer. *Computing Perceptual Organization in Computer Vision.* Singapore: World Scientific, 1994.

[127] F. Scalzo and J. H. Piater. Statistical learning of visual feature hierarchies. In *Workshop on Learning, CVPR*, 2005.

[128] A. Schoups, R. Vogels, N. Qian, and G. Orban. Practising orientation identification improves orientation coding in v1 neurons. *Nature*, (412):549–553, 2001.

[129] J. Schwartz and P. Felzenszwalb. Hierarchical matching of deformable shapes. In *IEEE Computer Vision and Pattern Recognition*, 2007.

[130] T. Serre, L. Wolf, S. Bileschi, M. Riesenhuber, and T. Poggio. Object recognition with cortex-like mechanisms. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 29(3):411–426, 2007.

[131] A. Shokoufandeh, L. Bretzner, D. Macrini, M. Demirci, C. Jonsson, and S. Dickinson. The representation and matching of categorical shape. *Computer Vision and Image Understanding*, Vol. 103:139–154, 2006.

[132] J. Shotton, A. Blake, and R. Cipolla. Efficiently combining contour and texture cues for object recognition. In *British Machine Vision Conference*, 2008.

[133] J. Shotton, A. Blake, and R. Cipolla. Multi-scale categorical object recognition using contour fragments. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 30(7):1270–1281, 2008.

[134] L. Sigal, S. Bhatia, S. Roth, M. J. Black, and M. Isard. Tracking loose-limbed people. volume 1, pages 421–428, 2004.

[135] J. Sivic, B. C. Russell, A. Zisserman, W. T. Freeman, and A. A. Efros. Unsupervised discovery of visual object class hierarchies. In *IEEE Computer Vision and Pattern Recognition*, 2008.

[136] D. Skočaj and A. Leonardis. Appearance-based localization using cca. In *Computer vision - CVWW '04 : proceedings of the 9th Computer Vision Winter Workshop*, pages 205–214, 2004.

[137] D. Skočaj, A. Leonardis, and S. Fidler. Robust estimation of canonical correlation coefficients. In *Digital imaging in media and education : 28th workshop of the Austrian Association for Pattern Recognition*, pages 15–22, 2004.

[138] I. Stainvas, N. Intrator, and A. Moshaiov. Improving recognition via reconstruction. Technical report, 1999.

[139] M. Stark and B. Schiele. How good are local features for classes of geometric objects? In *IEEE International Conference on Computer Vision*, 2007.

[140] E. Sudderth, A. Torralba, W. Freeman, and A. Willsky. Learning hierarchical models of scenes, objects, and parts. In *IEEE International Conference on Computer Vision*, pages 1331–1338, 2005.

[141] E. Sudderth, A. Torralba, W. T. Freeman, and A. Willsky. Describing visual scenes using transformed objects and parts. *International Journal of Computer Vision*, 77(1-3):291–330, 2008.

[142] D. L. Swets and J. Weng. Using discriminant eigenfeatures for image retrieval. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 18(8):831–837, 1996.

[143] K. Tanaka. Columns for complex visual object features in the inferotemporal cortex: Clustering of cells with similar but slightly different stimulus selectivities. *Cereb. Cortex*, (13):90–99, 2003.

[144] S. J. Thorpe, D. Fize, and C. Marlot. Speed of processing in the human visual system. *Nature*, 381:520–522, 1996.

[145] S. Todorovic and N. Ahuja. Unsupervised category modeling, recognition, and segmentation in images. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 2009.

[146] A. Torralba, K. P. Murphy, and W. T. Freeman. Sharing visual features for multiclass and multiview object detection. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 29(5):854–869, 2007.

[147] E. Trucco and A. Verri. *Introductory Techniques for 3-D Computer Vision*. Prentice Hall, 1998.

[148] J. Tsotsos. What roles can attention play in recognition? In *7th International Conference on Development and Learning*, 2008.

[149] J. K. Tsotsos. Analyzing vision at the complexity level. *Behavioral and Brain Sciences*, 13(3):423–469, 1990.

[150] K. Tsunoda, Y. Yamane, M. Nishizaki, and M. Tanifuji. Complex objects are represented in macaque inferotemporal cortex by the combination of feature columns. *Nature Neuroscience*, (4):832–838, 2001.

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

[152] S. Ullman. Object recognition and segmentation by a fragment-based hierarchy. *Trends in Cognitive Sciences*, 11:58–64, 2007.

[153] S. Ullman and B. Epshtein. *Visual Classification by a Hierarchy of Extended Features.* Towards Category-Level Object Recognition. Springer-Verlag, 2006.

[154] J. Utans. Learning in compositional hierarchies: Inducing the structure of objects from data. In *Neural Information Processing Systems*, pages 285–292, 1994.

[155] R. VanRullen. The power of the feed-forward sweep. *Advances in Cognitive Psychology*, 3(1–2):167–176, 2007.

[156] M. Varma and D. Ray. Learning the discriminative power-invariance trade-off. In *IEEE International Conference on Computer Vision*, 2007.

[157] M. Weber, M. Welling, and P. Perona. Towards automatic discovery of object categories. In *IEEE Computer Vision and Pattern Recognition*, pages 2101–2108, 2000.

[158] M. Wertheimer. Untersuchungen zur lehre von der gestalt. *Psychologische Forschung: Zeitschrift fuer Psychologie und ihre Grenzwissenschaften*, 4(1):301–350, 1923.

[159] L. Wolf, S. Bileschi, and E. Meyers. Perception strategies in hierarchical vision systems. In *IEEE Computer Vision and Pattern Recognition*, pages 2153–2160, 2006.

[160] J. Yang and J. Y. Yang. Why can lda be performed in pca transformed space? *Pattern Recognition*, 36:563–566, 2003.

[161] H. Yao, L. Shi, F. Han, H. Gao, and Y. Dan. Rapid learning in cortical coding of visual scenes. *Nature Neuroscience*, 10:772–778, 2007.

[162] K. Yu, W. Xu, and Y. Gong. Deep learning with kernel regularization for visual recognition. In *NIPS*, pages 1889–1896, 2008.

[163] W. Zhao, A. Krishnaswamy, R. Chellappa, D. Swets, and J. Weng. Discriminant analysis of principal components for face recognition. In *Face Recognition: From Theory to Applications*, pages 73–85. Springer-Verlag, 1998.

[164] L. Zhu, C. Lin, H. Huang, Y. Chen, and A. Yuille. Unsupervised structure learning: Hierarchical recursive composition, suspicious coincidence and competitive exclusion. In *European Conference on Computer Vision*, volume 2, pages 759–773, 2008.

[165] L. Zhu and A. Yuille. A hierarchical compositional system for rapid object detection. In *Neural Information Processing Systems*, 2005.

[166] S. Zhu and D. Mumford. A stochastic grammar of images. *Foundations and Trends in Computer Graphics and Vision*, 2(4):259–362, 2006.

Dodatek A

Povzetek disertacije v slovenskem jeziku

A.1 Uvod

Doktorska disertacija bo posvečena tematiki vizualnega razpoznavanja kategorij objektov iz dvodimenzionalnih naravnih slik. Vizualno razpoznavanje objektov je že vrsto desetletij zelo dejavno raziskovalno področje in predstavlja nekakšen "sveti gral" računalniškega vida. Napisanih je bilo večje število člankov, med drugim tudi vrsta preglednih člankov [117, 32, 42]. Eden izmed poglavitnih ciljev tovrstnega raziskovanja je podeliti robotom vizualne kompetence za opravljanje različnih kognitivnih nalog kot so npr. pomoč ostarelim in invalidom [1] ter opravljanje človeku nevarnih nalog. Poleg tega obstajajo tudi mnoge druge aplikacije kot so video nadzor, iskanje po slikovnih in video zbirkah, analiza medicinskih slik, itd.

Vizualno razpoznavanje objektov se po Trucco in Verri [147] razveji na štiri podprobleme, urejene po naraščajoči računski kompleksnosti:

1. *Ali je to objekt (kategorija) X?* Pri tem, najlažjem izmed vprašanj, imamo vnaprej podano identiteto/kategorijo objekta ter del slike (zaplata oz. *image patch* v ang.) na katerem se objekt nahaja. Čeprav se zdi problem enostaven, pa moramo upoštevati, da je lahko objekt različno osvetljen (kar vodi do velikih sprememb v izgledu) ter delno zakrit. V kolikor pa naš cilj ni prepoznava le specifičnega objekta, temveč tudi kategorijo, kateri ta pripada (steklenica Coca Cole proti kategoriji steklenic), je naloga še dodatno računsko otežena, saj se predstavniki objektov znotraj kategorije lahko med seboj precej razlikujejo tako po izgledu kot tudi obliki.

2. *Kateri je ta objekt (kategorija)?* Pri tem vprašanju imamo dano zaplato v sliki, ki

vsebuje natančno en objekt in naša naloga je, da ugotovimo kateri kategoriji objekt pripada. Lokacija objekta znotraj slike je torej podana, iskanje pa mora potekati preko celotnega nabora kategorij. Računska kompleksnost glede na prejšnji problem je tako n-krat večja, kjer n predstavlja število kategorij objektov, ki nas zanimajo.

3. *Kje v sliki je dani objekt (kategorija)?* Za dano sliko in podano kategorijo je potrebno ugotoviti, kje v sliki se objekt nahaja. Poleg lokacije moramo ugotoviti tudi velikost in orientacijo objekta. Računska kompleksnost glede na prvi problem je tako približno za faktor $300 \times 300 \times 10$ orientacij $\times$ 10 velikosti $\approx 10^7$ večja, kjer 300×300 predstavlja povprečno velikost slike.

4. *Kateri objekti (kategorije) so prisotni na dani sliki in kje v sliki se nahajajo?* To vprašanje je najbolj splošno in zahteva razpoznavo vseh objektov (kategorij) na dani sliki kot tudi določitev pripadajočih lokacij na sliki. Računska kompleksnost glede na prejšnji, tretji podproblem je tako povečana še za faktor n, kjer n predstavlja število kategorij objektov. V najbolj splošni obliki (kjer je n reda 10^4 ter objekti lahko nastopajo tudi v kateremkoli 3D pogledu) je problem dokazano računsko izjemno težek [149]. Izjemne zmožnosti človeškega vida pa kažejo na to, da učinkovite rešitve le obstajajo.

Doktorska disertacija je razdeljena na dva dela. V prvem delu se bomo osredotočili na problem 2, torej na razpoznavo objektov, pri čemer imamo podano njihovo lokacijo in velikost v sliki. V drugem, glavnem delu disertacije, pa se bomo ukvarjali s problemom, ki bi ga po kompleksnosti lahko uvrstili nekje med problema 3 in 4. Naš cilj bo razpoznava več kategorij objektov ter njihova lokalizacija v slikah, pri tem pa bo močan poudarek na razpoznavi večjega števila le-teh.

Pričujoči povzetek se nanaša na glavni del disertacije, torej na učenje hierarhičnih predstavitev kategorij objektov.

A.2 Povzetek

Naloga prepoznavanja in lokalizacije vizualnih kategorij objektov je poiskati lokacijo (ali celo segmentacijo) objektov na sliki in ugotoviti, katerim kategorijam pripadajo. Največji izziv predstavljata prepoznavanje in lokalizacija velikega števila kategorij v sprejemljivem časovnem okviru.

Analiza računske kompleksnosti je pokazala, da je problem računsko izjemno zahteven [149]. Vzrok za veliko kompleksnost je dejstvo, da se lahko objekti nahajajo na katerikoli lokaciji, v katerikoli velikosti, orientaciji in 3D pogledu na sliki. Poleg tega

so lahko objekti različno osvetljeni, artikulirani ali delno zakriti. Primerki objektov znotraj ene kategorije se poleg tega lahko vizualno zelo razlikujejo (npr. skodelice, psi, mačke, itd.), kar dodatno oteži našo nalogo. Seveda pa h kompleksnosti problema najbolj prispeva samo število kategorij, katerih število je v naravi nekaj desettisoč [28, 16]. Cilj naše disertacije bo razviti metodo, ki bo učinkovita za učenje, razpoznavo in lokalizacijo več kategorij objektov in bo, vsaj potencionalno, primerna za razpoznavo večjega števila kategorij.

Čeprav v literaturi obstaja kar nekaj uspešnih algoritmov za lokalizacijo posamične kategorije [44, 88, 153, 61, 103], je bilo raziskovanje razpoznavanja in lokalizacije več kategorij hkrati veliko redkejše [8, 111, 102]. Ta problem zaobjema tri med seboj povezane koncepte: računalniško predstavitev kategorij objektov, ki naj bi rasla logaritemsko s številom kategorij, metodo učenja predstavitve iz nabora slik, ter učinkovitega inferenčnega procesa, ki poišče ujemanje med računalniško predstavitvijo objekta ter tisto pridobljeno s slike.

Gradnja slovarjev vizualnih značilk je v literaturi trenutno najbolj uporabljen način predstavitve objektov in je vodila do nekaj najuspešnejših metod računalniškega vida do sedaj [44, 146, 111, 133, 46, 88]. Večina teh del pa uporablja enonivojske slovarje, kjer je vsak objekt predstavljen kot geometrijska kompozicija malega števila kompleksnih objektnih značilk.

Po drugi strani, hierarhije vpeljujejo strukturne odvisnosti med značilkami na več nivojih: objekti so definirani s svojimi deli, ki so naprej sestavljeni iz manjših komponent, le-ti še iz enostavnejših, itd. [21, 153, 166, 55, 109]. Takšne predstavitve omogočajo, da vizualno podobne kot tudi vizualno zelo različne kategorije objektov uporabljajo nekatere skupne značilke na različnih nivojih specifičnosti [149, 6, 109, 55]. Če želimo npr. modelirati avtomobile in žirafe, lahko uporabimo nekatere enostavnejše značilke kot so koti ali različne ukrivljenosti v predstavitvi obeh kategorij hkrati in tako zmanjšamo velikost skupne predstavitve. Uporaba skupnih značilk (ang. shareable features) med kategorijami pomeni delitev skupnih izračunov in vodi v pospešitev inferenčnega procesa [146].

V literaturi lahko zasledimo kar precejšnje število hierarhičnih kompozicionalnih pristopov za prepoznavanje kategorij objektov. Večina del pa temelji na ročno načrtovanih predstavitvah, vnaprej določenih pravilih grupiranja (kot so npr. Gestaltova pravila [158]) ali pa nadzorovanem učenju ročno označenih komponent objektov [66, 80, 38, 123, 121, 129, 126, 165, 166].

Učenje hierarhij brez ali le z manjšim nadzorom je primarnega pomena za predstavitve večjega števila kategorij objektov. Z učenjem minimiziramo čas potreben za ročno označevanje objektov ter njihovih komponent na slikah [166, 91] ter se izognemo pristranosti vnaprej določenih pravil grupiranja ali ročno načrtovanih značilk [131, 126,

66, 129]. Poleg tega nam statistično učenje predstavitve zagotovi značilke, ki so skupne več kategorijam, katerih morda ne moremo dobro predvideti z ročnim označevanjem na slikah [36]. Vendar pa je kompleksnost nenadzorovanega učenja hierarhične predstavitve velika: obstaja ogromno število možnih kombinacij značilk, število katerih raste eksponentno s številom hierarhičnih nivojev — zato moramo uporabiti učinkovit učni algoritem.

Raziskav o *nenadzorovanem hierarhičnem učenju* je bilo relativno malo. Večina del sodi na področje nevronskih mrež [74, 119, 98], ki pa se konceptualno precej razlikujejo od kompozicionalnih hierarhij [66, 51], ki jih bomo uporabili v naši disertaciji.

Eno izmed prvih del na tematiko nenadzorovanega učenja kompozicionalnih predstavitev je predstavil Utans [154]. Predlagana metoda je bila sposobna učenja kompozicij enostavnih vizualnih vzorcev točk.

Po vzoru modela Fukushime [64], sta Riesenhuber in Poggio [121] predlagala pristop HMAX, ki predstavi objekte z dvonivojsko hierarhijo kombinacij Gaborjevih značilk. Prvotni HMAX je uporabljal slovar vnaprej določenih značilk, kasneje pa so le-te zamenjali z naključno izbranimi značilkami s slik [103, 130]. Ker pristop ne uporablja statističnega učenja, potrebuje več tisoč značilk, da dobro predstavi kategorije objektov. Že za lokalizacijo ene same kategorije potrebuje metoda nekaj minut za sorazmerno majhno sliko.

Bouchard in Triggs [21] sta razširila *konstalacijski model* avtorjev Fergus et al. [44] na tronivojsko hierarhijo in podoben model je predlagal tudi Torralba et al. [146, 141]. Vsi ti modeli so zaradi kompleksnosti uporabljenih algoritmov prisiljeni uporabljati zelo redko slikovno informacijo, kjer se pred učnim procesom na sliki poišče majhno število (okoli 30) "zanimivih" točk (interest points v ang.). Vendar pa uporaba redke informacije in zelo diskriminativnih SIFT značilk onemogoča uspešnejše delovanje na slikah z veliko teksture, prekrivajočih se objektov, itd., in na strukturno enostavnih objektih. Ponovljivost značilk SIFT znotraj kategorij je tudi vprašljiva [130]. V disertaciji bomo uporabljali veliko bolj enostavne in številčne značilke, ki bodo omogočale večjo ponovljivost na slikah, lažje hierarhično kombiniranje, in zato pripomogle k večji učinkovitosti metode.

Epshtein in Ullman [153, 37, 152] sta predlagala grajenje hierarhije iz nasprotnega konca; hierarhični model se gradi od zgoraj navzdol, z rekurzivnim razstavljanjem slikovnih zaplat na manjše dele. Pristop je bil uporabljen pri učenju in lokalizaciji vsake kategorije posebej, medtem ko skupna predstavitev več kategorij ni bila zasledovana. Mikolajczyk et al. [102] so ravno tako predlagali hierarhični model kategorij, ki pa je zgrajen z gručenjem slikovnih zaplat glede na izgled in ne glede na geometrijo (obliko) objektov. Ker so slikovne zaplate precej rigidne glede na lokalne deformacije oblike, jih je potrebno uporabiti zelo veliko število (več milijonov), če želimo dobro predstaviti

objekte. Ravno tako je hierarhični učni pristop Hintona [72] vezan na izgled in ne na geometrijsko strukturo objekta.

Todorovic in Ahuja [145, 5] sta predlagala metodo, ki hierarhično predstavitev objekta zgradi avtomatsko na podlagi segmentacijskega algoritma. Ker so segmentacijski algoritmi ponavadi nestabilni (so zelo odvisni od osvetlitev, šuma, itd.), je potrebno generirati ogromno število hipotez hierarhičnih grupiranj, kar vodi v počasni učni kot tudi inferenčni proces.

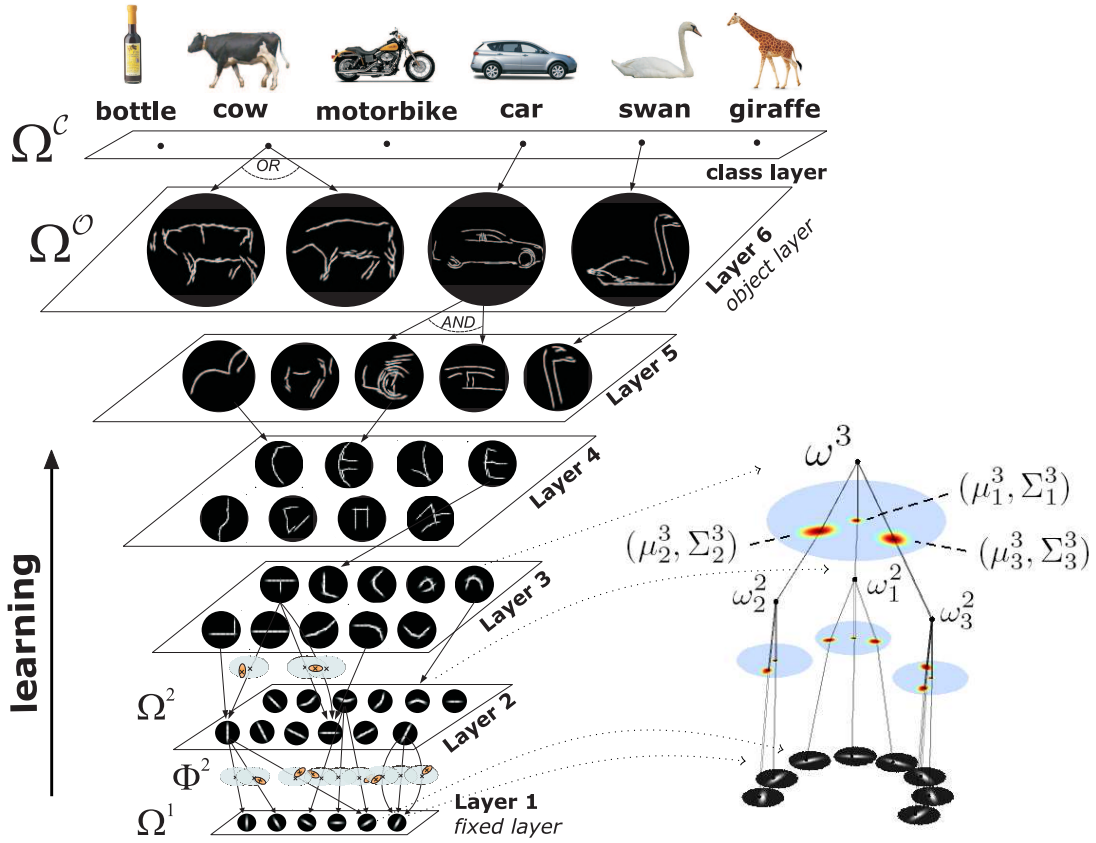
Ommer in Buhmann [109] sta predlagala metodo za nenadzorovano učenje hierarhične kompozicionalne predstavitve kategorij in sta jo uspešno uporabila pri problemu klasifikacije objektov. Značilke vsakega nivoja predlagane predstavitve so definirane kot histogrami značilk iz prejšnjega nivoja. Pri našem pristopu bomo eksplicitno modelirali prostorske relacije med posameznimi značilkami, kar bi moralo voditi do bolj zanesljive predstavitve oblik objektov ter tako tudi do bolj zanesljive lokalizacije kategorij na slikah.

Pristopi, ki so konceptualno najbolj podobni našemu so [61, 127] in [164]. Vendar pa vsi ti pristopi zgradijo hierarhično predstavitev za vsako kategorijo posebej in tako ne izkoristijo možnosti uporabe skupnih značilk kategorij, kar bi naredilo razpoznavanje in lokalizacijo več kategorij hkrati precej učinkovitejšo.

Na pristop, ki ga bomo razvili v disertaciji, je najbolj vplivalo delo avtorjev Amit et al. [6, 7, 8, 84], ki se ravno tako ukvarjajo z modeliranjem več kategorij s skupno hierarhično predstavitvijo njihovih oblik. Vendar pa se pristopa ločita v večih pogledih: značilke njihovega pristopa so narejene ročno, hierarhija je sestavljena le iz treh nivojev (konturni segmenti, deli ter celotni objekti), in aplikacija pristopa je namenjena predvsem branju registrskih tablic avtomobilov in ne razpoznavanju bolj splošnih, naravnih kategorij objektov.

Poleg omenjenih del so sorodna tudi dela, ki lokalizacijo objektov izvajajo s pomočjo hierarhičnih klasifikatorjev in drsečih oken (ang. sliding windows). V naši disertaciji pa bomo objekte predstavili s kompozicionalnim in generativnim modelom oblik, ki bo poleg učinkovitejše lokalizacije sposoben objekte na slikah tudi segmentirati.

Hierarhični pristop predlagan v doktorski disertaciji zajema tri pomembne aspekte: predstavitev vizualnih kategorij objektov (sekcija A.3), detekcija objektov v slikah (sekcija A.4), ter učenje hierarhične predstavitve z nabora učnih slik objektov (sekcija A.5).



Slika A.1: **Levo**: Ilustracija hierarhičnega slovarja oblike za predstavitev večih kategorij objektov. Prikazane oblike predstavljajo kompozicije ω^ℓ, medtem ko povezave med njimi določajo kompozicionalne relacije med njimi. **Desno**: Primer celotnega generativnega modela kompozicije z nivoja 3.

A.3 Predstavitev: Hierarhični kompozicionalni slovar oblike

Naš cilj je na računsko učunkovit način modelirati porazdelitve oblik objektov znotraj posameznih vizualnih kategorij. Ideja je predstaviti objekte z naučenim kompozicionalnim slovarjem oblike, ki ima naslednjo arhitekturo. Na vsakem nivoju slovarja imamo množico hierarhičnih fleksibilnih modelov, ki jim bomo rekli *kompozicije*. Vsaka kompozicija je definirana rekurzivno: je hierarhični generativni verjetnostni model, ki predstavlja prostorsko konfiguracijo med manjšim številom *delov* (enostavnejših ter manjših oblik), ki so ravnotako hierarhični verjetnostni modeli — kompozicije z nižjega nivoja hierarhije. Različne kompozicije lahko uporabljajo skupen nabor modelov za dele, kar naredi predlagan slovar učinkovit tako v velikosti (število modelov na vsakem nivoju) kot tudi v hitrosti razpoznave objektov s slik.

Vsak del v kompoziciji ima "izgled", katerega predstavimo z diskretno porazdelitvijo definirano nad množico kompozicij s prejšnjega nivoja. Prostorsko konfiguracijo delov bomo predstavili z relativnimi prostorskimi relacijami med vsemi deli kompozicije ter izbranim, takoimenovanim *referenčnim* delom znotraj kompozicije. Topologija vsake kompozicije bo drevo, kar pomeni, da se noben par delov v posamezni kompoziciji prostorsko ne prekriva. Različne kompozicije imajo lahko različne topologije, ki se jih bomo naučili iz podatkov *brez nadzora*, tako kot tudi vse parametre predstavitve. Število kompozicij na posameznem nivoju ne bo podano vnaprej ampak ravnotako ugotovljeno iz učnih podatkov. Predvidevali pa bomo nadzor pri učenju kategorij objektov: množica pozitivnih in validacijskih slik bo podana za vsako kategorijo objektov.

Na najnižjem, prvem, nivoju hierarhični slovar vsebuje manjše število enostavnih baznih modelov, ki predstavljajo orientirane robove v slikah. Število orientacij (baznih modelov) bo podano vnaprej. Na najvišjem nivoju slovarja kompozicije predstavljajo celotne oblike objektov. Za vsako vizualno kategorijo bomo uporabili več kompozicij, kjer bo vsaka izmed kompozicij predstavljala posamezno artikulacijo objektov znotraj kategorije ali pa 3D pogled. Slika A.1 prikazuje predlagano hierarhično predstavitev.

Uporabili bomo naslednje oznake. Z Ω bomo označili množico vseh kompozicij v slovarju, s Θ pa njihove pripadajoče parametre. Ker ima slovar hierarhično arhitekturo, ga bomo zapisali kot $\Omega = \Omega^1 \cup \Omega^2 \cup \cdots \cup \Omega^{\mathcal{O}} \cup \Omega^C$, kjer $\Omega^\ell = \{\omega_i^\ell\}_i$ predstavlja množico kompozicij na nivoju ℓ. Kadarkoli bo jasno iz konteksta, bomo opustili indeks i. Z $\Omega^{\mathcal{O}}$ bomo označili vrhnji, *objektni nivo* (v tej disertaciji je $\mathcal{O} = 6$). Končni, *kategorični nivo* Ω^C ni kompozicionalen, vendar predstavlja ALI operacijo med kompozicijami pripadajočimi posameznim kategorijam.

Definicija kompozicije glede na le prejšnji nivo je podobna predstavitvi *konstelacijskega modela* [43]. Kompozicija ω^ℓ, kjer je $\ell > 1$, je sestavljena iz P delov (število P je lahko različno za različne kompozicije) s porazdelitvami izgleda $\boldsymbol{\theta_{app}}^\ell = [{\theta_{app}}_j^\ell]_{j=1}^P$ ter porazdelitvami geometrijskih relacij (oz. njihovih parametrov) $\boldsymbol{\theta_g}^\ell = [{\theta_g}_j^\ell]_{j=1}^P$. Geometrijska relacija med posameznim delom znotraj kompozicije ter njenim referenčnim delom je modelirana z normalno porazdelitvijo s parametri ${\theta_g}_j^\ell = (\mu_j^\ell, \Sigma_j^\ell)$.

Bazni modeli na prvem nivoju so definirani nad slikovnimi značilkami, ki bodo v tej disertaciji n-dimenzionalni Gaborjevi filtri. Vsak model ω_i^1 predstavlja porazdelitev slikovnih značilk (n-dimenzionalni vektor), ki bo predstavljena z normalno porazdelitvijo s parametri (μ_i^1, Σ_i^1).

Modeli kategorij objektov so predstavljeni le s porazdelitvami izgleda. Kategorični model c ima tako porazdelitev ${\theta_{app}}_c^C$, ki je neničelna le za tiste kompozicije z objektnega nivoja, ki predstavljajo oblike objektov znotraj pripadajoče kategorije. To množico kompozicij objektnega nivoja bomo označili z $\{\omega_{i_c}^{\mathcal{O}}\}_{i_c} \subset \Omega^{\mathcal{O}}$, kjer je ${\theta_{app}}_c^C(\omega_{i_c}^{\mathcal{O}}) > 0$ za vsak i_c ter ${\theta_{app}}_c^C(\omega_{i'_c}^{\mathcal{O}}) = 0$ za vse preostale i'_c. Množici takšnih kompozicij za dve

različni kategoriji objektov bosta vedno disjunktni.

Potrebno je pripomniti, da lahko vsaka kompozicja generira oblike le znotraj okolice omejene velikosti. Vse kompozicije istega nivoja imajo enako veliko okolico, radij katere bomo označili s r^ℓ. Velikosti okolic r^ℓ se večajo eksponentno z nivojem hierarhije. Slika A.2 prikazuje postopno večanje kompleksnosti ter velikosti oblik kodiranih s kompozicijami iz hierarhije.

A.4 Prepoznavanje in detekcija objektov

Hierarhični model bomo najprej razložili s postopkom inference. Predvidevajmo torej, da je celotna predstavitev že znana in ja naša naloga prepoznati ter detektirati objekte kategorij na slikah. Učenje predstavitve pa bomo predstavili v sekciji A.5.

V razdelku A.4.1 bomo najprej predstavili slikovne značilke, ki jih bomo uporabili za detekcijo robov v slikah. V razdelku A.4.2 bomo predlagali teoretični postopek razpoznave skritih stanj modela ter opisali aproksimativni postopek za računsko učinkovito razpoznavo. Nato bomo v razdelku A.4.3 razložili kako na podlagi dobljenih stanj določimo kateri objekti se nahajajo na slikah.

A.4.1 Slikovne značilke

Naj I predstavlja dano testno sliko. Za detekcijo robov v sliki bomo uporabili nabor Gaborjevih filtrov:

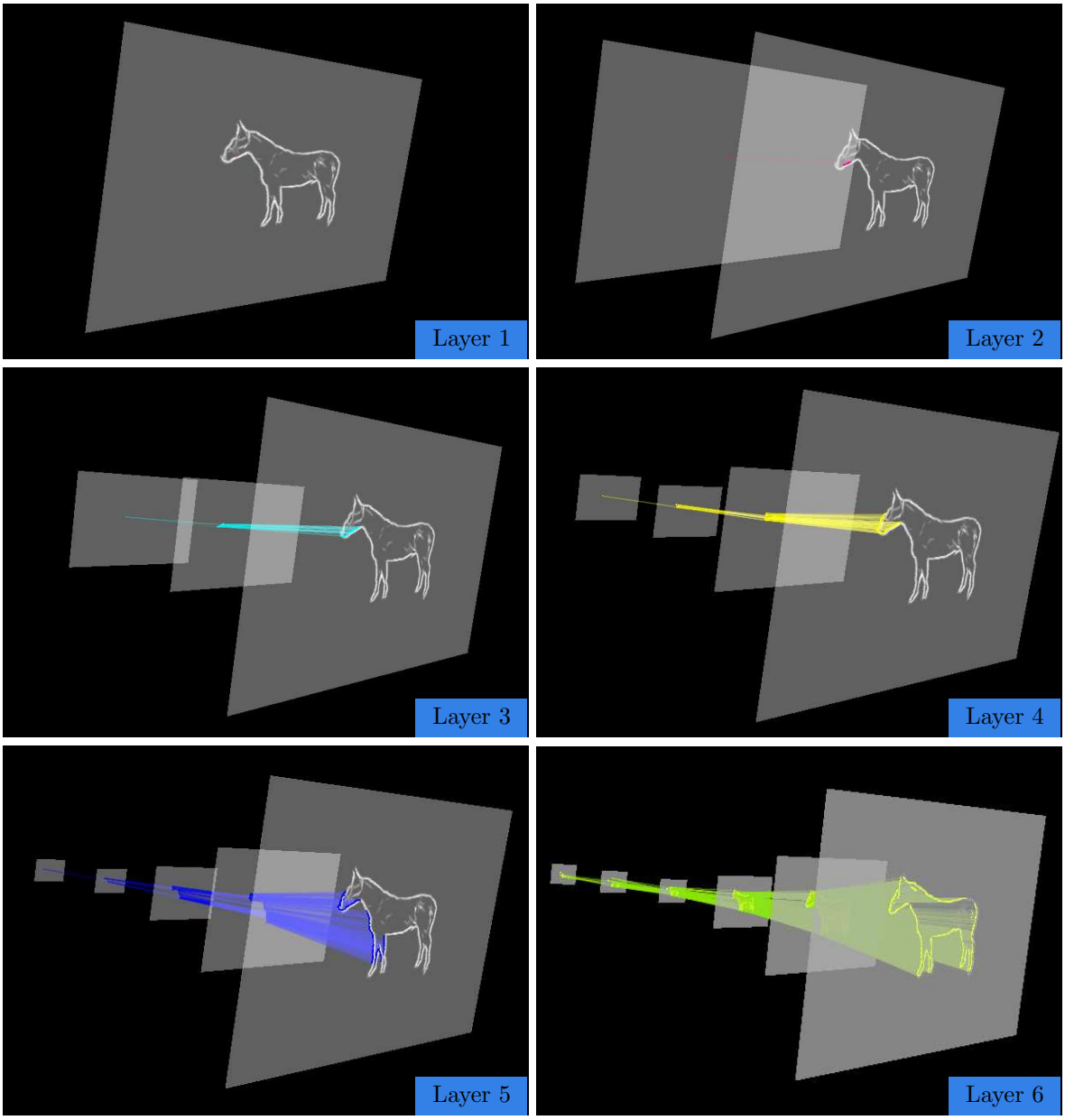
$$g_{\lambda,\varphi,\gamma,\sigma}(x,y,\psi) = \; e^{-\frac{u^2+\gamma^2v^2}{2\sigma^2}} \cos\big(\frac{2\pi u}{\lambda}+\varphi\big)$$
$$u = x\cos\psi - y\sin\psi, \quad v = x\sin\psi + y\cos\psi,$$

kjer (x,y) predstavlja center domene filtra. Uporabljamo naslednje vrednosti parametrov: $(\lambda,\gamma,\sigma) = (6, 0.75, 2)$. Uporabili bomo dva nabora filtrov, in sicer tistega s sodimi, $\varphi = 0$, ter lihimi, $(\varphi = -\frac{\pi}{2})$, Gaborjevimi jedri, definiranimi z n enakomerno porazdeljenimi orientacijami, $\psi = i\frac{\pi}{n}$, $i = 0, 1, \ldots, n-1$. Število smeri, n, je za potrebe te disertacije enak 6, glej tudi sliko A.3.

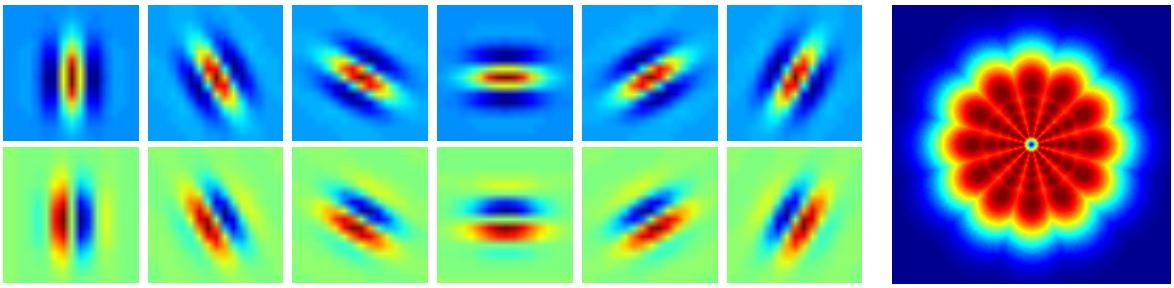
Nad sliko I izvedemo konvolucijo z množico filtrov ter izračunamo energijo za vsako posamezno orientacijo kot je predlagano v delu Petkova. [114],

$$\mathcal{E}(x,y,\psi) = \sqrt{r_0^2(x,y,\psi) + r^2_{-\pi/2}(x,y,\psi)}, \tag{A.1}$$

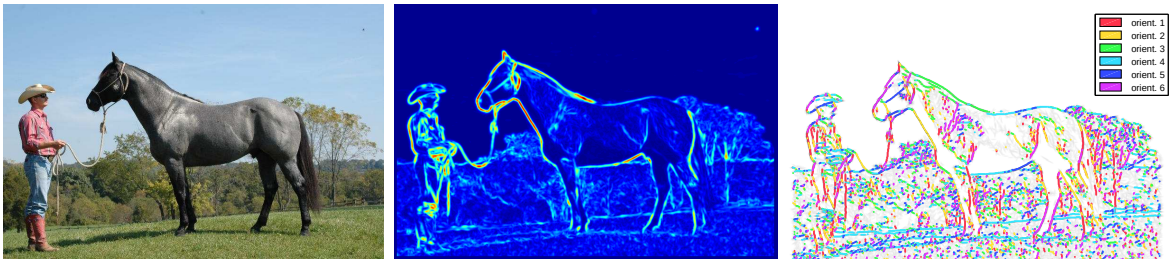
kjer sta $r_0(x,y,\psi)$ in $r_{-\pi/2}(x,y,\psi)$ rezultata konvolucije slike s sodimi in lihimi Gaborjevimi filtri v točki (x,y) in orientaciji ψ. Rezultat $\mathcal{E}$ normaliziramo tako, da ima največjo



Slika A.2: Slika prikazuje postopno večanje velikosti in kompleksnosti kompozicij z nivoji hierarhije.



Slika A.3: Primeri uporabljenih filtrov. Zgornja vrstica: Gaborjevi filtri s sodimi jedri. Spodnja vrstica: Gaborjevi filtri z lihimi jedri. Desno: Spekter množice filtrov.



Slika A.4: Levo: Originalna slika. Sredina: Maksimum čez nabor orientacij, $\arg\max_{\psi} E(x, y, \psi)$. Desno: Točke v katerih vzamemo Gaborjeve značilke $\mathbf{f}$. Različne barve označujejo različne dominantne orientacije značilk $\mathbf{f}$.

vrednost v sliki enako 1. V točkah lokalnih ekstremov (x, y) v $\mathcal{E}$, množico katerih bomo označili s $\mathbf{X} = \{(x, y)\}$, pa vzamemo Gaborjeve značilke $\mathbf{f} = \mathbf{f}(x, y)$, ki so definirane kot n-dimenzionalni vektorji vrednosti energij orientacij v (x, y), $\mathbf{f}(x, y) = \left[\mathcal{E}(x, y, i\frac{\pi}{n})\right]_{i=0}^{n-1}$. Množico vseh Gaborjevih značilk v sliki bomo označili s $\mathbf{F}$, kjer $\mathbf{F} = \{\mathbf{f}(x, y), (x, y) \in \mathbf{X}\}$. Slika A.4 prikazuje primer obdelave slike z Gaborjevimi filtri ter detekcije slikovnih značilk.

A.4.2 Inferenca

S postopkom inference bi radi izračunali hierarhijo skritih stanj na podlagi opazovanj (slikovne značilke $(\mathbf{F}, \mathbf{X})$). Skrita stanja prvega nivoja so edina, ki kot vhod dobijo podatke direktno iz opazovanj, vsa ostala stanja pa so izračunana na podlagi stanj prejšnjih nivojev.

Vsako skrito stanje z^ℓ je predstavljeno z dvema spremenljivkama $z^\ell = (\omega^\ell, x^\ell)$, kjer ω^ℓ predstavlja kompozicijo iz slovarja ter x^ℓ lokacijo v sliki (za krajšo in bolj pregledno notacijo bomo uporabljali le x namesto (x, y)). Ker je ω^ℓ v resnici verjetnostni model, prva spremenljivka predstavlja le referenco na model.

Ukvarjali se bomo s tremi problemi:

1. **Verjetje.** Izračun verjetja posamezne slikovne okolice J za izbrano kompozicijo ω^ℓ (hierarhični model): $p(J|\omega^\ell)$.

2. **Najverjetnejše drevo stanj.** Iskanje najbolj verjetnega drevesa stanj $\mathcal{T}^*(J, \omega^\ell)$, ki je generiralo izbrano slikovno okolico J glede na model ω^ℓ: $\mathcal{T}^*(J, \omega^\ell) = \arg\max_{\mathcal{T}} p(\mathcal{T}|J; \omega^\ell)$.

3. **Prepoznavanje in detekcija objektov večih kategorij.** Interpretacija slike v okviru naučenih kategorij objektov:

   (a) Iskanje slikovnih okolic J objektov ter kategorije c kateri pripadajo: $\{(J_i, c_i) : p(J_i|c_i) > \tau(c_i)$ in $J_i \cap J_k = \emptyset;\ \forall\,(i,k), i \neq k\}$, kjer je τ *prag* specifičen za vsako kategorijo, ki loči objekt od ozadja.

   (b) Določitev segmentacije objektov, ki bodo definirane z listi najbolj verjetnih dreves stanj $\mathcal{T}^*(J_i, c_i)$ za najdene objekte (J_i, c_i).

Računanje verjetij

Ker se bomo ukvarjali s slikami dosti večjimi kot so velikosti r^ℓ okolic kompozicij, je priročno definirati $I(x^\ell)$ kot *slikovno okolico* v sliki I, ki ima center v x^ℓ ter polmer (krožne) okolice enak r^ℓ,

$$I(x^\ell) = \{(\mathbf{f}, x) \in (\mathbf{F}, \mathbf{X}) :\ ||x - x^\ell||_2 \leq r^\ell\} \tag{A.2}$$

Pri tem okolica $I(x^\ell)$ ne vsebuje slikovnih elementov s slike, temveč vsebuje značilke v krožni okolici centrirani v x^ℓ.

Za dano kompozicijo ω^ℓ bi radi izračunali verjetje $p(I(x^\ell)|\omega^\ell)$ slikovne okolice $I(x^\ell)$ pri modelu ω^ℓ. Označimo z $z^\ell = (\omega^\ell, x^\ell)$ skrito stanje v točki x^ℓ (središče slikovne okolice). Potem so naslednje formulacije ekvivalentne in se bodo izmenjujoče uporabljale skozi vso disertacijo: $p(I(x^\ell)|\omega^\ell) = p(I(x^\ell), z^\ell|\omega^\ell) = p(I(x^\ell)|z^\ell)$. S $\mathcal{T}$ pa označimo drevo skritih stanj s korenom v stanju z^ℓ, ter z listi (layer 0), ki so opazovanja dobljena s slike. Za izračun verjetja moramo tako sešteti verjetnosti čez vsa možna takšna drevesa:

$$p(I(x^\ell)|\omega^\ell) = \sum_{\mathcal{T}} p(I(x^\ell), \mathcal{T}|\omega^\ell) \tag{A.3}$$

Ker je takšnih dreves eksponentno mnogo, je neposreden izračun izraza (A.3) neizračunljiv v praksi. Vendar pa lahko izkoristimo rekurzivno definicijo kompozicij, da dobimo rekurzivni izraz tudi za izračun verjetja.

Najprej napišimo porazdelitev na desni strani izraza. S $\mathcal{T}^{\ell'}$ označimo množico stanj na nivoju ℓ' drevesa $\mathcal{T}$, ter s $\mathcal{T}(z^{\ell'})$ pod-drevo drevesa $\mathcal{T}$, ki ima koren v $z^{\ell'} \in \mathcal{T}^{\ell'}$. Naj $T^{\ell-1} = [z_j^{\ell-1} := (\omega_j^{\ell-1}, x_j^{\ell-1}))]_{j=1}^{|\mathcal{T}^{\ell-1}|}$ predstavlja množico stanj na nivoju $\ell-1$ drevesa $\mathcal{T}$ (torej le en nivo nižje od korena). Porazdelitev $p(I(x^\ell), \mathcal{T}|\omega^\ell)$ bomo zapisali rekurzivno. Najprej jo razcepimo na naslednji način:

$$p(I(x^\ell), \mathcal{T}|\omega^\ell) = \prod_{j=1}^{P} \underbrace{p_F(I(x^\ell), \mathcal{T}(z_j^{\ell-1}), z_j^{\ell-1}|z^\ell)}_{\text{ospredje pod delom } j} \cdot \underbrace{\prod_{k=P+1}^{|\mathcal{T}^{\ell-1}|} p_B(I(x^\ell), \mathcal{T}(z_k^{\ell-1}), z_k^{\ell-1}|z^\ell)}_{\text{ozadje}} \tag{A.4}$$

Izraz za ospredje lahko razcepimo naprej:

$$p_F(I(x^\ell), \mathcal{T}(z_j^{\ell-1}), z_j^{\ell-1}|z^\ell) = p_F(z_j^{\ell-1}|z^\ell) \cdot p_F(I(x^\ell), \mathcal{T}(z_j^{\ell-1})|z_j^{\ell-1}; \omega^\ell) \tag{A.5}$$

Ker glede na nivo ℓ nižji nivoji postanejo pogojno neodvisni, je drugi izraz enak:

$$p_F(I(x^\ell), \mathcal{T}(z_j^{\ell-1})|z_j^{\ell-1}; \omega^\ell) = p(I(x_j^{\ell-1}), \mathcal{T}(z_j^{\ell-1})|\omega_j^{\ell-1}) \tag{A.6}$$

Glede na našo definicijo kompozicij, se izraz $p_F(z_j^{\ell-1}|z^\ell)$ razcepi na izgled ter geometrijo na naslednji način:

$$p_F(z_j^{\ell-1}|z^\ell) = \underbrace{p(\omega_j^{\ell-1} \mid \theta_{app_j}{}^\ell)}_{\text{izgled}} \underbrace{p(x_j^{\ell-1} \mid x^\ell; \theta_{g_j}{}^\ell)}_{\text{geometrija}} = \theta_{app_j}{}^\ell(\omega_j^{\ell-1})\, \mathcal{N}(x_j^{\ell-1} - x^\ell \mid \mu_j^\ell, \Sigma_j^\ell) \tag{A.7}$$

kjer je prvi izraz verjetnost, da ima j-ti del pri modelu ω^ℓ izgled $\omega_j^{\ell-1}$. Drugi izraz pa predstavlja geometrijsko kompatibilnost med lokacijo stanja $z_j^{\ell-1}$ in lokacijo x^ℓ pri modelu za j-ti del znotraj kompozicije.

Podobno kot osprednji del v A.5 lahko razcepimo tudi izraz za ozadje v (A.4):

$$\begin{aligned} p_B(I(x^\ell), \mathcal{T}(z_k^{\ell-1}), z_k^{\ell-1}|\omega^\ell) &= p_B(z_k^{\ell-1}|z^\ell)\, p_B(I(x_k^{\ell-1}), \mathcal{T}(z_k^{\ell-1})|z_k^{\ell-1}; \omega^\ell) \\ &= \alpha \cdot \big(1 - p(I(x_k^{\ell-1}), \mathcal{T}(z_k^{\ell-1})|\omega_k^{\ell-1})\big), \end{aligned} \tag{A.8}$$

kjer $p_B(z_k^{\ell-1}|z^\ell)$ predstavlja verjetnost, da stanje $z_k^{\ell-1}$ pripada ozadju (vzeli bomo, da je enak α za vse kompozicije). Za verjetje opazovanj pri modelu za ozadje pa smo vzeli $1-$verjetje, da isto opazovanje pripada ospredju.

Porazdelitev v (A.4) lahko sedaj zapišemo rekurzivno:

$$\begin{aligned} p(I(x^\ell), \mathcal{T}|\omega^\ell) = &\Big(\prod_{j=1}^{P} \theta_{app_j}{}^\ell(\omega_j^{\ell-1})\, \mathcal{N}(x_j^{\ell-1} - x^\ell \mid \mu_j^\ell, \Sigma_j^\ell) \cdot p(I(x_j^{\ell-1}), \mathcal{T}(z_j^{\ell-1})|\omega_j^{\ell-1}) \Big) \\ &\cdot \Big(\alpha^{N_B} \prod_{k=P+1}^{|\mathcal{T}^{\ell-1}|} \big(1 - p(I(x_k^{\ell-1}), \mathcal{T}(z_k^{\ell-1})|\omega_k^{\ell-1})\big) \Big), \end{aligned} \tag{A.9}$$

kjer je $N_B = |\mathcal{T}^{\ell-1}| - P$ število stanj v $\mathcal{T}^{\ell-1}$, ki pripada ozadju.

Zaradi rekurzivnega zapisa zgornje porazdelitve, lahko hitro dobimo tudi rekurzivni zapis verjetja v (A.3). Definirajmo $q(I, z) = p(I(x)|\omega)$, kjer je $z = (\omega, x)$. Z manipulacijo vsot v (A.3) in v (A.9), dobimo

$$q(I, z^\ell) = \sum_{\mathcal{T}^{\ell-1}} \Big[\Big(\prod_{j=1}^{P} {\theta_{app}}_j^{\ell}(\omega_j^{\ell-1}) \, \mathcal{N}(x_j^{\ell-1} - x^\ell \mid \mu_j^\ell, \Sigma_j^\ell) \cdot q(I, z_j^{\ell-1}) \Big) \cdot \alpha^{N_B} \prod_{k=P+1}^{|\mathcal{T}^{\ell-1}|} (1 - q(I, z_k^{\ell-1})) \Big] \tag{A.10}$$

Rekurzijo zaključimo s členom prvega nivoja:

$$q(I, z^1) = p(I(x^1) \mid z^1) = p(\mathbf{f}(x) \mid z^1) = \mathcal{N}(\mathbf{f} \mid \mu^1, \Sigma^1), \tag{A.11}$$

pri čemer je $q(I, z^1)$ verjetnost, da je značilka (6-dimenzionalni vektor) $\mathbf{f}$ generirana z modelom ω^1.

Verjetja vseh stanj v hierarhiji lahko izračunamo učinkovito z dinamičnim programiranjem, podobno kot to naredijo v "forward" algoritmu za hierarhične skrite markovske modele (ang., hierarchical hidden markov models) [57]. V tej disertaciji predlagamo še dodatne pospešitve, ki so podrobneje razložene v angleškem delu disertacije (razdelek 5.2.2).

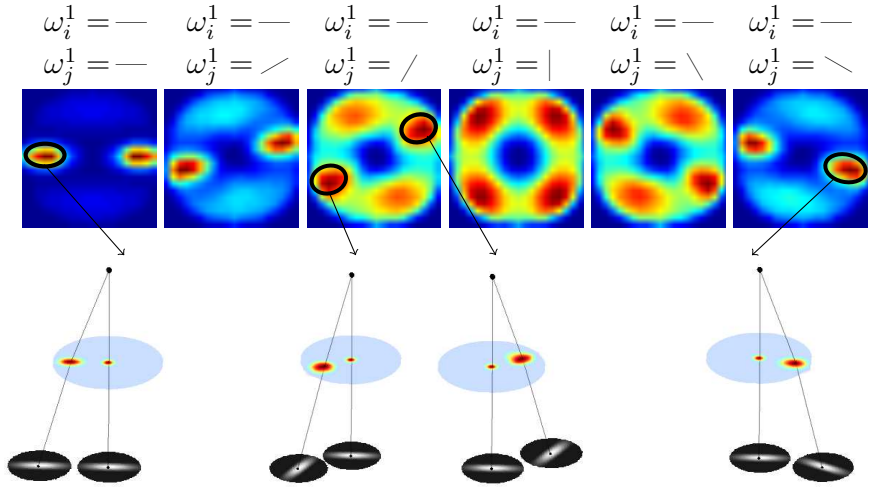
Iskanje najbolj verjetnega drevesa stanj

Pri danem modelu (kompoziciji) ω^ℓ bi radi poiskali najbolj verjetno drevo stanj, ki je generiralo slikovno okolico $I(x^\ell)$:

$$\mathcal{T}^*(z^\ell) = \arg\max_{\mathcal{T}} p(\mathcal{T}|I(x^\ell); \omega^\ell) = \arg\max_{\mathcal{T}} p(I(x^\ell), \mathcal{T}|\omega^\ell) \tag{A.12}$$

V kolikor definiramo $\delta(I, z) = \max_{\mathcal{T}} p(I(x), \mathcal{T}|\omega)$, lahko zapišemo rekurzijo za $\delta(I, z^\ell)$ podobno kot v (A.10), le da zamenjamo vsoto z maksimumom. Za prvi nivo imamo $\delta(I, z^1) = \mathbf{f}(x^1)$. Drevo $\mathcal{T}$ lahko učinkovito poiščemo z dinamičnim programiranjem, podobno Viterbijevemu algoritmu za hierarhične HMMje: $\delta(I, z^\ell)$ izračunamo postopno, od prvega do višjih nivojev, pri tem pa pomnimo kazalce na stanja prejšnjega nivoja, ki so vodila do maksimuma. Celotno drevo $\mathcal{T}^*$ poiščemo tako, da sledimo kazalcem od korena nazaj k listom.

Listom drevesa $\mathcal{T}^*(z^\ell)$ pravimo *nosilec* skritega stanja z^ℓ ter jih označili s $\text{supp}(z^\ell)$.



Slika A.5: **Zgoraj**: Primeri naučnih histogramov $h_{ij}^{\ell=2}$. **Spodaj**: Primeri dupletov dobljenih iz modusov naučenih histogramov.

A.4.3 Prepoznava objektov na slikah

Ko imamo poračunana verjetja stanj v hierarhiji, je prepoznava objektov enostavna. Odločitev ali je objekt na sliki ali ne naredimo tako, da pogledamo ali je verjetje na najvišjem (objektnem) nivoju večje od praga značilnega za dano kategorijo (vse prage se naučimo v postopku učenja). Več podrobnosti o prepoznavi in segmentaciji objektov na slikah je podanih v angleškem delu disertacije.

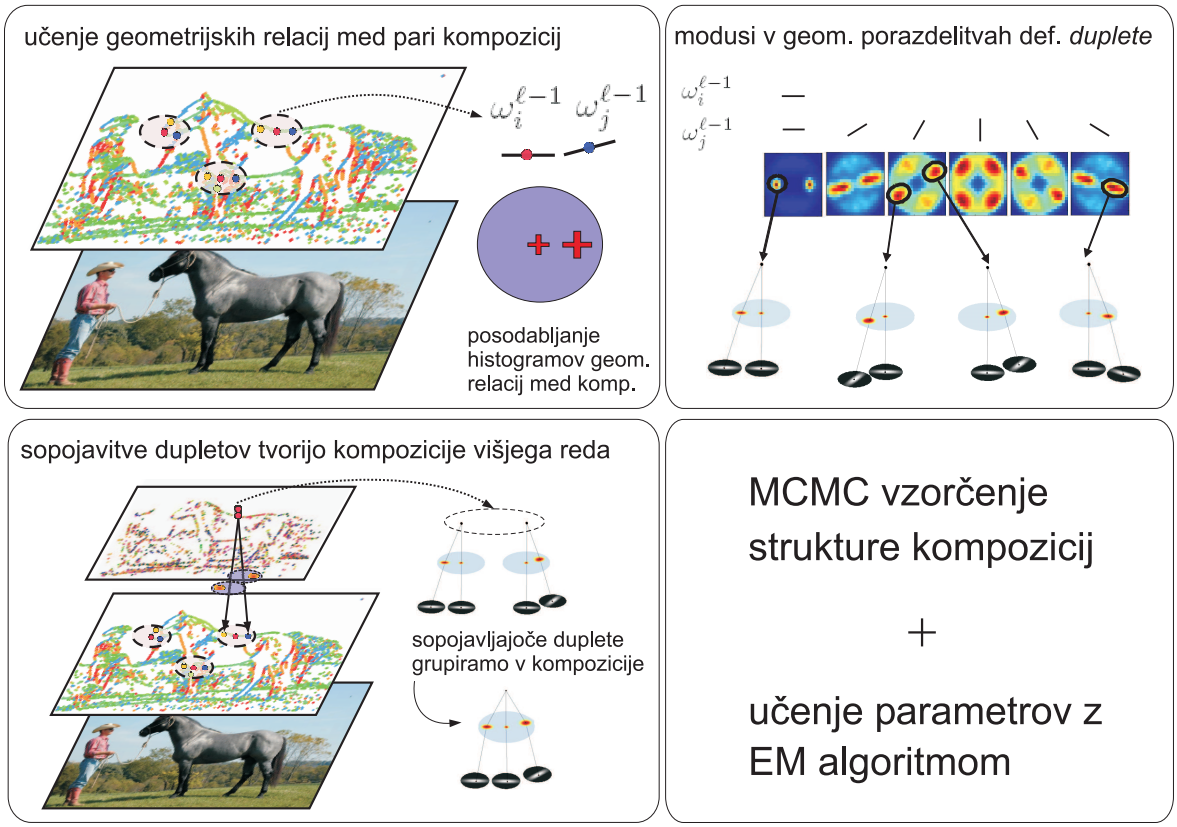
A.5 Učenje

Slovar kompozicij ℓ-tega nivoja bomo naučili z maksimizacijo verjetja:

$$\begin{aligned} L(D;\Omega^\ell,\Theta^\ell) &= \sum_{k=1}^{N}\sum_{i}\log p(I_k(x_i^\ell)\mid\Omega^\ell,\Theta^\ell) - \lambda^\ell\,|\Omega^\ell| \\ &= \sum_{k=1}^{N}\sum_{i}\log\sum_{\mathcal{T}_{ki},\omega_{ki}^\ell} p(I_k(x_i^\ell),\mathcal{T}_{ki},\omega_{ki}^\ell|\Theta^\ell) - \lambda^\ell\,|\Omega^\ell| \end{aligned} \tag{A.13}$$

Ker ne poznamo niti števila niti strukture kompozicij (število delov ter parametri), je maksimizacija izraza (A.13) neizračunljiva v praksi. Zato bo naša strategija učenja naslednja:

1. Najprej se bomo naučili geometrijske relacije med *pari* kompozicij prejšnjega nivoja.



Slika A.6: Učenje slovarja na posameznem nivoju hierarhije.

2. Poiskali bomo moduse naučenih porazdelitev preko katerih bomo definirali takoimenovane *duplete* (kompozicije sestavljene iz dveh delov). Postopek je predstavljen na sliki A.5.

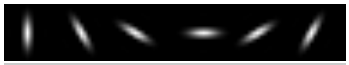
3. Poiskali bomo pogosto so-pojavljajoče duplete, ki bodo definirali kompozicije višjih redov.

4. S požrešno metodo bomo poiskali podmnožico kompozicij iz točke (3), nato pa uporabili stohastično MCMC optimizacijo za iskanje optimalnega slovarja kompozicij na nivoju ℓ.

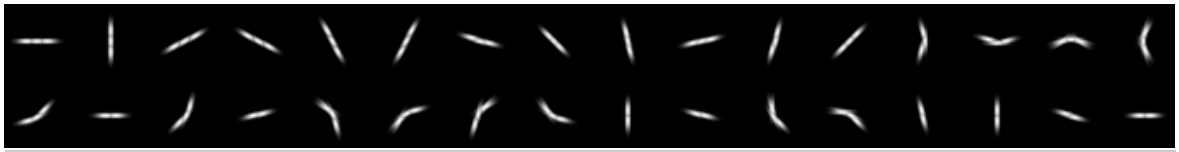
Celotni učni pristop je ilustriran na sliki A.6, podrobnosti pa so podane v angleškem delu disertacije (razdelek 5.3).

A.6 Eksperimentalni rezultati

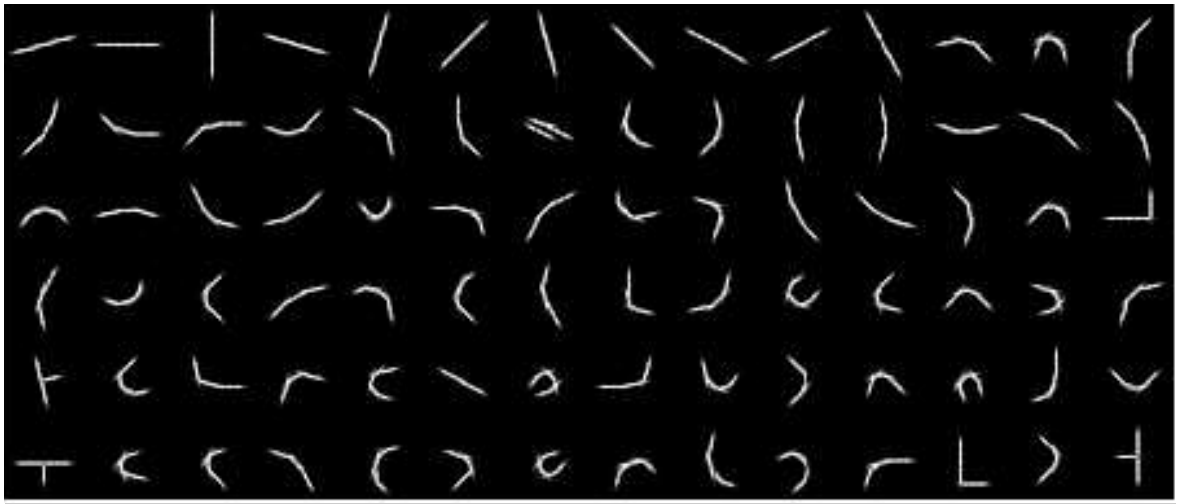
Metodo smo preskusili na 15-tih kategorijah objektov ter več znanih podatkovnih bazah računalniškega vida. Slovar smo naučili na naslednji način: spodnje nivoje (nivoje



Nivo 1



Nivo 2



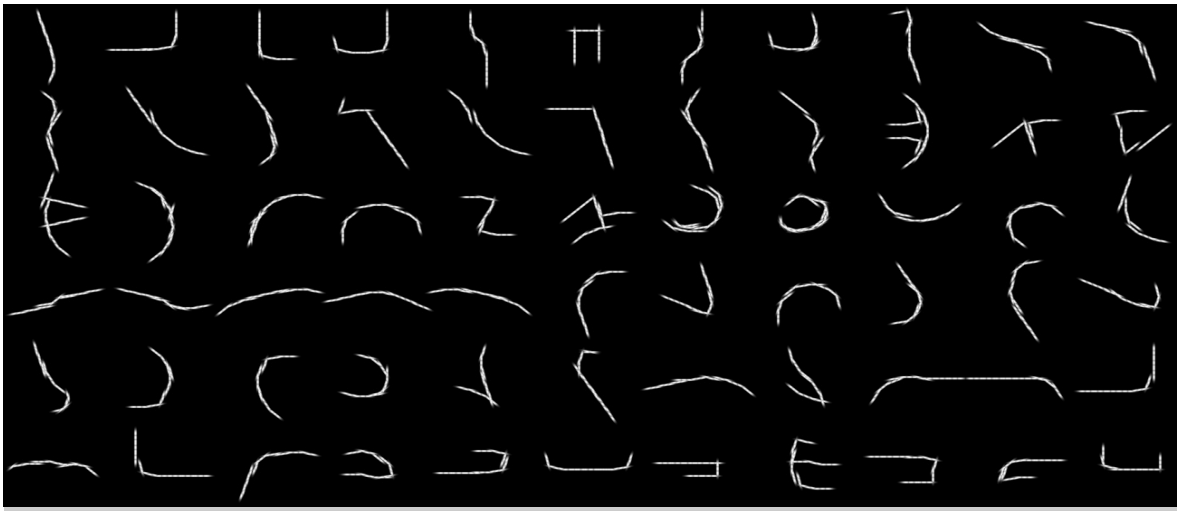
Nivo 3

Slika A.7: Prvi trije nivoji hierarhičnega slovarja oblik naučenega na 1500 naravnih slikah. Naučene oblike vključujejo različne ukrivljenosti ter kote, kar so bila dosedaj vnaprej določena pravila grupiranja Gestalt teorije [158].

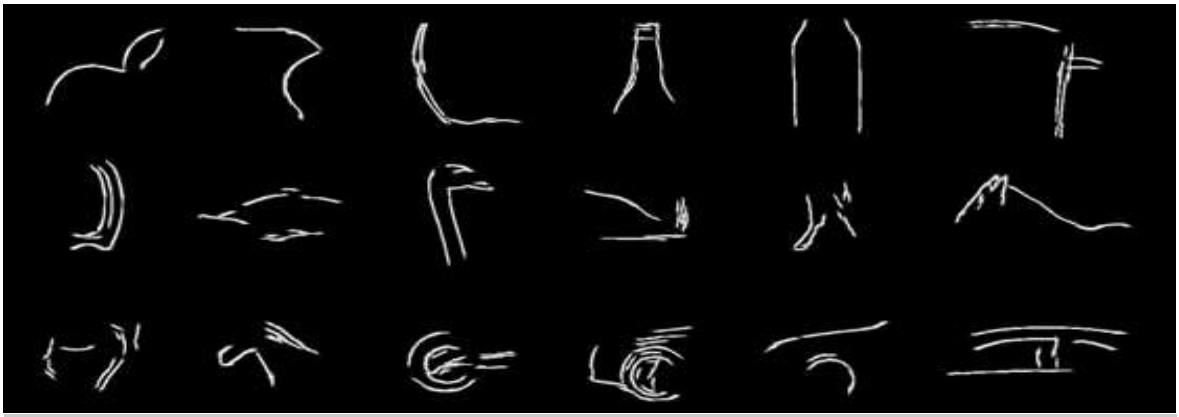
od 2 do 3) smo naučili na naravnih slikah, višje nivoje pa smo učili postopoma, torej kategorijo za kategorijo. Primeri naučenih kompozicij so prikazani na slikah A.7 in A.8.

Rezultati natančnosti so predstavljeni v tabeli A.1. Metoda je primerljiva z rezultati trenutno najboljših metod. Prednost naše metode je predvsem v računski učinkovitosti. Čas razpoznave na sliko za eno kategorijo je približno $2-4$ sekunde, medtem ko druge metode potrebujejo vsaj 5 do 10-krat več časa. Najpomembnejše pa je obnašanje metode v odvisnosti od števila naučenih kategorij, ki je prikazano na sliki A.10. Ko število kategorij narašča, je čas potreben za razpoznavo sublinearen v številu kategorij. To je prvi takšen rezultat dobljen z generativno metodo do sedaj.

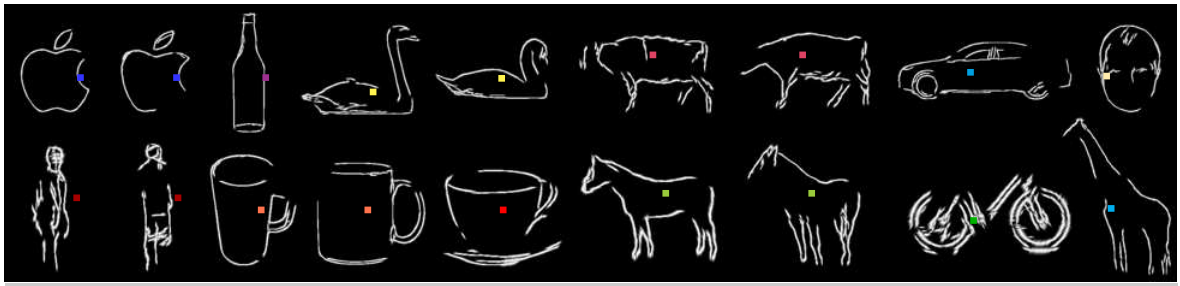
Primeri razpoznanih objektov so podani na sliki A.11. Potrebno je poudariti, da naša metoda ne vrne le okvirja okoli razpoznanega objekta, temveč je sposobna tudi segmentacije ter razpoznave delov objekta na več nivojih granularnosti.



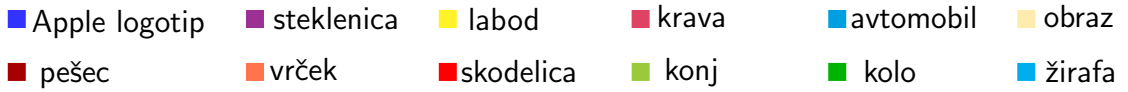
Nivo 4



Nivo 5



Nivo $\mathcal{O} = 6$ - objektni nivo



Slika A.8: Primeri naučenih kompozicij z višjih nivojev hierarhičnega kompozicionalnega slovarja.

Tabela A.1: Natančnost razpoznave metode na več standardnih bazah. Na ETH shape in INRIA poročamo natančnost detekcije (v %) pri maksimalni stopnji 0.4 napačnih pozitivnih razpoznavah. Za vse ostale baze pa poročamo natančnost razponave pri enaki stopnji napačnih pozitivnih in napačnih negativnih razpoznanih objektih (ang. recall at equal-error-rate (EER)).

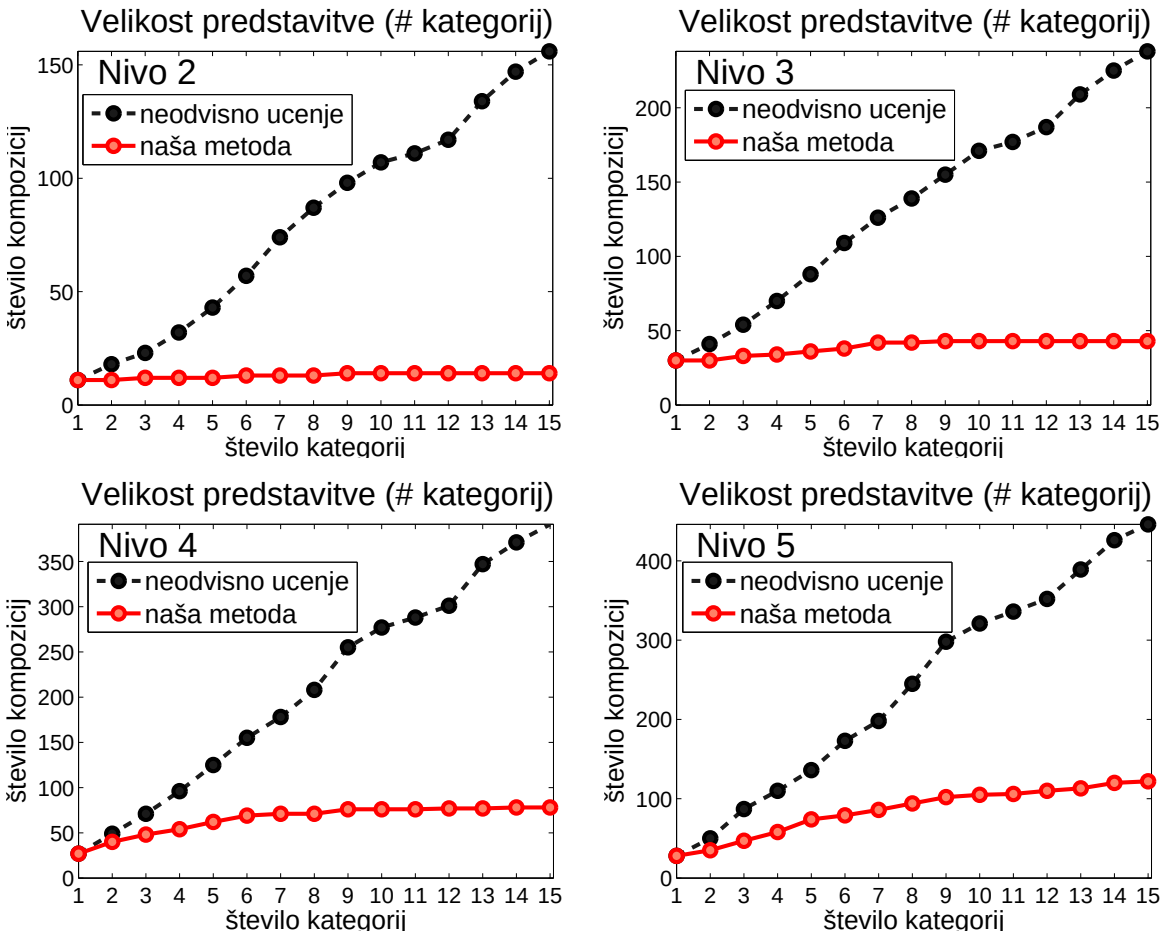
	kategorija	**[45]**	**[63]**	**naš pristop**	
ETH shape	Apple logotip	83.2 (1.7)	89.9 (4.5)	87.3 (2.6)	0.32 FPPI
	steklenica	83.2 (7.5)	76.8 (6.1)	**86.2** (**2.8**)	**0.36** FPPI
	žirafa	58.6 (14.6)	90.5 (5.4)	83.3 (4.3)	0.21 FPPI
	vrček	83.6 (8.6)	82.7 (5.1)	**84.6** (**2.3**)	**0.27** FPPI
	labod	75.4 (13.4)	84.0(8.4)	78.2 (5.4)	0.26 FPPI
	povprečje	76.8	84.8	83.7	0.28 FPPI
INRIA	konj	84.8(2.6)	/	**85.1**(**2.2**)	**0.37** FPPI

	kategorija	**sorodna dela**		**naš pristop**
UIUC	avtomobil	90.6 [103]	93.5 [5]	**93.5**
Weizmann	konj	89.0 [133]	93.0 [132]	**94.3**
TUD	motor	87 [88]	88 [102]	83.2

	kategorija	**[111]**	**[133]**	**naš pristop**
GRAZ	obraz	96.4	97.2	94
	kolo	72	67.9	68.5
	steklenica	91	90.6	89.1
	krava	100	98.5	96.9
	skodelica	81.2	85	**85**
	avto_spredaj	90	70.6	76.5
	avto_zadaj	97.7	98.2	97.5
	konj	91.8	93.7	**93.7**
	motor	95.6	99.7	93.0
	vrček	93.3	90	90
	pešec	52.6	52.4	**60.4**

A.7 Zaključek

Doktorsko delo je razdeljeno na dva dela. V prvem delu smo predstavili metodo za kombiniranje diskriminativnih in rekonstrukcijskih metod za klasifikacijo objektov v primeru, ko so le-ti delno zakriti. Uporabnost metode za robustno klasifikacijo smo prikazali na več primerih računalniškega vida kot so razpoznavanje objektov z več 3D

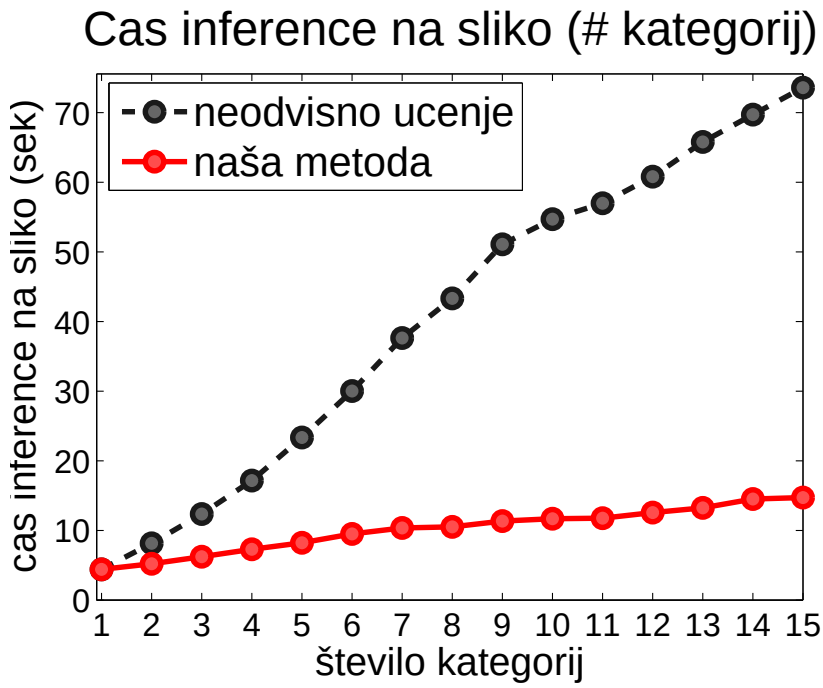


Slika A.9: Velikost slovarja v odvisnosti števila naučenih kategorij

pogledov, razpoznavanje obrazov ter lokalizacija mobilnega robota. Pokazali smo, da je metoda uspešna tudi, ko je večji del objekta zakrit.

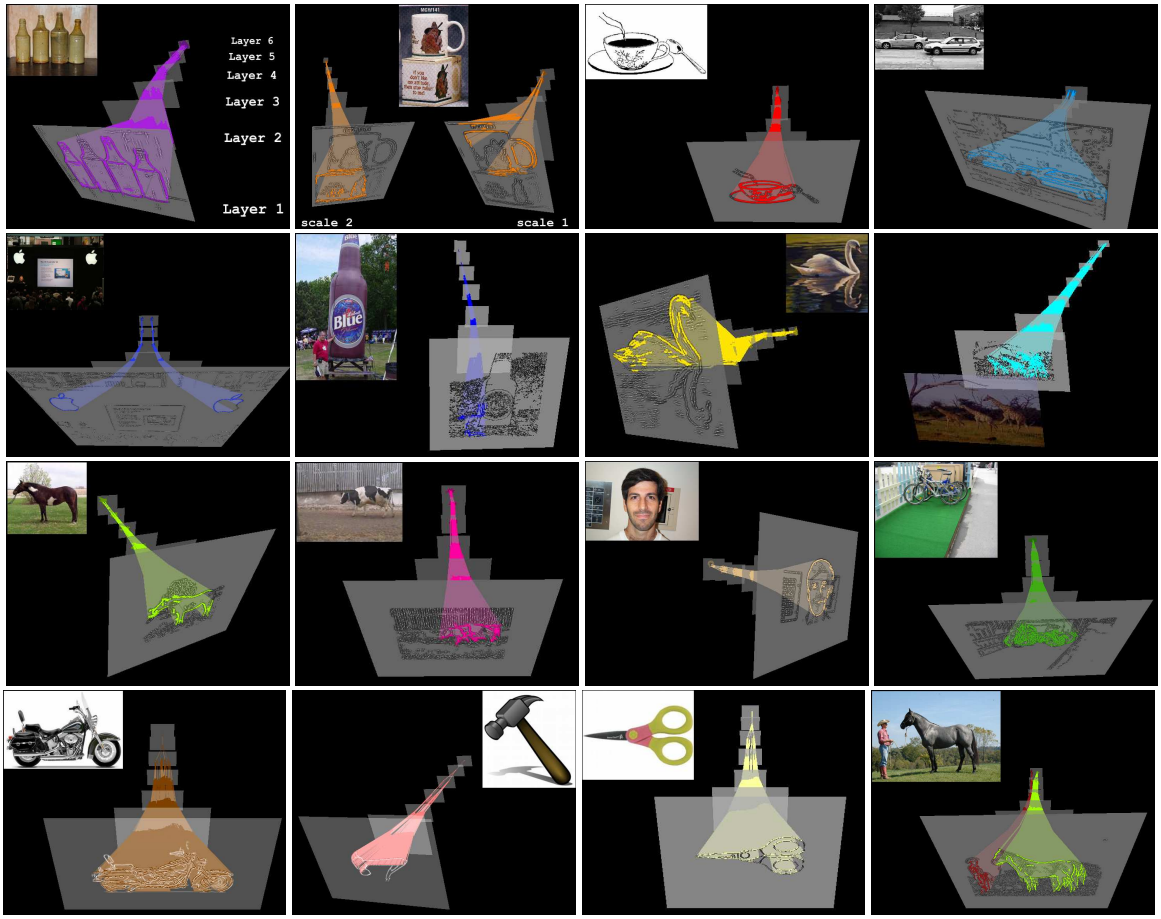
V drugem delu doktorske disertacije smo predlagali nov pristop za učenje hierarhičnega kompozicionalnega slovarja oblik za predstavitev več vizualnih kategorij objektov. Učenje poteka od spodaj navzgor, od enostavnih orientiranih robov do celotnih oblik objektov. Slovar učimo rekurzivno s kombiniranjem kompozicij prejšnjega nivoja preko prostorskih relacij.

Eksperimentalni rezultati so pokazali, da se je metoda sposobna nenadzorovano naučiti generične model oblik z naravnih slik ter jih uspešno uporabiti za klasifikacijo. Še več, našo metodo smo uporabili za učenje več kategorij objektov (15) in pokazali, da se obnaša sublinearno v odvisnosti števila naučenih kategorij. To je prvi tovrsten rezultat dobljen z generativnim modelom do sedaj in prikazuje potencialno možnost uporabe hierarhij za razpoznavo velikega števila kategorij. Poleg tega je metoda sposobna razpoznave objektov na več nivojih granularnosti, kar je predvsem pomembno za po-



Slika A.10: Povprečni čas razpoznave na sliko v odvisnosti od števila naučenih kategorij objektov.

dročje robotike, kjer mora biti sistem poleg razpoznave sposoben tudi manipulacije z objekti.



Slika A.11: Primeri razpoznanih objektov. Povezave prikazujejo najbolj verjetna drevesa stanj pri danih razpoznavah objektov. Od levo zgoraj: *steklenica*, *vrček*, *skodelica*, *avtomobil*, *Apple logotip*, *Apple logotip* (napačna pozitivna razpoznava), *labod*, *žirafa*, *konj*, *krava*, *obraz*, *kolo*, *motor*, *kladivo*, *škarje*, *pešec* in *konj*.