DISS. ETH NO. 23670

# NON-PARAMETRIC MODELS FOR STRUCTURED DATA AND APPLICATIONS TO HUMAN BODIES AND NATURAL SCENES

A thesis submitted to attain the degree of
DOCTOR OF SCIENCES of ETH ZURICH
(Dr. sc. ETH Zurich)

presented by
ANDREAS M. LEHRMANN

Dipl.-Inform., University of Tuebingen

born on 16.02.1984
citizen of Germany

accepted on the recommendation of

Prof. Dr. Luc van Gool
Dr. Peter V. Gehler
Prof. Dr. Jürgen Gall

2016

*To my father,*
*in loving memory*

# Zusammenfassung

Die vorliegende Arbeit untersucht nicht-parametrische Modelle für strukturierte Daten und deren Anwendungsgebiete in der Computervision. Ziel ist die Entwicklung von kontext-sensitiven Architekturen, die zugleich expressiv und effizient sind. Unser Fokus liegt hierbei auf gerichteten graphischen Modellen, speziell Bayes'schen Netzen, in denen wir die Flexibilität nicht-parametrischer Verteilungen mit der Effizienz von Topologien mit beschränkter Baumweite kombinieren. Beschränkte Baumweiten werden entweder durch Limitierung des maximalen Eingangsgrads der unterliegenden Graphstruktur oder durch Einführung von Determinismus erzielt. Die nicht-parametrischen Verteilungen in den Knoten des Graphen sind durch Entscheidungsbäume oder Kerndichteschätzer gegeben.

Der durch spezifische Netztopologien implizierte Informationsfluss zwischen den Variablen, insbesondere die resultierenden Unabhängigkeiten, erlaubt eine natürliche Integration und Kontrolle von Kontextinformationen. Wir unterscheiden zwischen drei verschiedenen Arten von Kontext: Statisch, dynamisch und semantisch. In vier Ansätzen schlagen wir Modelle vor, die verschiedene Kombinationen dieser Kontextformen aufweisen und je nach Ausprägung die Modellierung von strukturierten Daten in Raum, Zeit und davon abgeleiteten Hierarchien erlauben. Der generative Charakter der vorgestellten Modelle ermöglicht die direkte Synthese plausibler Hypothesen.

Umfassende Experimente validieren die entwickelten Modelle in zwei Anwendungsszenarien, die in der Computervision von besonderer Bedeutung sind: Menschliche Körper und natürliche Szenen. In den praktischen Teilen dieser Arbeit behandeln wir diese beiden Großbereiche unter verschiedenen Gesichtspunkten und zeigen Anwendungen bei der Modellierung menschlicher Posen, Bewegungen und Segmentierungen sowie bei Objektkategorisierung und -lokalisierung. Wir profitieren dabei von der Verfügbarkeit moderner Datensätze mit bislang unerreichter Größe und Diversität. Der Vergleich mit klassischen Ansätzen und aktuellen Entwicklungen auf Basis etablierter Bewertungskriterien erlaubt die objektive Einordnung unserer Beiträge.

# Abstract

The purpose of this thesis is the study of non-parametric models for structured data and their fields of application in computer vision. We aim at the development of context-sensitive architectures which are both expressive and efficient. Our focus is on directed graphical models, in particular Bayesian networks, where we combine the flexibility of non-parametric local distributions with the efficiency of a global topology with bounded treewidth. A bound on the treewidth is obtained by either constraining the maximum indegree of the underlying graph structure or by introducing determinism. The non-parametric distributions in the nodes of the graph are given by decision trees or kernel density estimators.

The information flow implied by specific network topologies, especially the resultant (conditional) independencies, allows for a natural integration and control of contextual information. We distinguish between three different types of context: static, dynamic, and semantic. In four different approaches we propose models which exhibit varying combinations of these contextual properties and allow modeling of structured data in space, time, and hierarchies derived thereof. The generative character of the presented models enables a direct synthesis of plausible hypotheses.

Extensive experiments validate the developed models in two application scenarios which are of particular interest in computer vision: human bodies and natural scenes. In the practical sections of this work we discuss both areas from different angles and show applications of our models to human pose, motion, and segmentation as well as object categorization and localization. Here, we benefit from the availability of modern datasets of unprecedented size and diversity. Comparisons to traditional approaches and state-of-the-art research on the basis of well-established evaluation criteria allows the objective assessment of our contributions.

# Acknowledgements

First and foremost, I want to thank my parents Brigitte Picot-Lehrmann and Harald Lehrmann. For their trust, support, and unconditional love.

I am also deeply grateful to my PhD supervisors Dr. Peter Gehler (MPI for Intelligent Systems) and Dr. Sebastian Nowozin (Microsoft Research). They introduced me to the field of computer vision, pushed me to the edge, and let me explore uncharted territory. Talking to them has been a constant source of inspiration and this work wouldn't have been possible without their guidance and patience. I want to thank Prof. Dr. Luc van Gool (ETH Zurich) for his immediate consent to act as my ETH supervisor as well as for his understanding and encouraging words when time was of the essence. I am thankful to Prof. Dr. Jürgen Gall for agreeing to be part of my thesis committee. My sincere thanks go to Dr. Leonid Sigal, Dr. Brian McWilliams and the rest of the team at Disney Research for an incredible time in Pittsburgh and Zurich; it has been a dream come true.

Writing a PhD thesis is a daunting task. In times of doubt, I've been in the lucky position to have my friends Karin Schaller, Thomas Nestmeyer, Martin Kiefel, Varun Jampani, Christoph Lassner, Fatma Güney, Naejin Kong, Akash Sethi, Yanwei Fu, and Albert Li around. I want to express my utmost gratitude for candid discussions and their genuine advice in decisive moments. Special thanks go to Nina Sophie Klett, Thomas Nestmeyer, Fatma Güney, and Christoph Lassner for their helpful feedback and comments on this manuscript.

Thomas Nestmeyer, thank you for dragging me up the mountains, through the ocean, and beyond any physical limit. Anything is possible.

Karin Schaller, thank you for virtual flowers, real cookies, and a unique friendship. You are a constant in my life that I can always rely on.

Nina Sophie Klett, thank you for being my compass and anchor in a confusing world. You turn the deafening bass of life into a faint whisper.

— Andreas M. Lehrmann

# Table of Contents

**List of Figures** **xv**

**List of Tables** **xvii**

**1 Introduction** **1**
1.1 Approach . . . . . . . . . . 2
1.2 Overview . . . . . . . . . . 3

**2 Non-parametric Learning and Inference** **7**
2.1 Basic Framework . . . . . . . . . . 8
2.2 Probabilistic Graphical Models . . . . . . . . . . 9
2.3 Non-parametric Local Models . . . . . . . . . . 11
2.3.1 Kernel Density Estimation . . . . . . . . . . 13
2.3.2 Decision Trees . . . . . . . . . . 15
2.3.3 Ensemble Models . . . . . . . . . . 17
2.4 Semantics . . . . . . . . . . 18
2.5 Context . . . . . . . . . . 20
2.6 Complexity . . . . . . . . . . 21
2.6.1 Learning . . . . . . . . . . 22
2.6.2 Inference . . . . . . . . . . 23
2.6.3 Consensus Models . . . . . . . . . . 24

**3 Non-parametric Priors of Human Pose** **27**
3.1 Introduction . . . . . . . . . . 28
3.1.1 Related Work . . . . . . . . . . 29
3.2 Non-parametric Networks of Human Pose . . . . . . . . . . 30
3.2.1 Learning the Graph Structure . . . . . . . . . . 31
3.2.2 Learning the Local Models . . . . . . . . . . 33
3.2.3 Exact Log-likelihoods and Sampling . . . . . . . . . . 34
3.3 Experiments . . . . . . . . . . 35

3.3.1 Pose Model . . . . . . . . . . . . . . . . . . . . . . . . . . . 35
3.3.2 Comparison of Hold-out Log-likelihoods . . . . . . . . . . . . 36
3.3.3 Compositionality . . . . . . . . . . . . . . . . . . . . . . . . 38
3.4 Real-time Scoring . . . . . . . . . . . . . . . . . . . . . . . . . . . . 39

**4 Non-parametric Markov Models for Human Motion 43**
4.1 Introduction . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 44
4.1.1 Dynamic Bayesian Networks . . . . . . . . . . . . . . . . . . 44
4.1.2 Contributions . . . . . . . . . . . . . . . . . . . . . . . . . . 45
4.2 Nonlinear Markov Models . . . . . . . . . . . . . . . . . . . . . . . 45
4.2.1 Autoregressive Trees . . . . . . . . . . . . . . . . . . . . . . 46
4.2.2 Dynamic Forests . . . . . . . . . . . . . . . . . . . . . . . . 47
4.2.3 Training . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 48
4.3 Latent Space View . . . . . . . . . . . . . . . . . . . . . . . . . . . 49
4.4 Experiments . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 51
4.4.1 Human Action Recognition . . . . . . . . . . . . . . . . . . 52
4.4.2 Motion Completion . . . . . . . . . . . . . . . . . . . . . . . 54
4.4.3 Predicting 3D from 2D . . . . . . . . . . . . . . . . . . . . . 57
4.5 Computational Aspects . . . . . . . . . . . . . . . . . . . . . . . . 58

**5 Non-parametric Scene Prior Hierarchies 59**
5.1 Introduction . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 60
5.1.1 Approach . . . . . . . . . . . . . . . . . . . . . . . . . . . . 61
5.1.2 Related Work . . . . . . . . . . . . . . . . . . . . . . . . . . 62
5.2 Non-parametric Scene Priors . . . . . . . . . . . . . . . . . . . . . . 63
5.2.1 Occurrence Model . . . . . . . . . . . . . . . . . . . . . . . . 65
5.2.2 Location Model . . . . . . . . . . . . . . . . . . . . . . . . . 67
5.3 Inference . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 73
5.4 Experiments . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 74
5.4.1 Expected Log-likelihood . . . . . . . . . . . . . . . . . . . . 75
5.4.2 Occurrence and Location Accuracy . . . . . . . . . . . . . . 76
5.5 Qualitative Results . . . . . . . . . . . . . . . . . . . . . . . . . . . 80
5.5.1 Scene Layout Synthesis . . . . . . . . . . . . . . . . . . . . . 80
5.5.2 Occurrence Network Predictions . . . . . . . . . . . . . . . . . 80
5.5.3 Location Network Predictions . . . . . . . . . . . . . . . . . 80

**6 Non-parametric Semantic Video Segmentation 85**
6.1 Introduction . . . . . 86
6.1.1 Contribution . . . . . 87
6.1.2 Related Work . . . . . 87
6.2 Autocontext . . . . . 89
6.3 Dynamic Autocontext . . . . . 90
6.3.1 Instantiation . . . . . 91
6.3.2 Hierarchical Training . . . . . 94
6.4 Experiments . . . . . 94
6.4.1 Datasets . . . . . 94
6.4.2 Quantitative Analysis . . . . . 97
6.4.3 Qualitative Analysis . . . . . 100

**7 Conclusions and Directions 101**
7.1 Conclusions . . . . . 102
7.2 Directions . . . . . 104

**Appendices 111**

**A Acronyms 113**

**B Curriculum Vitae 115**

**Bibliography 119**

**Index 131**

# List of Figures

1.1 Thesis Overview. . . . . . . . . . . . . . . . . . . . . . . . . . . . 3
1.2 Non-parametric Priors of Human Pose. . . . . . . . . . . . . . . . . . 4
1.3 Non-parametric Markov Models for Human Motion. . . . . . . . . . . . 5
1.4 Non-parametric Scene Prior Hierarchies. . . . . . . . . . . . . . . . . 5
1.5 Non-parametric Semantic Video Segmentation. . . . . . . . . . . . . . 6

2.1 Probabilistic Graphical Models. . . . . . . . . . . . . . . . . . . . . . 10
2.2 Parametric vs. Non-parametric Local Models. . . . . . . . . . . . . . 12
2.3 Overview: Kernel Density Estimation. . . . . . . . . . . . . . . . . . 14
2.4 Overview: Decision Trees. . . . . . . . . . . . . . . . . . . . . . . . . 15
2.5 Decision Tree Classes. . . . . . . . . . . . . . . . . . . . . . . . . . . 16
2.6 Parameterizations of Human Poses and Natural Scenes. . . . . . . . . 19
2.7 Types of Context. . . . . . . . . . . . . . . . . . . . . . . . . . . . . 20
2.8 Learning Bayesian Networks. . . . . . . . . . . . . . . . . . . . . . . 23
2.9 Topological and Distributional Error. . . . . . . . . . . . . . . . . . . 24

3.1 Key Message: Static Context. . . . . . . . . . . . . . . . . . . . . . . 27
3.2 Optimal Graph Structure. . . . . . . . . . . . . . . . . . . . . . . . . 32
3.3 Samples from Human Pose Prior. . . . . . . . . . . . . . . . . . . . . 35
3.4 Compositionality. . . . . . . . . . . . . . . . . . . . . . . . . . . . . 38
3.5 Approximate Log-likelihoods. . . . . . . . . . . . . . . . . . . . . . . 40
3.6 Approximation Trade-off and Real-time Inference. . . . . . . . . . . . 41

4.1 Key Message: Dynamic Context. . . . . . . . . . . . . . . . . . . . . 43
4.2 Autoregressive Tree. . . . . . . . . . . . . . . . . . . . . . . . . . . . 46
4.3 Dynamic Forest Model. . . . . . . . . . . . . . . . . . . . . . . . . . 47
4.4 Visualization of a 2D→3D Task. . . . . . . . . . . . . . . . . . . . . 56

5.1 Key Message: Extrinsic Semantic Context. . . . . . . . . . . . . . . . 59
5.2 Scene Prior Hierarchy. . . . . . . . . . . . . . . . . . . . . . . . . . . 61
5.3 Location Network Architecture. . . . . . . . . . . . . . . . . . . . . . 68

5.4 Object Category Topologies. . . . . . . . . . . . . . . . . . . . . . . . . 70
5.5 Dynamic Location Topology. . . . . . . . . . . . . . . . . . . . . . . . . 71
5.6 Distribution of Test Log-likelihoods. . . . . . . . . . . . . . . . . . . . . 75
5.7 Quantitative Evaluation on COCO. . . . . . . . . . . . . . . . . . . . . 77
5.8 Scene Layout Synthesis. . . . . . . . . . . . . . . . . . . . . . . . . . . 81
5.9 Occurrence Network Predictions. . . . . . . . . . . . . . . . . . . . . . 82
5.10 Location Network Predictions. . . . . . . . . . . . . . . . . . . . . . . . 83

6.1 Key Message: Intrinsic Semantic Context. . . . . . . . . . . . . . . . . 85
6.2 Dynamic Autocontext. . . . . . . . . . . . . . . . . . . . . . . . . . . . 91
6.3 Feature Hierarchy. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 92
6.4 Comparison of Static and Dynamic Autocontext (J-HMDB). . . . . . 95
6.5 Comparison of Static and Dynamic Autocontext (FBMS). . . . . . . 96
6.6 Positive Examples on the Buffy Dataset. . . . . . . . . . . . . . . . . . 98
6.7 Failure Cases on the Buffy Dataset. . . . . . . . . . . . . . . . . . . . . 98

7.1 Online Annotation Tool. . . . . . . . . . . . . . . . . . . . . . . . . . . 108

# List of Tables

3.1 Expected Log-likelihoods. . . . . . . . . . . . . . . . . . . . . . . . . . 37

4.1 Action Classification. . . . . . . . . . . . . . . . . . . . . . . . . . . . . 53
4.2 Motion Completion. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 55
4.3 3D from 2D. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 58

5.1 Comparison to Baseline Priors. . . . . . . . . . . . . . . . . . . . . . . . 79

6.1 Quantitative Analysis of Dynamic Autocontext. . . . . . . . . . . . . . . 99

# Chapter 1

# Introduction

In recent years, many domains in computer vision have reached a stage of practical applicability, a development that can be attributed to the availability of powerful image features based on convolutional neural networks (CNNs) [87] and large-scale datasets with hundreds of thousands of training data points (Human 3.6M [71], Microsoft COCO [99]). While these advancements continue to push the envelope on performance of local classifiers in multi-class detection tasks (*e.g.*, pose estimation, scene understanding), it has been a long-held belief in the computer vision and machine learning community that contextual information in the form of label consistency, structural dependencies or, more generally, priors remains important. First, because it is unlikely that independent classifiers can achieve perfect performance with local evidence only; and second, because generative models can produce image- and context-conditioned hypotheses and likelihoods, which enables reasoning in scenarios with partial evidence or incomplete data.

In addition to context awareness, most modern applications require models that also feature other desirable properties, such as high flexibility, fast runtime, and good generalization. It is clear that these cannot all be achieved at the same time and, while universal statements are difficult, different paradigms exhibit different tendencies: Unstructured approaches often lead to high-dimensional learning and inference problems and, for that reason, tend to suffer from overfitting and computational intractabilities, especially in case of flexible non-parametric approaches with high variance. Parametric models, on the other hand, are typically more robust and efficient but may not be able to represent the true generating distribution, leading to high bias. Structured approaches like graphical models make conditional independence assumptions to alleviate some of these issues. In particular, they can simplify learning and allow for tractable inference, but the best-researched cases focus on either discret(iz)e(d) distributions or unimodal Gaussian models, which often results in false modeling assumptions when considering tasks with continuous domain.

## 1.1 Approach

The aforementioned progress in terms of features and datasets allows for new types of data-driven models with a more balanced set of properties. In this thesis, we investigate the use of structured non-parametric models to achieve such a compromise between context awareness, flexibility, and efficiency.

**Context Awareness.** A context-aware model is able to incorporate surrounding evidence to reduce local uncertainty. We take a probabilistic perspective and model context as a conditional distribution or, in the more general setting of dependencies between multiple variables, as a directed graphical model. We distinguish between two fundamental model classes: Static models, which operate on single images, and dynamic models, which additionally condition their actions on other images, for instance related frames in a video or motion capture sequence. In both cases, consideration of contextual information has proven to be beneficial in terms of predictive performance: While static context can exploit spatial dependencies, dynamic models are given access to differential or even global information that can be incorporated into discriminative features as well. Beyond this well-known form of spatio-temporal context on the feature level, there is an orthogonal type of semantic context on the model level, which in turn can be either static or dynamic. One instance of such semantic interactions between models is given by a hierarchical, *i.e.*, chain-like, sequence of models that relate to each other through an either intrinsic ('auto-') or extrinsic ('allo-') type of model context. Figure 1.1 summarizes these relationships and provides an overview of the concepts developed in this thesis.

**Flexibility.** Independent of the specific form of context, the flexibility of a directed graphical model, such as a (dynamic) Bayesian network, depends on two factors: Its global topology and its local distributions. We argue that tractable topologies with bounded treewidth but optimized structure can be the foundation of powerful models when combined with the flexibility of non-parametric local distributions. While enforcing conditional independencies through topological constraints introduces a global estimation bias, reliable non-parametric estimation of the low-dimensional local models becomes feasible and modeling of spurious interactions is avoided. Together with a large corpus of training data, this implicit form of regularization can effectively counter the susceptibility of global non-parametric models to overfitting.

**Efficiency.** Since treewidth is directly related to the complexity of common inference tasks, *e.g.*, the computation of (max-)marginals or filtering/smoothing dis-

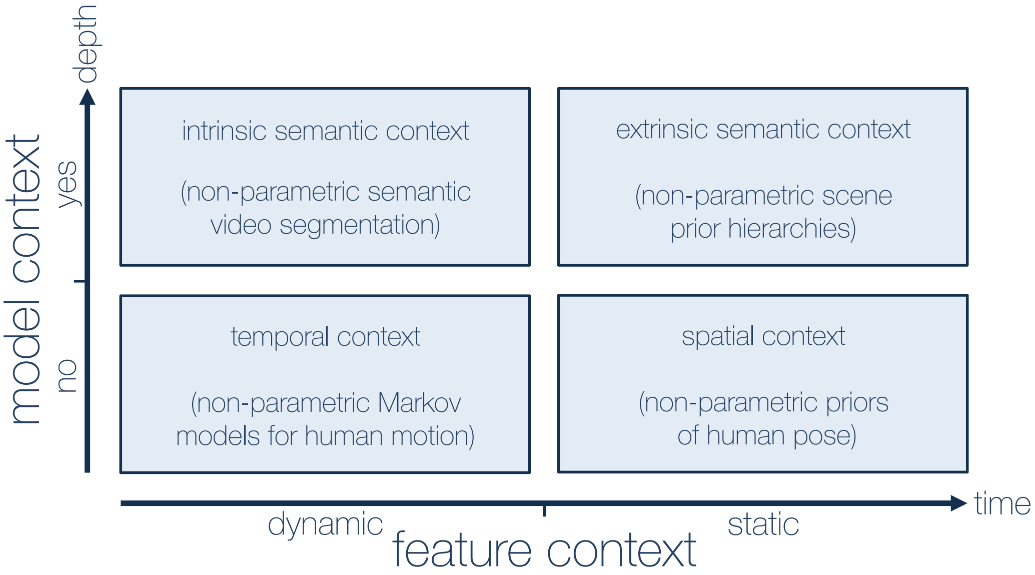

model context
depth
yes
no
intrinsic semantic context
(non-parametric semantic video segmentation)
extrinsic semantic context
(non-parametric scene prior hierarchies)
temporal context
(non-parametric Markov models for human motion)
spatial context
(non-parametric priors of human pose)
time
dynamic
static
feature context

**Figure 1.1: Thesis Overview.** Contextual information can be static, dynamic, or semantic. We develop four structured non-parametric models that exploit different combinations of these types of context and show applications to human body and natural scene modeling.

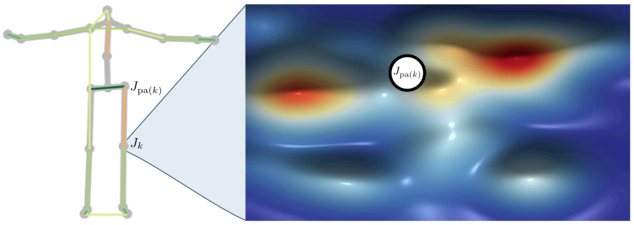
tributions, we obtain efficient inference procedures. Further speed-ups are possible by combining message-based inference schemes like belief propagation with efficient non-parametric local models like decision trees or fast approximations for typically expensive operations like the evaluation of kernel density estimates.

## 1.2 Overview

We make technical contributions to all four principled model classes shown in Figure 1.1. While the presented tools are agnostic with regard to the particular application, our guiding examples will be

- human bodies, where the co-location of joints or foreground pixels defines a pose or segmentation in Euclidean space and the movement of limbs describes a trajectory in time.
- natural scenes, where the co-occurrence of object categories defines a setting and the co-location of object instances describes a layout in Euclidean space.

Both application scenarios are of great relevance in computer vision and unique targets due to the specific challenges involved, including the high articulation, multimodal dependencies, and non-linear motions of human bodies and the diverse appearance, flexible layout and dynamic object dependencies of natural scenes. We describe our approaches in some more detail:



Optimal tree topology Non-parametric local distributions

**Figure 1.2: Non-parametric Priors of Human Pose.**

**Chapter 3 – Spatial Context.** Having a sensible prior of human pose is a vital ingredient in many computer vision applications, including 2D pose estimation from images and 3D pose tracking from videos. In this chapter, we introduce a structured prior of static pose whose generative principles allow for an easy integration of spatial context. A high-dimensional density estimation problem at its heart, we factor the parameterization of a human pose into a Bayesian network with low treewidth to achieve compositionality, avoid overfitting, and ensure the properties established in section 1.1. Our approach makes two novel contributions concerning the global topology and local distributions of continuous Bayesian networks: First, we learn an optimal arborescence by conditioning the local distributions on the most informative joints given the topological constraint (Figure 1.2, left), leveraging non-parametric mutual information estimators on continuous joint data. Second, we augment the expressiveness of Bayesian networks by proposing Bayesian networks with non-parametric local distributions in the form of conditional kernel density estimates (Figure 1.2, right). Our framework can serve as a regularization term or hypothesis generator for any pose-related system and overcomes typical disadvantages of both unstructured non-parametric methods, which lack compositionality, and parametric networks, which are limited in their expressiveness. Despite its non-parametric nature, we achieve real-time performance for up to $10^5$ training data points.

**Chapter 4 – Temporal Context.** A natural extension of the static frame-by-frame setting is to condition a model on previous observations. Traditional approaches for this type of task include latent variable models following the state-space equations, *i.e.*, time-homogeneous first-order Markov chains on a hidden state

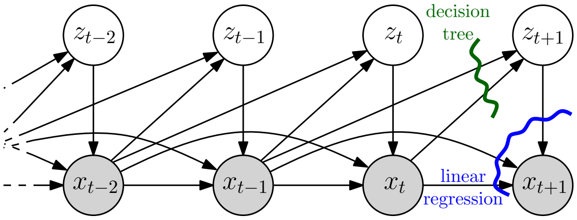


**Figure 1.3: Non-parametric Markov Models for Human Motion.**

sequence that generate the observable variables via some conditional emission distribution. While the use of hidden variables makes them flexible models, inference tasks like the computation of marginal observation likelihoods become in general intractable, because the marginalized model is non-Markovian. Tractable but computationally expensive special cases include Hidden Markov Models (HMMs), where the state space is discrete, and the Kálmán filter, where both hidden and observable distributions are linear functions with additive Gaussian noise. Instead, we investigate a non-parametric model without a latent space that can be understood as an efficient $\delta$-approximation to the latent distribution of an HMM. In particular, we propose dynamic forests, a temporal extension of random forests in which we condition the model at time $t$ on features of multiple previous frames $t-1, \ldots, t-K$. The conditional models stored at the leaf nodes of each tree correspond to latent states in an HMM (Figure 1.3). As a result of this deterministic selection of a 'latent state', dynamic forests decouple in time, leading to very efficient inference. Applications of dynamic forests to high-dimensional action recognition and motion infilling tasks on human motion capture data confirm their remarkable performance.

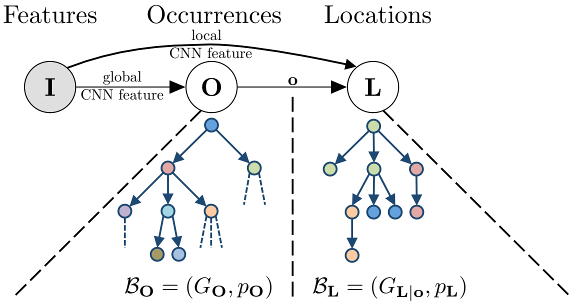

Features
Occurrences
Locations
local CNN feature
global CNN feature
**I**
**O**
**o**
**L**
$\mathcal{B}_{\mathbf{O}} = (G_{\mathbf{O}}, p_{\mathbf{O}})$
$\mathcal{B}_{\mathbf{L}} = (G_{\mathbf{L}|\mathbf{o}}, p_{\mathbf{L}})$

**Figure 1.4: Non-parametric Scene Prior Hierarchies.**

**Chapter 5 – Extrinsic Semantic Context.** In this chapter, we move from human poses to natural scenes. We observe that common tasks in this area, such as the recognition of objects as well as their spatial extent and global layout in an image, relate to each other through a coarse-to-fine hierarchy following information enclosure, *i.e.*, fine-grained tasks comprise, either explicitly or implicitly, coarser tasks. We mimic these natural dependencies and propose a novel first-order hierarchy of conditional Bayesian networks to support inference tasks with such a rich structure. Information enclosure ensures that each hierarchical layer can be conditioned on its predecessor and thus needs to encode differential information only. Specifically, we propose a bilayered architecture with task-dependent parametric and non-parametric structure, where the first layer models the occurrence of objects in a scene and the second layer the spatial layout of those objects conditioned on the first layer (Figure 1.4). One key advantage of our model is its ability to utilize a dynamic graph topology and flexible non-parametric local models in the spatial layout layer. Our architecture has a number of appealing properties, including the ability to synthesize plausible sets and layouts of co-occurring objects and the ability to efficiently reason about a scene at different levels of granularity. We also show how to integrate the proposed prior with rich CNN-based likelihoods to achieve improved performance in object categorization, quantification, and detection tasks. Large-scale experiments on the recently introduced Microsoft COCO dataset illustrate the benefits of our proposed model.

**Chapter 6 – Intrinsic Semantic Context.** In our last project, we consider an intrinsic form of semantic context based on autocontext features. Autocontext is a multi-stage approach that iteratively improves the semantic labeling of an arbitrary base classifier by conditioning on the output of previous stages, thereby abstracting away irrelevant variation of the original input image. In contrast to image context, where temporal dependencies are well-studied, causal versions of semantic context have not been looked into so far. We are therefore proposing a non-parametric dynamic autocontext model for semantic labeling tasks in videos. In particular, we investigate the benefits of such a model for the challenging task of human video segmentation. Our dynamic autocontext model represents temporal dependencies by conditioning on the predictions made by a static autocontext model, both for the current frame as well as the previous frame (Figure 1.5). At this stage, appearance variation that is not relevant to the task has already been largely removed. Hence, the dynamic autocontext can now model temporal dependencies among the semantic labels; for example, it can learn an implicit motion model and shape prior between frames, independent of the image appearance.

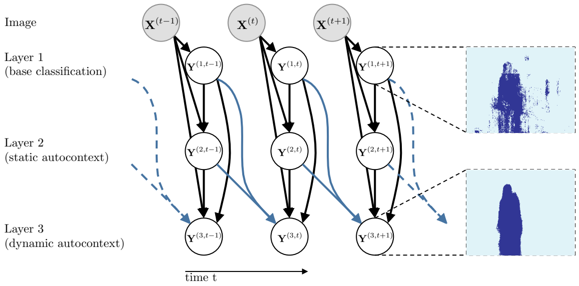


**Figure 1.5: Non-parametric Semantic Video Segmentation.**

# Chapter 2

# Non-parametric Learning and Inference with Structured Data

In this chapter, we introduce the notation and structure of a non-parametric learning and inference framework that constitutes the foundation of all methods we discuss in later chapters.

After the definition of some fundamental statistical concepts (section 2.1) and a brief review of graphical models (section 2.2), we turn to non-parametric graphical models and describe two particularly useful approaches that will be essential elements in chapters 3–6. We also draw connections to ensemble methods (section 2.3) and highlight some of their (meta-level) properties. We proceed with a description of the semantics underlying our models (section 2.4), especially parameterizations of human poses and natural scenes, and show how we can use our framework to incorporate different types of contextual information frequently encountered in various computer vision tasks (section 2.5). We conclude with a review of the complexity of learning and inference, which motivates the topological and distributional structure of our models (section 2.6).

The presentation in this chapter focuses on basic concepts; extensions and applications will be discussed directly in the respective chapters.

## 2.1 Basic Framework

In this work, we discuss both models with discrete and continuous random variables $X$. To be able to treat them in a common framework, we assume that the $X$-associated push-forward measure $X_{*}p$ is absolutely continuous with respect to a base measure $\mu$ and refer to the Radon-Nikodym derivative $f_X = \frac{\mathrm{d}X_{*}p}{\mathrm{d}\mu}$ as *density*. Special cases include probability mass functions ($X$ discrete, $\mu$ counting measure) and traditional probability density functions ($X$ continuous, $\mu$ Lebesgue measure).

Let $\mathbf{X}^{(1)}, \ldots, \mathbf{X}^{(N)} \sim F_{\mathbf{X}}$ be an *i.i.d.* sample of $d$-dimensional random vectors $\mathbf{X}^{(i)} = (X_1^{(i)}, \ldots, X_d^{(i)})^\top$ and

$$\mathcal{D}_{\mathbf{x}} = \left(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(N)}\right) = \left(x_j^{(i)}\right)_{\substack{j=1,\ldots,d\\ i=1,\ldots,N}} \in \mathbb{R}^{d\times N} \tag{2.1}$$

the corresponding random variates.[1] We refer to $\mathcal{D}_{\mathbf{x}}$ as a *training set* and to $\mathbf{x}^{(i)}$ as a *training point*. We can think of a training point either directly as an element of a *feature space* $\Phi$ or, alternatively, as an element of an *input space* $\mathcal{X}$ that is related to $\Phi$ through a feature mapping $\phi : \mathcal{X} \to \Phi$.

In a supervised setting, we are additionally given a set of corresponding response variables $\mathbf{Y}^{(i)} \sim F_{\mathbf{Y}|\mathbf{X}^{(i)}}$ with realizations $\mathcal{D}_{\mathbf{y}}$, which lie either in $\mathbb{Z}_K^N$ (classification with $K$ target classes) or $\mathbb{R}^{d'\times N}$ (regression with $d'$ output dimensions). We will sometimes refer to the codomain of $\mathbf{Y}$ as the *output space* and to the tuple $(\mathcal{D}_{\mathbf{x}}, \mathcal{D}_{\mathbf{y}})$ as a *supervised training set*.

**Learning.** $F_{\mathbf{X}}$ (or $F_{\mathbf{Y}|\mathbf{X}}$) is the true (conditional) cumulative distribution function and typically unknown. Our goal is thus to learn $F_{\mathbf{X}}' = f_{\mathbf{X}}$ from $\mathcal{D}_{\mathbf{x}}$ (density estimation) or $f_{\mathbf{Y}|\mathbf{X}}$ from $(\mathcal{D}_{\mathbf{x}}, \mathcal{D}_{\mathbf{y}})$ (classification, regression). In a supervised task, to make an actual prediction, we will usually consider the conditional expectation,

$$g(\mathbf{x}) = \mathbb{E}[\mathbf{Y} \mid \mathbf{X} = \mathbf{x}] = \int_{\mathrm{codom}(\mathbf{Y})} \frac{\mathbf{y} f_{\mathbf{X},\mathbf{Y}}(\mathbf{x}, \mathbf{y})}{f_{\mathbf{X}}(\mathbf{x})} \,\mathrm{d}\mathbf{y}. \tag{2.2}$$

In this fractional form, it is obvious that we can always view supervised tasks as an instance of density estimation in which we condition the joint density on the input/feature space. Estimates will sometimes be denoted by a hat, $\widehat{\bullet}$, but we will also frequently omit this designation when clear from the context.

---

[1] We will sometimes interpret $\mathbf{X}$ as an unordered set and use $|\mathbf{X}|$ to refer to the length $d$ of $\mathbf{X}$.

**Inference.** Given a learned probabilistic model $\widehat{f}_{\mathbf{X}}$ (or $\widehat{f}_{\mathbf{Y}|\mathbf{X}}$), we are interested in solving a number of inference tasks which allow us to assess an existing observation, generate a new hypothesis, or determine the most likely configuration:

1. Compute the log-likelihood $\log \widehat{f}_{\mathbf{X}}(\mathbf{x})$ of an observation $\mathbf{x}$.
2. Draw a sample $\mathbf{x}$ from $\mathbf{X} \sim \widehat{f}_{\mathbf{X}}$.
3. Estimate the maximum a-posteriori configuration $\mathbf{x}^{\mathrm{MAP}} = \arg\max_{\mathbf{x}} \widehat{f}_{\mathbf{X}}(\mathbf{x})$.[2]

Analogous concepts exist for the supervised cases. Sometimes, we are also interested in solving these tasks given evidence, marginals, or both. For instance, given a subset $\mathbf{W} \subset \mathbf{X}$, we might be interested in either the marginal log-likelihood

$$\log \widehat{f}_{\mathbf{W}}(\mathbf{w}) = \log \int_{\mathbf{x}\setminus\mathbf{w}} \widehat{f}_{\mathbf{X}}(\mathbf{x}) \; \mathrm{d}(\mathbf{x}\setminus\mathbf{w}), \tag{2.3}$$

the conditional log-likelihood

$$\log \widehat{f}_{\mathbf{X}\setminus\mathbf{W}}(\mathbf{x}\setminus\mathbf{w} \mid \mathbf{W} = \mathbf{w}) = \log \widehat{f}_{\mathbf{X}}(\mathbf{x}) - \log \widehat{f}_{\mathbf{W}}(\mathbf{w}), \tag{2.4}$$

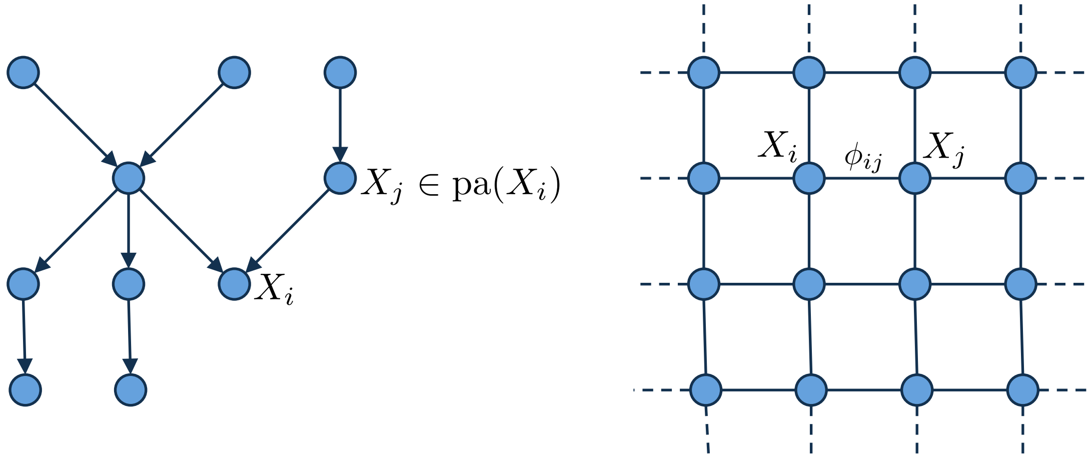
or a combination of both. Similar tasks exist for sampling and MAP estimation.

## 2.2 Probabilistic Graphical Models

Most of the inference tasks introduced in the previous section are computationally intractable for all but the simplest probabilistic models, either due to the sheer number of discrete states or the lack of analytical solutions for the involved continuous integrals and optimizations. Probabilistic *graphical* models can alleviate some of these issues by introducing additional modeling assumptions that allow for easier learning and inference. Since their popularization in the 1970's and 1980's, they have been the statistical foundation in countless areas of computer science, including major impacts in areas like bioinformatics, finance, and computer vision. The literature on graphical models is rich and we refer to [81] as an excellent source with an emphasis on discrete and parametric graphical models. Here, we concentrate on the absolute essentials and pave the way for the following section, which introduces continuous and non-parametric graphical models.

---

[2] We use the term 'maximum a-posteriori' interchangeably with 'mode of a density', even though its precise meaning refers to the mode of a posterior density.



**(a)** Bayesian Network. **(b)** Markov Network.

**Figure 2.1: Probabilistic Graphical Models.** Bayesian networks and Markov networks are the two most common classes of graphical models. **(a)** Polytrees are a subclass of Bayesian networks that allow for efficient inference. **(b)** Grid networks are a subclass of Markov networks that mimic the topological structure of an image.

A graphical model over a random vector $\mathbf{X} = (X_j)_{j=1}^n$ consists of two fundamental components, a graph structure $G = (\mathbf{X}, E)$ and a joint distribution $f_{\mathbf{X}}(\mathbf{x})$. The graph structure induces a factorization of the joint distribution and can be either directed or undirected, which leads to different factorizations and, eventually, to the two primary classes of graphical models: Bayesian networks and Markov networks.

**Bayesian networks.** Bayesian networks are rooted in the area of influence diagrams [67] and were first described in the 1980's by Pearl [118, 117] as an economical description of a joint distribution that is based on the representation of (in)dependencies in the human brain. A simple and intuitive motivation for their use is to consider the chain rule of probability, which allows us to write *any* joint density $f_{\mathbf{X}}(\mathbf{x})$ as a product of conditional densities,

$$f_{\mathbf{X}}(\mathbf{x}) = \prod_{i=1}^{n} f_{X_i}(x_i \mid \mathbf{x}_{1:i-1}). \tag{2.5}$$

A Bayesian network introduces conditional independence assumptions into this factorization by restricting the set of conditioning variables with the help of a graph structure. Formally, a Bayesian network is a tuple $\mathcal{B} = (G, f_{\mathbf{X}})$ consisting of a joint density $f_{\mathbf{X}}$ and a directed, acyclic graph $G = (V, E)$ with vertex set $V := \mathbf{X}$ and edge set $E \subseteq \mathbf{X} \times \mathbf{X}$. We say that $f_{\mathbf{X}}$ *factorizes* over $G$, if

$$f_{\mathbf{X}}(\mathbf{x}) = \prod_{i=1}^{n} f_{X_i}(x_i \mid \mathrm{pa}_G(x_i)), \tag{2.6}$$

where $\mathrm{pa}_G(\bullet)$ returns the parents of a node w.r.t. $G$. For $E = \varnothing$, $f_{\mathbf{X}}$ is an independent product of marginal densities, for $E = \{(X_i, X_j) \mid 1 \leqslant i < j \leqslant n\}$, Eq. (2.6) reduces to Eq. (2.5). We refer to the individual factors of such a factorization as *local distributions* or *local models*. It is easy to verify that

$$f_{\mathbf{X}} \text{ factorizes over } G \iff \{(\mathbf{X}' \perp \mathbf{X}'' \mid \mathbf{E}) \mid d\text{-sep}_G((\mathbf{X}', \mathbf{X}''), \mathbf{E})\} \subseteq \mathcal{I}(f_{\mathbf{X}}). \tag{2.7}$$

Here, $d$-sep$_G$ refers to $d$-separation [53] of $\mathbf{X}' \subseteq \mathbf{X}$ and $\mathbf{X}'' \subseteq \mathbf{X}$ given $\mathbf{E} \subseteq \mathbf{X}$ in $G$ and $\mathcal{I}(f_{\mathbf{X}})$ is the set of all conditional independencies that hold in $f_{\mathbf{X}}$. See [81] for a proof.

**Markov networks.** The history of Markov networks can be traced back to the works of Lenz [97] and Ising [72]. Generalizations of those early approaches by Spitzer [149] and Preston [120] as well as a comprehensive and more accessible treatment by Kindermann and Snell [79] have lead to the modern notion of a Markov network and numerous applications in many areas of science. Briefly, a Markov network is a graphical model with an undirected graph structure $G$, such that each factor of a Gibbs distribution $f_{\mathbf{X}}(\mathbf{x})$ with partition function $Z$ is a (maximal) clique in $G$,

$$f_{\mathbf{X}}(\mathbf{x}) = Z^{-1} \prod_{c \in \mathrm{cl}(G)} \phi_c(\mathbf{x}_c), \tag{2.8}$$

where $\mathrm{cl}(\bullet)$ returns the cliques of a graph and $\phi_\bullet$ is a non-negative factor.

Markov networks will not play a prominent role in this work and are included for the sake of completeness only. Instead, we focus on directed graphical models as introduced in Eq. (2.6), mainly because their generative principles allow for a probabilistic treatment of conditional queries and an easy synthesis of plausible hypotheses.

## 2.3 Non-parametric Local Models

Estimating a Bayesian network $\mathcal{B} = (\widehat{G}, \widehat{f}_{\mathbf{X}})$ from a training set $\mathcal{D}_{\mathbf{x}}$ requires the specification of a directed acyclic graph $\widehat{G}$ and a set of local probabilistic models

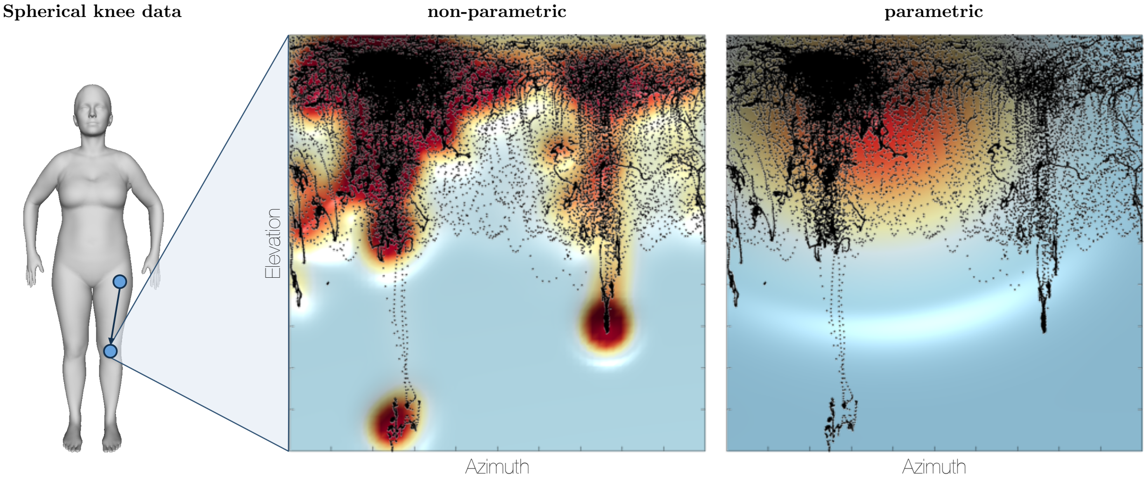


**Figure 2.2: Parametric vs. Non-parametric Local Models.** We show spherical coordinates of locations of the left knee (black points) and compare a non-parametric density estimate (left) with a parametric density estimate (right) [red $\widehat{=}$ high density; blue $\widehat{=}$ low density]. The non-parametric model is more flexible and able to capture the multimodality in the data.

$\{\widehat{f}_{X_i}\}_{i=1}^{n}$ that form a joint density $\widehat{f}_{\mathbf{X}}$. Given $\widehat{G}$, learning $\widehat{f}_{\mathbf{X}}$ is a model selection task in the model space

$$\mathcal{M}(\widehat{G}) = \left\{ \widetilde{f}_{\mathbf{X}}(\mathbf{x};\theta) \,\middle|\, \theta \in \Theta,\ \widetilde{f}_{\mathbf{X}} \text{ factorizes over } \widehat{G} \right\}, \tag{2.9}$$

where $\Theta$ is the parameter space. Parametric models, where the dimensionality of $\Theta$ is finite, make assumptions about the shape of a distribution, such as the number of modes, which may not hold in practice. We can thus not be sure that *any* $\widehat{f}_{\mathbf{X}}(\mathbf{x};\theta) \in \mathcal{M}(\widehat{G})$, $\theta \in \Theta$, is close to the true generating distribution $f_{\mathbf{X}}(\mathbf{x})$.

In this work, we therefore focus on non-parametric graphical models, that is, graphical models with non-parametric local distributions. A rigorous definition of a non-parametric model is subtle. Following [168], we define a non-parametric model as one that makes as few assumptions about the generating distribution as possible. Fig. 2.2 highlights a common use case: While the non-parametric approach (left) is able to capture most of the structure in the generating distribution, the parametric maximum-likelihood estimate (right) results in a poor fit due to its limited expressiveness. In the remainder of this section, we review two widely used non-parametric methods that will play central roles as local models of Bayesian networks in all subsequent chapters: kernel density estimation and decision trees.

### 2.3.1 Kernel Density Estimation

Kernel density estimation [114, 128] is a non-parametric density estimation technique that places a scaled kernel over each training point. For our purpose, a *kernel* $\kappa$ is a symmetric density of a zero-mean random variable with positive variance, *cf.* [168]. Examples include the Gaussian, Epanechnikov, tricube, and uniform kernel (Fig. 2.3a). Every 1-dimensional kernel $\kappa$ can be turned into a $d$-dimensional product kernel $\kappa_d(\mathbf{x}) = \prod_{j=1}^{d} \kappa(x_j)$.

Formally, a kernel density estimate is defined as

$$\widehat{f}_{\mathbf{B}}^{\mathcal{D}_{\mathbf{x}}}(\mathbf{x}) = \frac{1}{N|\mathbf{B}|} \sum_{i=1}^{N} \kappa_d\big(\mathbf{B}^{-1}\big(\mathbf{x} - \mathbf{x}^{(i)}\big)\big), \tag{2.10}$$

where $\mathbf{B} = (b_{ij})_{i,j=1,\dots,d} \in \mathbb{R}^{d\times d}$ is a positive-definite bandwidth matrix that controls the smoothness of $\widehat{f}_{\mathbf{B}}^{\mathcal{D}_{\mathbf{x}}}(\mathbf{x})$.[3] Equivalently, we can express $\widehat{f}_{\mathbf{B}}^{\mathcal{D}_{\mathbf{x}}}(\mathbf{x})$ as a convolution between the empirical density and a product kernel,

$$\widehat{f}_{\mathbf{B}}^{\mathcal{D}_{\mathbf{x}}}(\mathbf{x}) = (\widehat{f}_{\text{emp}} * \kappa_d)(\mathbf{x}) = \int_{\mathbb{R}^d} \widehat{f}_{\text{emp}}(\mathbf{x} - \mathbf{t}) \cdot \kappa_d(\mathbf{t})\ \mathrm{d}\mathbf{t}, \tag{2.11}$$

where $\widehat{f}_{\text{emp}}(\mathbf{x}) = \frac{1}{N}\sum_{i=1}^{N} \delta\big(\mathbf{x} - \mathbf{x}^{(i)}\big)$ and $\delta$ is the *Dirac delta function.*

In contrast to the kernel $\kappa_d$, which is not of great importance [168], the bandwidth matrix $\mathbf{B}$ is crucial. We discuss two popular ways for its selection: plug-in estimation and least-squares cross-validation.

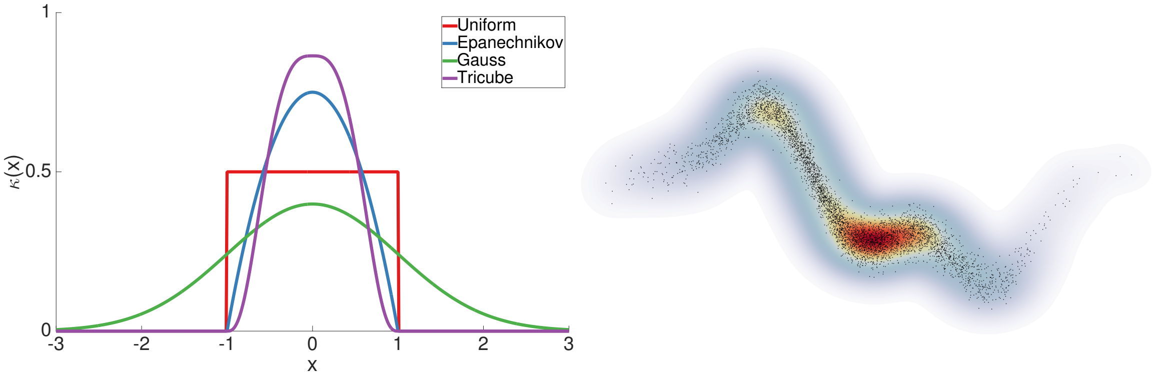
**Plug-in Estimation [141].** Plug-in estimation gives good results if the true density can be assumed to be smooth. In this case, one can use a bandwidth matrix proportional to the square root of the sample covariance matrix $\widehat{\mathbf{\Sigma}} = (\widehat{\sigma}_{ij}^2)_{i,j=1,\dots,d} \in \mathbb{R}^{d\times d}$,

$$\mathbf{B} = N^{\frac{-1}{d+4}} \sqrt{\widehat{\mathbf{\Sigma}}}, \tag{2.12}$$

which is a (not necessarily optimal) generalization of Scott's rule $b_{ii} = N^{\frac{-1}{d+4}} \hat{\sigma}_{ii}$ for diagonal bandwidth matrices. The latter is, in turn, a simplification of the normal reference rule $b_{ii} = \left(\frac{4}{(d+2)N}\right)^{\frac{1}{d+4}} \hat{\sigma}_{ii}$, which is asymptotically optimal under the assumption of a Gaussian density and a Gaussian kernel.

---

[3] We will often drop the reference to the training set $\mathcal{D}_{\mathbf{x}}$ or the bandwidth $\mathbf{B}$.



**(a)** Kernels. **(b)** Superposition of Kernels.

**Figure 2.3: Overview: Kernel Density Estimation. (a)** Four popular unscaled kernels, *i.e.*, proper densities. Note that the Gaussian kernel is the only one with infinite support. **(b)** Superposition of scaled kernels over each training point generates high density regions in densely sampled areas. [red ≙ high density; blue/white ≙ low/negligible density]

**Least-squares Cross-validation [129, 133].** Cross-validation is an effective bandwidth selection method for diagonal bandwidth matrices $\mathbf{B} = \mathrm{diag}(\mathbf{b}), \mathbf{b} \in \mathbb{R}^d_+$. In a nutshell, we minimize the mean integrated squared error with respect to $\mathbf{b}$ by means of a leave-one-out cross-validation estimator,

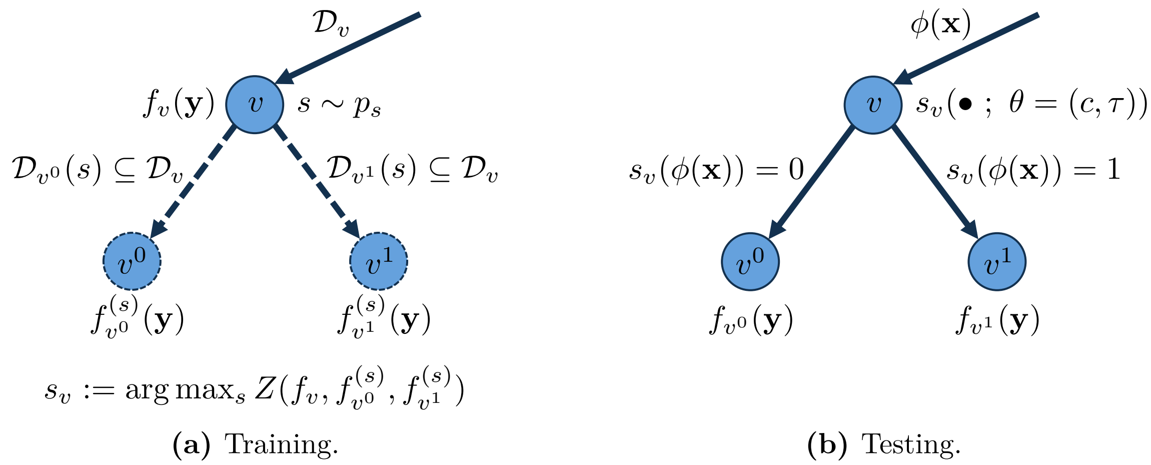
$$\begin{aligned}\mathbf{b} &= \operatorname*{arg\,min}_{\mathbf{b}'\in\mathbb{R}^d_+} \mathbb{E}\left[\int \left(\widehat{f}^{\mathcal{D}_\mathbf{x}}_{\mathrm{diag}(\mathbf{b}')}(\mathbf{x}) - f(\mathbf{x})\right)^2 \mathrm{d}\mathbf{x}\right]\\ &= \operatorname*{arg\,min}_{\mathbf{b}'\in\mathbb{R}^d_+} \mathbb{E}\left[\int \widehat{f}^{\mathcal{D}_\mathbf{x}}_{\mathrm{diag}(\mathbf{b}')}(\mathbf{x})^2 \,\mathrm{d}\mathbf{x} - \frac{2}{N}\sum_{i=1}^{N} \widehat{f}^{\mathcal{D}_\mathbf{x}\setminus\mathbf{x}^{(i)}}_{\mathrm{diag}(\mathbf{b}')}\big(\mathbf{x}^{(i)}\big)\right],\end{aligned} \tag{2.13}$$

and approximate the cross-validation score $\mathrm{CV}(\mathbf{b}')$ inside the last expectation by

$$\mathrm{CV}(\mathbf{b}') \approx \frac{1}{N^2|\mathrm{diag}(\mathbf{b}')|}\sum_{j,k=1}^{N} \kappa^*_d\big(\mathrm{diag}(\mathbf{b}')^{-1}\big(\mathbf{x}^{(j)} - \mathbf{x}^{(k)}\big)\big) + \frac{2\kappa_d(\mathbf{0})}{N|\mathrm{diag}(\mathbf{b}')|}, \tag{2.14}$$

where $\kappa^*_d(\mathbf{x}) = (\kappa_d * \kappa_d)(\mathbf{x}) - 2\kappa_d(\mathbf{x})$ [168]. Under the assumption that the minimizer of Eq. (2.14) is close to the minimizer of Eq. (2.13), an FFT-based grid search [146, 147] will return a good bandwidth matrix.

While the two approaches above work well in many scenarios that are of practical relevance, we note that the literature on optimal bandwidth selection is extensive and refer to [76, 163, 141] as excellent starting points for further details.


$\mathcal{D}_v$
$f_v(\mathbf{y})$
$v$
$s \sim p_s$
$\mathcal{D}_{v^0}(s) \subseteq \mathcal{D}_v$
$\mathcal{D}_{v^1}(s) \subseteq \mathcal{D}_v$
$v^0$
$v^1$
$f^{(s)}_{v^0}(\mathbf{y})$
$f^{(s)}_{v^1}(\mathbf{y})$
$s_v := \arg\max_s Z(f_v, f^{(s)}_{v^0}, f^{(s)}_{v^1})$
$\phi(\mathbf{x})$
$v$
$s_v(\bullet\ ;\ \theta = (c, \tau))$
$s_v(\phi(\mathbf{x})) = 0$
$s_v(\phi(\mathbf{x})) = 1$
$v^0$
$v^1$
$f_{v^0}(\mathbf{y})$
$f_{v^1}(\mathbf{y})$

**(a)** Training. **(b)** Testing.

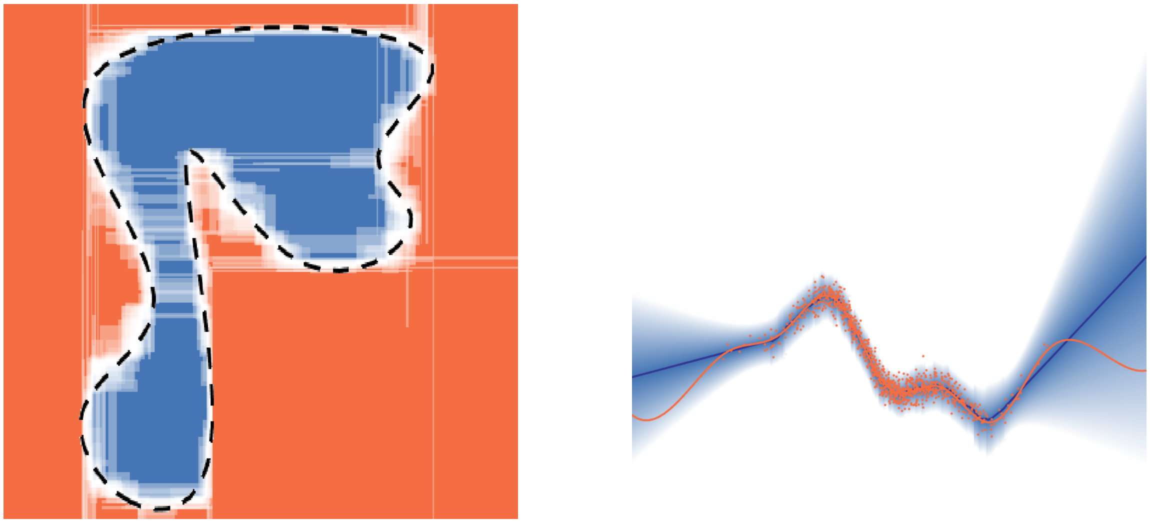
**Figure 2.4: Overview: Decision Trees. (a)** At training time, a candidate split $s$ at leaf node $v$ induces child nodes $v^0, v^1$ and a split-dependent partition of the data $\mathcal{D}_v$ at $v$. We evaluate a collection of such splits by means of a model-dependent score function $Z$, split $v$ according to the split $s_v$ with maximum score and proceed recursively. **(b)** At test time, we inject a feature vector $\phi(\mathbf{x})$ into the tree and, guided by a series of split evaluations, select a particular leaf node for prediction.

### 2.3.2 Decision Trees

Decision trees are a distributed approach to statistical modeling that partitions a feature space $\Phi$ into disjoint regions and associates each region with a probabilistic model. Since the development of algorithms for their automatic generation in the early works of Breiman (CART [16]) and Quinlan (ID3 [121], C4.5 [122]), they have been used for a broad range of tasks, including classification, regression, and density estimation, with applications to various computer vision domains [143, 35].

Formally, a decision tree models a conditional output distribution $f_{\mathbf{Y}}(\mathbf{y} \mid \phi(\mathbf{x}))$ through a directed rooted binary tree $\mathcal{T} = (V, E)$, where $V = V_I \cup V_L$ is a disjoint union of inner nodes $V_I$ and leaf nodes $V_L$ that are connected by a directed edge set $E \subseteq V \times V$. Each inner node $v \in V_I$ is associated with a binary split function $s_v : \Phi \times \mathbf{\Theta} \to \{0, 1\}$ with parameter space $\mathbf{\Theta}$ and each leaf node $\ell \in V_L$ is associated with a density $f_\ell$.

The role of the split functions is to guide a feature vector $\phi(\mathbf{x}) \in \Phi$ from a designated root node $v_{\text{root}} \in V_I$ to a leaf node $\ell(\phi(\mathbf{x})) \in V_L$. At each inner node $v$ with children $v^0$ and $v^1$, we send $\phi(\mathbf{x})$ to $v^{s_v(\phi(\mathbf{x});\ \theta_v)}$ until we reach a leaf node. This is illustrated in Fig. 2.4b. The split functions can, in principle, be any binary



**(a)** Classification Tree. **(b)** Regression Tree.

**Figure 2.5: Decision Tree Classes.** Depending on the type of leaf model, we obtain either a classification tree or a regression tree. **(a)** A two-dimensional feature space with color-coded MAP predictions of a classification tree (orange, blue) and a non-linear ground truth class boundary (dashed line). Hue encodes uncertainty, which is larger near the border. **(b)** Noisy samples from a ground truth spline (orange) and regression tree predictions (blue). The blue bands indicate three standard deviations.

classifier. In practice, axis-aligned hyperplane splits

$$s(\phi(\mathbf{x});\ \theta = (c, \tau)) = \mathbb{I}[\phi(\mathbf{x})_c \geqslant \tau] \tag{2.15}$$

are predominant due to their simplicity and efficiency; we use them throughout this work. Mapping features to leaves effectively partitions the input space into connected, disjoint regions $\Phi_\ell = \{\phi(\mathbf{x}) \in \Phi \mid \ell(\phi(\mathbf{x})) = \ell\}$, such that $\Phi = \dot{\bigcup}_{\ell \in V_L} \Phi_\ell$ and

$$f_{\mathbf{Y}}(\mathbf{y} \mid \phi(\mathbf{x})) = f_{\ell(\phi(\mathbf{x}))}(\mathbf{y}). \tag{2.16}$$

If the leaf densities are discrete, we refer to $\mathcal{T}$ as a classification tree, whereas in the continuous case we speak of a regression tree. A collection of decision trees together with an aggregate function (*e.g.*, the arithmetic mean) is called a decision forest. Fig. 2.5 illustrates the partition of the feature space by means of two examples that highlight the non-parametric nature of decision trees, one classification example and one regression example. Deep trees correspond to a fine-grained partition, which allows modeling of arbitrary class boundaries and non-linear regression functions.

**Training.** In order to learn a decision tree $\mathcal{T}$, we use a supervised training set $\mathcal{D}$ to estimate a split for each inner node and a density for each leaf node. Starting with a single leaf node $v_{\text{root}} \in V_L$ with leaf model $f_{v_{\text{root}}}$ and associated data $\mathcal{D}_{v_{\text{root}}} := \mathcal{D}$, we grow a tree by recursively splitting leaf nodes. To split a node $v$, we introduce child nodes $v^0, v^1$ and draw a candidate split $s$ from a (possibly data-dependent) split distribution $p_s$. Using $s$, we obtain a data partition $\mathcal{D}_v = \mathcal{D}_{v^0}(s) \biguplus \mathcal{D}_{v^1}(s)$ that we use to fit new leaf models $f_{v^0}^{(s)}, f_{v^1}^{(s)}$. We repeat this process for a set $\mathcal{S}$ of candidate splits and select

$$s_v := \operatorname*{arg\,max}_{s \in \mathcal{S}} Z(f_v, f_{v^0}^{(s)}, f_{v^1}^{(s)}), \tag{2.17}$$

where $Z$ is a split objective, such as information gain (classification trees) or residual error (regression trees). We assign $s_v$ to $v$, $f_{v^0} = f_{v^0}^{(s_v)}$ to $v^0$, $f_{v^1} = f_{v^1}^{(s_v)}$ to $v^1$, and proceed with the next leaf node until we meet a stopping criterion, such as a maximum tree depth or a minimum leaf sample size.

### 2.3.3 Ensemble Models

Kernel density estimation and decision forests are both examples of ensemble models. More generally, an ensemble model comprises a collection of base models and a method to form a consensus prediction from the set of base predictions. It is typically much more flexible than any of the underlying base models, which can be learned either independently (*e.g.*, bootstrap aggregation [15]) or sequentially (*e.g.*, AdaBoost [48]).

For our purpose, it is sufficient to think of an ensemble model as an instance of a *mixture distribution*. A mixture distribution $f_{\mathbf{X}}(\mathbf{x})$ with mixture weights $\{\alpha_i\}_{i=1}^m$ is a convex combination of $m$ component distributions $\{f_{\mathbf{X}^{(i)}}\}_{i=1}^m$,

$$f_{\mathbf{X}}(\mathbf{x}) = \sum_{i=1}^{m} \alpha_i f_{\mathbf{X}^{(i)}}(\mathbf{x}). \tag{2.18}$$

**Example.** A kernel density estimate $\widehat{f}_{\mathbf{B}}^{\mathcal{D}_{\mathbf{x}}}(\mathbf{x})$ with a Gaussian kernel is a Gaussian mixture with $|\mathcal{D}_{\mathbf{x}}|$ components and a uniform weight distribution $\alpha_i = |\mathcal{D}_{\mathbf{x}}|^{-1}$.

**Example.** A decision forest with $C$ trees is a conditional mixture distribution consisting of the $C$ selected leaf models.

From a statistical point of view, it is interesting to look at the variance of a mixture model as a measure of uncertainty. To keep the notation uncluttered, we focus on the 1-dimensional case; generalizations to multiple dimensions are straightforward. Denoting the means and variances of the $m$ component distributions with $\mu_i$ and $\sigma_i^2$, respectively, and assuming that $X$ has a mixture distribution as defined in Eq. (2.18), we have

$$\mathbb{V}[X] = \sum_{i=1}^{m} \alpha_i \left( \left( \mu_i - \sum_{j=1}^{m} \alpha_j \mu_j \right)^2 + \sigma_i^2 \right), \tag{2.19}$$

that is, a weighted sum of intra-model variances and squared inter-model variances. Important special cases include the variance of a Gaussian mixture, which follows directly from Eq. (2.19), and that of a categorical mixture, which evaluates to

$$\mathbb{V}[[X = j]] = \sum_{i=1}^{m} \alpha_i \big( \overline{p_j}^2 - 2\overline{p_j} p_{ij} + p_{ij} \big), \tag{2.20}$$

where $p_{ij} := p(X^{(i)} = j)$ and $\overline{p_j} := \sum_{k=1}^{m} \alpha_k p_{kj}$.

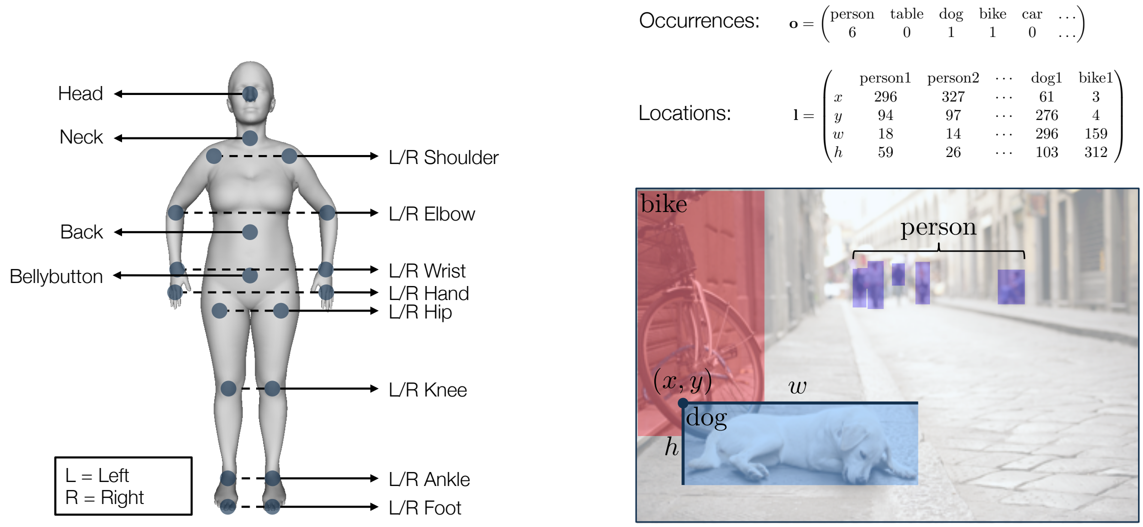
## 2.4 Semantics

In the previous sections, we discussed graphical models and their associated distributions without assigning any semantic meaning to them. In a typical computer vision task, the semantics of a training point $\mathbf{x} \in \mathbb{R}^d$ is given by the type of object we are modeling. We give two examples that will be relevant throughout this work:

1. A *human pose* is a collection of two- or three-dimensional points that correspond to joint locations, see Fig. 2.6a. If there are $n'$ joints in total, we can parameterize a pose as a point in a $2n'$- or $3n'$-dimensional Euclidean space. In chapters 3 and 4 of this work, we use a three-dimensional pose model consisting of $n' = 20$ joints.

2. A *natural scene* is a collection of four-dimensional bounding boxes that correspond to object locations and associated object categories, see Fig. 2.6b. Note that, in constrast to a human pose, a natural scene has no a-priori fixed dimensionality, because the number of objects is unbounded. In chapter 5 of this work, we address this challenge with a hierarchical approach.

The factorization of a pose or scene into a graphical model requires two important choices, parameterization and granularity.

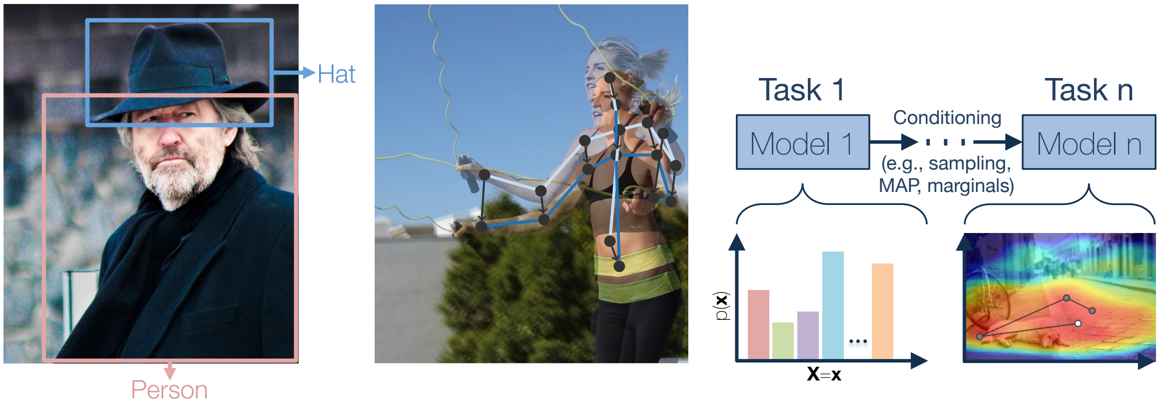

Head
Neck
Back
Bellybutton
L/R Shoulder
L/R Elbow
L/R Wrist
L/R Hand
L/R Hip
L/R Knee
L/R Ankle
L/R Foot
L = Left
R = Right
Occurrences: o = (person 6, table 0, dog 1, bike 1, car 0, ...)
Locations: l = (person1: x 296, y 94, w 18, h 59; person2: x 327, y 97, w 14, h 26; ... ; dog1: x 61, y 276, w 296, h 103; bike1: x 3, y 4, w 159, h 312)
bike
person
(x, y)
w
h
dog

**(a)** Human Pose. **(b)** Natural Scene.

**Figure 2.6: Parameterizations of Human Poses and Natural Scenes.** **(a)** The three-dimensional locations of $n' = 20$ joints in the human body allow to model a human pose in a Euclidean space with $d = 60$ dimensions. **(b)** A natural scene with $n'$ object instances and $K$ possible object categories can be thought of as a point in a $(4n' + K)$-dimensional Euclidean space.

**Parameterization.** We can model a $d$-dimensional training point in Euclidean space using a variety of relative or absolute coordinate systems: For poses, these include collections and combinations of polar, spherical, cylindrical, or Cartesian coordinates for each joint. For scenes, we can represent the individual bounding boxes in terms of two diagonally opposing corners, one corner and their spatial extent, or their center, scale and aspect ratio. Although these coordinate spaces are all equivalent in the sense that, with some care, they can uniquely identify a training point, the chosen parameterization can affect the identifiability of the model, the weight of a coordinate axis and, ultimately, the training procedure itself.

**Granularity.** Given a $d$-dimensional parameterization, the number of nodes in a Bayesian network is given by a partition of the coordinates. Typically, this assignment will be clear from the context: A pose network will usually consist of $n = n'$ two- or three-dimensional nodes, one for each joint, but we could also choose to model all $d$ dimensions separately or to collapse all joints belonging to the same limb into one node. If the dimensionality of a node is larger than 1, we make the implicit assumption that there are no conditional independencies between those coordinates.



**(a)** Spatial Context. **(b)** Temporal Context. **(c)** Extrinsic Semantic Context.

**Figure 2.7: Types of Context.** Context can be static, dynamic, and semantic. In all three cases, the flow of information is determined by the conditional independence assumptions of the underlying factorization. Some specific instances: **(a)** Spatial context between *variables* (*e.g.*, object instances) influences their locations in space. **(b)** Temporal context between *objects* (*e.g.*, human poses) influences their trajectory in time. **(c)** Extrinsic semantic context between *models* (*e.g.*, object categorization, object detection) influences their behavior across tasks.

## 2.5 Context

We can think of a parameterized object $\mathbf{X}$, such as a pose or a scene, as a collection of variables whose semantics are given according to section 2.4. If we obtain evidence for one of those variables, *i.e.*, if we move from the joint distribution to a conditional distribution, this context information will affect our beliefs about the states of the remaining variables [65]. Graphical models provide a natural framework to incorporate such information in a sparse and tractable way: In a Bayesian network, where each variable is represented by a node, introduction of evidence into the model affects the flow of information along the edges of the graph according to the *d*-separation criterium, as discussed in Eq. (2.7).

In this work, we distinguish between three different types of context information: Static context, dynamic context, and semantic context.

**Static context.** A single and independent parameterized object is said to be static and context between its variables is referred to as *static context*. Examples include human poses (chapter 3) and natural scenes (chapter 5), where we have spatial context as a specific type of static context. For instance, persons and hats provide strong cues about each other and knowing the location of one reduces the uncertainty about the other (Fig. 2.7a). Static context can take on many forms beyond spatial context, *e.g.*, expression levels of genes or character traits of people, but they will not play a role in this work.

**Dynamic context.** If a collection of parameterized objects can be written as an ordered sequence of dependent random variables, they provide *dynamic context* for each other. This definition implies that dynamic context typically relies on a static type of context that is being extended across objects. The predominant manifestation of dynamic context is temporal context, which is available, inter alia, in video and motion capture sequences (chapter 4). As an example, knowing the pose of a person at previous points in time limits the possible poses in the present (Fig. 2.7b). Other types of dynamic context, *e.g.*, serial dilutions in a pharmaceutical setting, are possible but much less common.

**Semantic context.** Just like variables and objects, models can influence each other, too. We refer to this type of interaction as *semantic context*. A simple instance of semantic context is a chain-like sequence of conditional models (Fig. 2.7c). The elements along such a chain can be homogenous and solve the same task (chapter 6) or heterogenous and solve different tasks (chapter 5). In the former case, we speak of intrinsic semantic context, whereas the latter case is referred to as extrinsic semantic context. The distribution of a dependent model can be conditioned based on arbitrary information derived from the model it is depending on, such as its MAP state, its marginals, or a sample. Furthermore, a parent model can affect not just the local models but the entire structure of a child model, including its topology and even the semantics of its random variables.

Different types of context are not mutually exclusive and a probabilistic model can feature more than one type: A hierarchical model exploiting semantic context can contain individual models that make use of both static and dynamic information.

## 2.6 Complexity

Solving density estimation, classification, or regression tasks with a Bayesian network involves learning of and inference in distributions that factorize over a graph. From a theoretical point of view, we would want to perform these tasks in a setting that makes no structural assumptions, which implies a fully-connected graph

($\widehat{=}$ no topological independence assumptions) with non-parametric local models ($\widehat{=}$ no distributional shape assumptions). Such a model would be ideal in the sense that it could return (max-)marginals corresponding to those in the true density [150].

In practice, this is not feasible due to insufficient training data and intractable inference procedures. The following two sections make this statement more rigorous by summarizing some fundamental complexity results for Bayesian networks from both a learning and an inference perspective. The insights obtained will then motivate the class of models we use in the remainder of this work.

### 2.6.1 Learning

Learning a Bayesian network $\mathcal{B}$ involves the selection of a global topology $\widehat{G} = (\mathbf{X}, \widehat{E})$ and local models $\widehat{f}_{X_i}(x_i \mid \mathrm{pa}(x_i))$ from a dataset $\mathcal{D}_{\mathbf{x}}$ so that the joint distribution $\widehat{f}_{\mathbf{X}}(\mathbf{x}) = \prod_{i=1}^{n} \widehat{f}_{X_i}\big(x_i \mid \mathrm{pa}_{\widehat{G}}(x_i)\big)$ is close to the generating distribution $f_{\mathbf{X}}(\mathbf{x})$ w.r.t. some measure of distance.

If we use KL-divergence as a measure of distance, the estimation error $\mathrm{KL}\Big(f_{\mathbf{X}} \,\Big\|\, \widehat{f}_{\mathbf{X}}\Big)$ is composed of a topological error and a distributional error,

$$\mathrm{KL}\Big(f_{\mathbf{X}} \,\Big\|\, \widehat{f}_{\mathbf{X}}\Big) = \mathrm{KL}\Big(f_{\mathbf{X}} \,\Big\|\, \widehat{f}^{*}_{\mathbf{X}}\Big) + \Big(\mathrm{KL}\Big(f_{\mathbf{X}} \,\Big\|\, \widehat{f}_{\mathbf{X}}\Big) - \mathrm{KL}\Big(f_{\mathbf{X}} \,\Big\|\, \widehat{f}^{*}_{\mathbf{X}}\Big)\Big), \tag{2.21}$$

where $\widehat{f}^{*}_{\mathbf{X}}$ is the projection of $f_{\mathbf{X}}$ onto the set of distributions that are representable in the network structure $\widehat{G}$ (Fig. 2.9).

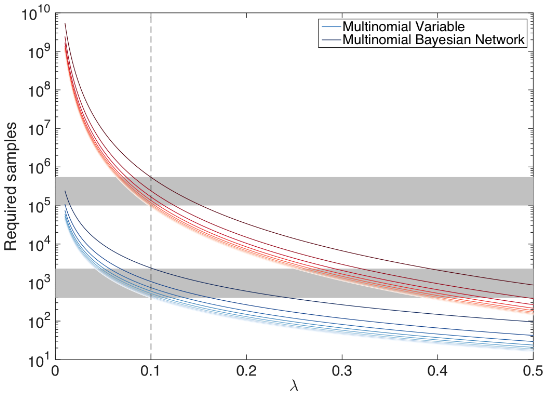
For categorical local models with $C$ categories, the distributional error evaluates to

$$\mathrm{KL}\Big(f_{\mathbf{X}} \,\Big\|\, \widehat{f}_{\mathbf{X}}\Big) - \mathrm{KL}\Big(f_{\mathbf{X}} \,\Big\|\, \widehat{f}^{*}_{\mathbf{X}}\Big) = \sum_{i=1}^{n} \mathrm{KL}\Big(f_{X_i}\big(x_i \mid \mathrm{pa}_{\widehat{G}}(x_i)\big) \,\Big\|\, \widehat{f}_{X_i}\big(x_i \,|\mathrm{pa}_{\widehat{G}}(x_i)\big)\Big), \tag{2.22}$$

which is the sum of errors in the local models [81]. A PAC-bound analysis of this case provides a lower bound on the number of samples that are required to achieve a desired accuracy $\epsilon$ with probability $\delta$ [64]: If $N \geqslant \frac{(1+\epsilon)^2}{2\epsilon^2 \lambda^{2(\mathrm{pa}_{\max}+1)}} \log \frac{nC^{\mathrm{pa}_{\max}+1}}{\delta}$, the distributional error of a maximum-likelihood (ML) estimate can be bound by

$$p\Big(\mathrm{KL}\Big(f_{\mathbf{X}} \,\Big\|\, \widehat{f}_{\mathbf{X}}\Big) - \mathrm{KL}\Big(f_{\mathbf{X}} \,\Big\|\, \widehat{f}^{*}_{\mathbf{X}}\Big) \leqslant n\epsilon\Big) \geqslant 1 - \delta, \tag{2.23}$$

where $\lambda$ is the minimum conditional probability in the true graph $G$ and $\mathrm{pa}_{\max}$ is the maximum indegree in the estimated graph $\widehat{G}$. A visualization of this situation is shown in Fig. 2.8a. It is easy to see that the dependence of $N$ on small conditional probabilities or large indegrees is dramatic.



| Dimension | Equivalent Sample Size | | |
|---|---|---|---|
| | AMIAE | RCV(0) | RRMSE(0) |
| 1 | 50 | 50 | 50 |
| 2 | 222 | 284 | 307 |
| 3 | 1087 | 1131 | 1468 |
| 4 | 5582 | 4317 | 7149 |
| 5 | 29462 | 16872 | 37242 |
| 6 | 159440 | 68559 | 208441 |
| 7 | 884474 | 289832 | 1245809 |
| 8 | 5029367 | 1270584 | 7894492 |

**(a)** PAC-bound analysis of a categorical network. **(b)** AMIAE analysis of a KDE network.

**Figure 2.8: Learning Bayesian Networks.** **(a)** Visualization of Eq. (2.23) for a categorical variable (blue) and a tree-structured categorical network with 80 nodes (red). Settings: $\delta := 0.05$, $C := 10$, and $0.5 \leqslant \epsilon \leqslant 5$ (step size: 0.5). **(b)** Number of required samples in dimensions $2-8$ to obtain the same optimal AMIAE of 0.5987 as with 50 samples in dimension 1 [142]. We also restate two local criteria, the root coefficient of variation (RCV) and the relative root mean squared error (RRMSE).

For KDE-based local models, which grow with and depend on the training set $\mathcal{D}_{\mathbf{x}}$, one can analyze an asymptotic version of the mean integrated absolute error

$$\mathrm{MIAE}(\widehat{f}_{\mathbf{B}}^{\mathcal{D}_{\mathbf{x}}}) = \mathbb{E}\left[\int \left|\widehat{f}_{\mathbf{B}}^{\mathcal{D}_{\mathbf{x}}}(\mathbf{x}) - f_{\mathbf{X}}(\mathbf{x})\right| \,\mathrm{d}\mathbf{x}\right], \tag{2.24}$$

called AMIAE [141], to get an impression of the distributional error. Scott and Wand [142] conduct a rigorous investigation of this case and find a strong dependence of $N$ on the dimensionality. A summary of those results is provided in Table 2.8b. When learning a Bayesian network, we incur this type of error for each local model $\widehat{f}_{X_i}\big(x_i \,\big|\, \mathrm{pa}_{\widehat{G}}(x_i)\big)$, meaning that the error increases with the indegree of the corresponding node in $\widehat{G}$.

## 2.6.2 Inference

The complexity of exact inference in a Bayesian network $\mathcal{B} = (G, f_{\mathbf{X}})$ by means of variable elimination or message passing is determined by the *treewidth* of $G = (\mathbf{X}, E)$, [127], which is defined as

$$\mathrm{tw}(G) := \min_{\overline{G}} \omega(\overline{G}) - 1, \tag{2.25}$$

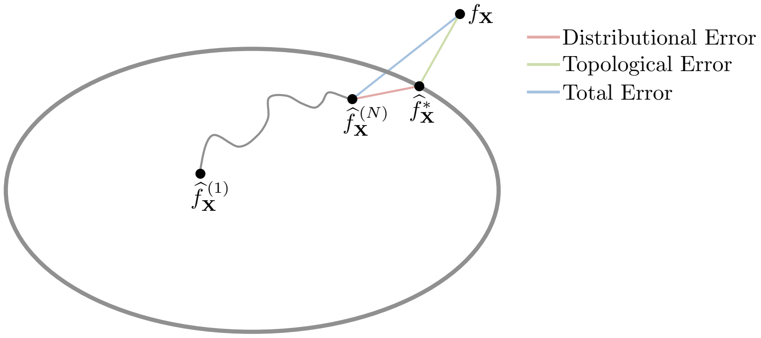

$f_{\mathbf{X}}$
Distributional Error
Topological Error
Total Error
$\widehat{f}_{\mathbf{X}}^{(N)}$
$\widehat{f}_{\mathbf{X}}^{*}$
$\widehat{f}_{\mathbf{X}}^{(1)}$

**Figure 2.9: Topological and Distributional Error.** The topology $\widehat{G}$ limits the set of distributions that can be represented by a Bayesian Network $\mathcal{B} = (\widehat{G}, \widehat{f}_{\mathbf{X}})$ (gray circle). As $N \to \infty$, the distributional error (red line) can converge to 0, but the total error (blue line) can never fall below the topological error (green line).

where $\overline{G}$ is a chordal completion of $G$ and $\omega(\bullet)$ returns the clique number of a graph [10]. From an inference point of view, we are therefore interested in graphical models with low treewidth. While the treewidth decision problem for general graphs is known to be NP-complete [5], closed-form solutions do exist for some topological classes, including

- graphs without edges: $\mathrm{tw}(G) = 0$.
- directed polytrees [31]: $\mathrm{tw}(G) = \max_{X \in \mathbf{X}} |\mathrm{pa}_G(X)|$.
- undirected square grids [126]: $\mathrm{tw}(G) = \sqrt{|\mathbf{X}|}$.
- fully-connected graphs: $\mathrm{tw}(G) = |\mathbf{X}| - 1$.

Given a suitable graph structure, we can use one of the available variants of belief propagation [116, 78, 94] to compute (max-)marginals. If the local models belong to the exponential family, the messages can be computed in closed-form [7]. For KDE-based local models, we can choose from a pool of non-parametric belief propagation methods [151, 70, 112].

### 2.6.3 Consensus Models

The preceding sections have shown that accurate learning and tractable inference are difficult to achieve, even with large training sets and powerful computing resources. Taking those results into account, we take another look at the global topology and local distributions of a graphical model and propose a class of Bayesian networks that offers a compromise in terms of representability and applicability.

**Topology.** Tractable topologies have low treewidth, which means that we need to identify the optimal topology in $\mathcal{G}(k) := \{G \mid G \text{ graph with } \operatorname{tw}(G) \leqslant k\}$. Although it is known that $\mathcal{G}(k)$ coincides with the class of partial $k$-trees [9], this task turned out to be challenging and, without further constraints, depends on heuristics [38, 108].

For static models with a polytree-structured topology, treewidth coincides with the maximum number of parents, a parameter that is more accessible and, moreover, directly related to the dimensionality of a local model. A conceptually simple way to do justice to both learning and inference is thus to enforce a strict upper bound $k$ on the indegree of a polytree. Learning an optimal polytree is NP-complete unless $k = 1$ [32], in which case we obtain the efficient subclass of arborescences (or directed rooted trees) that we will use extensively. In order to select an arborescence, it is sufficient to choose an undirected tree, because all $n$ arborescences with the same undirected skeleton are equivalent in terms of their conditional independence assumptions, as can be easily seen using $d$-separation. However, this is still a non-trivial problem, because there are $n^{n-2}$ undirected trees according to Cayley's formula [1]. We leverage ideas from [26] to address different flavors of this problem in our applications to human poses (chapter 3) and natural scenes (chapter 5).

For dynamic context models with an a-priori fixed topology, hidden Markov models (HMMs) have been very popular due to their expressive architecture and tractable inference [21]. They are essentially arborescences whose root is the first hidden variable, albeit with different semantics than in the static case. However, computing the likelihood of an observed sequence requires a full forward pass of message passing. If non-Markovian dependencies between the observed variables can be well approximated with an order-$L$ Markov assumption, we can employ a more efficient hybrid approach between a latent variable model and a Markov model. To this end, we combine ideas from switching auto-regressive models [7] and HMMs with deterministic hidden transitions [6] with the non-parametric power of decision forests. As a result, we obtain a dynamic context model with low treewidth that can be interpreted either as an $L$-th order Markov model with non-parametric conditional distributions or as a latent variable model with time-decoupled state sequence. We pursue this approch to model human motion (chapter 4).

**Local models.** In many cases, models with low treewidth are the only viable option, for instance due to small training sets, high-dimensional nodes, or limited computing resources. On the other hand, it is to be expected that some of the imposed conditional independence assumptions do not hold in the true generating distribution. While this introduces a type of bias into the model that cannot – and with regard to generalization performance quite possibly should not [84, 38] – be avoided, we can eliminate *additional* bias by employing flexible local models, *i.e.*, categorical models in the discrete case and non-parametric approaches, such as decision forests or kernel density estimates, in the continuous case.

The reasoning above is the motivation behind the architecture of our models in the subsequent chapters: We combine low-treewidth topologies with non-parametric local models to obtain an efficient and flexible bias-variance trade-off.

# Chapter 3

# Non-parametric Priors of Human Pose

The previous chapter has given a brief introduction to theoretical concepts and practical considerations that relate to non-parametric models for structured data. In this chapter, we present a first application of those techniques to the modeling of human poses as introduced in section 2.4. Assessing the likelihood of a parameterized pose $\mathbf{x} \in \mathbb{R}^d$ is a task that carries strong spatial dependencies. Motivated by our exposition in section 2.6, we encode this static context information by means of a Bayesian network with low treewidth and high flexibility. Unlike previous works, we do not make any parametric assumptions and learn both an optimal arborescence and all local models in a fully data-driven way.

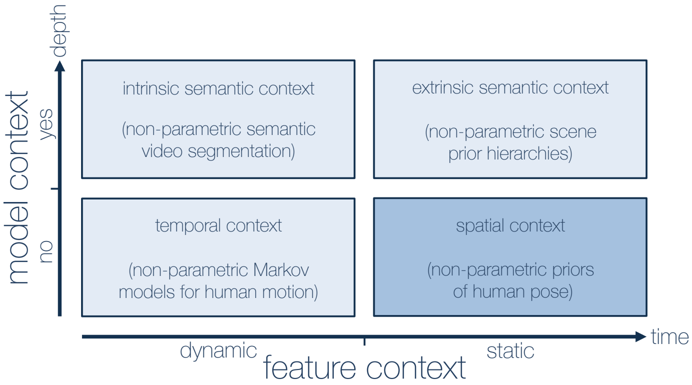

depth
model context
yes
no
intrinsic semantic context
(non-parametric semantic video segmentation)
extrinsic semantic context
(non-parametric scene prior hierarchies)
temporal context
(non-parametric Markov models for human motion)
spatial context
(non-parametric priors of human pose)
time
dynamic
static
feature context

**Figure 3.1: Key Message: Static Context.** Non-parametric density estimates can be used as local models of a tree-structured Bayesian network with optimal topology to obtain efficient and flexible priors of human pose.

## 3.1 Introduction

Reasoning about human pose is a key ingredient in recent successful applications of computer vision systems [143]. Accurately capturing the variability of human pose is challenging because there is both a variation between different persons as well as a combinatorial number of possible poses a single person can assume. In this chapter we propose a *pose prior*, a generative probabilistic model of static human pose. Such a general purpose prior model is useful in at least two ways; first, it can synthesize realistic poses that can be used for rendering or for generating plausible pose hypotheses, and second, in the context of a larger pose estimation or tracking system it can score any given pose by how a-priori likely it is, serving as a more specific regularization term. A good pose prior must generalize to unseen poses and persons. If it was merely reproducing poses seen in a training dataset it could never span the full variability of human pose. In order to generalize the prior must be *compositional*: it must represent the variations of parts that frequently occur together and produce a pose by combining these parts.

We achieve compositionality by factorizing the pose representation into a Bayesian network [81] with low treewidth. The sparse hierarchical structure of the network enables efficient computation of likelihoods and exact sampling. To apply a Bayesian network on human pose data we need to specify the network structure and conditional probability distributions along the network and it is here that we make two novel technical contributions. First, we enhance the representative power of Bayesian networks by proposing *non-parametric Bayesian networks* in which the conditional distributions are represented by conditional kernel density estimates. Second, we use structure learning to obtain the network structure by finding parts of the pose that strongly depend on each other, leveraging non-parametric mutual information estimators on continuous joint data.

Our data-driven approach is made possible by the recent availability of a large-scale dataset of human pose, the Human 3.6M dataset [71], which captures a large variety of poses and persons. We use this dataset to assess the generalization performance of our approach and demonstrate its good adaption to unseen test poses. Although our learned system is efficient, some applications require direct control of the runtime. For such scenarios we propose an approximation trade-off. With this approximation we demonstrate real-time scoring of Kinect tracker output.

### 3.1.1 Related Work

Pose priors are most often used within pose estimation systems and therefore some of the related works we discuss below incorporate a likelihood term that is computed from an observed image. Incorporating such an observation likelihood is possible in our model as well, but in the present work our focus is on a generative model. A natural idea to build a pose prior is to use the tree structure of the human skeleton as a starting point. Models that follow the skeletal structure are called *kinematic chain models* [14] and they allow us to incorporate prior beliefs about joint angles. In [96] the authors used a multivariate Normal distribution along the kinematic chain and estimate the parameters from motion capture data. The different choices of possible parametrizations in terms of joint angles or relative world coordinates in a kinematic tree model give rise to qualitatively different behaviours [60]. Despite this flexibility a kinematic tree model has clear limitations, as sharply argued in [93]; it is unable to express the coordination of different limbs and fails to represent global balance and gravity constraints. We will demonstrate that we can avoid these limitations by using a tree model that does not correspond to the kinematic chain but instead is chosen to optimally approximate the true distribution of poses. The resulting tree no longer corresponds to a skeleton (Fig. 3.2c and Fig. 3.4b) but retains all computational advantages of a tree-structured model.

Previous works have attempted to overcome the limitations of the kinematic tree model in different ways. In [18] the authors have used a global kernel density model on human pose. This model is global and does not reflect the combinatorial nature of human pose hence it is suitable only for modeling specific poses. Another approach proposed in [145] has been to add further interactions to the kinematic tree so that limb-limb coordination and penetration constraints are modelled. This is satisfying as a model but because the model now has cycles, exact inference becomes intractable and the authors have to resort to an expensive approximate particle belief propagation. Likewise in [174] a structure learning heuristic is used to learn a compositional model of pose; exact inference is again intractable and a heuristic based on likely hypotheses is used.

Another popular way to improve over the kinematic tree model is to add latent variables to the model. In [93] the authors augment the kinematic tree model by a few latent variables that are identified by factor analysis. The Gaussian Process latent variable model (GPLVM) [95] has been applied as a pose model [36]. In the GPLVM model a low-dimensional latent space is transformed to pose space by means of a Gaussian Process regression function. The GPLVM model has also been extended to incorporate a temporal model (GPDM) [164, 159]. The Laplacian eigenmap latent variable model (LELVM) [101] improves on the GPLVM by modeling the manifold of poses using a graph Laplacian and by providing tractable posterior inference in the latent space. An interesting recent model based on a large number of latent binary variables is the implicit mixture of conditional restricted Boltzmann machines (imCRBM) [154]; both estimation and inference are again approximate. While the global latent variable models (GPLVM and LELVM) are flexible they do not provide compositionality. In fact, each training pose is represented as one latent vector and they are not combined in an intelligent way.

## 3.2 Non-parametric Networks of Human Pose

In this section we introduce our non-parametric Bayesian network model of human pose and show its tractability.

In accordance with section 2.4, we represent a human pose by a $d$-dimensional vector whose components correspond either to angular (polar, spherical, cylindrical) or Cartesian coordinates of $n$ joints. Each pose decomposes on the joint level, $\mathbf{x} = [x_1, \ldots, x_n]^\top \in \mathbb{R}^d$, and we model the angle/position of joint $j$ by a possibly multi-dimensional random variable $X_j$. This setup fully specifies parameterization and granularity as discussed in section 2.4. The random vector $\mathbf{X} = (X_j)_{j=1,\ldots,n}$ is assumed to have a high-dimensional pose density $q_{\mathbf{X}}(\mathbf{x})$ whose samples we denote by $\mathcal{D}_{\mathbf{x}} = \{\mathbf{x}^{(i)}\}_{i=1,\ldots,N}$. In principle, we could use any global density estimation technique to learn $q_{\mathbf{X}}$, but as discussed in section 2.6, such approaches are either prone to overfitting, lack flexibility or are computationally intractable.

In this work, we therefore take another approach and learn a non-parametric Bayesian network $\mathcal{B} = (G, p_{\mathbf{X}})$ with low treewidth. Following section 2.2, we assume that the joint distribution $p_{\mathbf{X}}(\mathbf{x})$ factorizes over the directed acyclic graph $G = (\mathbf{X}, E)$,

$$p_{\mathbf{X}}(\mathbf{x}) = \prod_{j=1}^{n} p_{X_j}(x_j \mid \mathrm{pa}_G(x_j)), \tag{3.1}$$

where, as before, the operator $\mathrm{pa}_G(\bullet)$ maps a node to its parents w.r.t. $G$, [81, 7].[1]

Hence, a complete specification of $\mathcal{B}$ requires the definition of a global topology $G$ and the definition of local models $p_{X_j}(x_j \mid \mathrm{pa}_G(x_j)), 1 \leqslant j \leqslant n$. In the next two sections, we will introduce a fully non-parametric approach for both these elements.

Our model is different from an earlier type of non-parametric Bayesian network [69], which is based on a mixture distribution over all possible networks and therefore cannot compute exact likelihoods efficiently.

---

[1] We sometimes use $p_{\mathbf{X}}$ to refer to a density but note that $p_{\mathbf{X}}(\mathbf{x})$ is not a probability.

### 3.2.1 Learning the Graph Structure

The graph structure of a Bayesian network models the local and global (in)dependencies of a distribution (Eq. 2.7). In most cases, the object to be modeled carries some apparent structure and many approaches define $G$ in a way that reflects the objects intuitive dependencies. In case of the human body, an obvious structure is the kinematic chain, *i.e.*, a tree-structured network with one parent per variable that follows the adjacency of joints in the body (Fig. 3.2a). However, such a canonic representation does not necessarily lead to optimal conditional distributions in an information-theoretic sense.

Therefore, we take another approach and learn $G$ from $\mathcal{D}_{\mathbf{x}}$. Since our goal is to learn a topology with low treewidth, we keep the constraint of at most one parent per variable and search for a Bayesian network $\mathcal{B} = (G, p_{\mathbf{X}})$ with arborescence $G$ and minimal Kullback-Leibler divergence to $q_{\mathbf{X}}(\mathbf{x})$ [81],

$$G := \operatorname*{arg\,min}_{\mathrm{pa}} \mathrm{KL}\left( q_{\mathbf{X}}(\mathbf{x}) \,\middle\|\, \prod_{j=1}^{n} p_{X_j}\big(x_j \mid x_{\mathrm{pa}(j)}\big) \right). \tag{3.2}$$

The network minimizing this distance is known as a Chow-Liu tree and was introduced in [26] for discrete distributions. Given a complete graph $\widetilde{G}$ over $\mathbf{X}$ with edge weights $w(X_j, X_k)$ between $X_j$ and $X_k$ set equal to their mutual information $\mathrm{MI}(X_j, X_k)$,

$$w(X_j, X_k) := \mathrm{MI}(X_j, X_k) = \mathrm{KL}\big(p_{(X_j,X_k)}(x_j, x_k) \,\|\, p_{X_j}(x_j) p_{X_k}(x_k)\big), \tag{3.3}$$

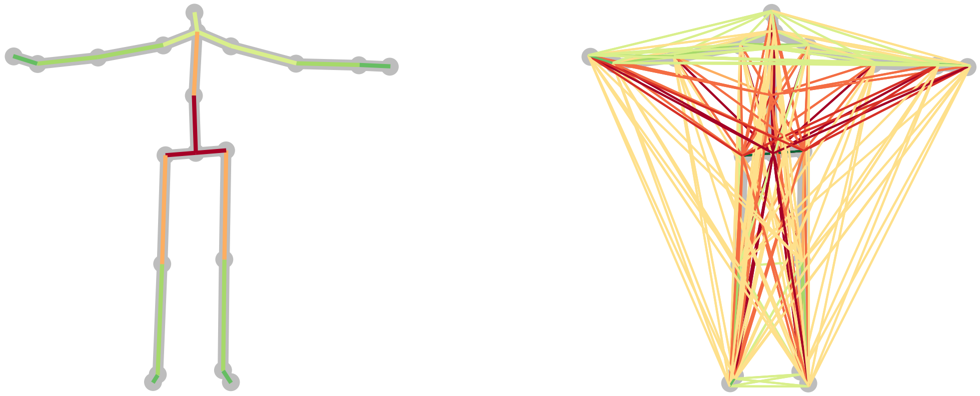
the globally optimal solution to Eq. (3.2) can be shown to be the maximum spanning tree of $\widetilde{G}$ (with edges directed away from an arbitrary root node) [7].[2] In contrast to the kinematic chain, a Chow-Liu tree is thus guaranteed to model those pairs of joints that exhibit a high flow of information, independent of their adjacency in the human body.

Here, we use a continuous variant of the Chow-Liu tree, where reliable estimation of mutual information is a hard problem [167]. An ad-hoc resolution is to either discretize the variables or to make simple parametric assumptions. Instead, we employ a fully non-parametric approach based on nearest neighbor distances: We first use the non-parametric entropy estimator [85] in $d_j := \dim(X_j)$ dimensions and calculate

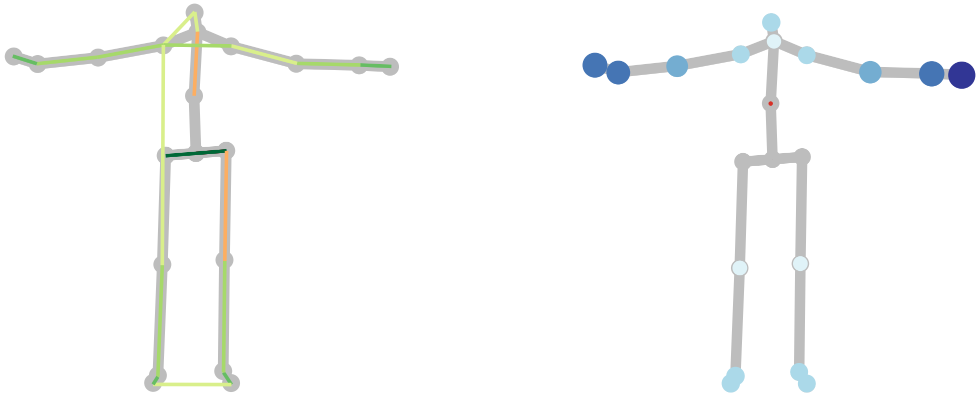
$$\widehat{H}(X_j) := \frac{d_j}{N} \sum_{i=1}^{N} \ln \left\| x_j^{(i)} - \eta_j^{(i)} \right\| + c, \tag{3.4}$$

---

[2] We omit arrows from our network visualizations and implicitly assume all edges to be directed away from the hip node.



**(a)** Kinematic Chain. **(b)** Pairwise Mutual Information.



**(c)** Optimal Topology. **(d)** Entropies.

**Figure 3.2: Optimal Graph Structure. (a)** Graph structure of the commonly used kinematic chain together with non-parametric estimates of its mutual informations (high MI in green, low MI in red). **(b)** Complete graph of all pairwise mutual informations. **(c)** The maximum spanning tree of the graph shown in (b), *i.e.*, the globally optimal Chow-Liu tree. Note how the uninformative edges present in (a) are circumvented. **(d)** Visualization of estimated joint entropies by means of a Hinton diagram.

with the constant $c = \ln(N-1) + \ln V_{d_j} + \gamma$. In the equation above, $\eta_j^{(i)}$ is the nearest neighbor of $x_j^{(i)}$, $V_{d_j} = \pi^{d_j/2} / \, \Gamma(d_j/2+1)$ is the volume of the $d_j$-dimensional unit ball, and $\gamma \approx 0.5772$ is the Euler-Mascheroni constant. A more general class of entropy estimators including the one above was shown to be asymptotically unbiased and consistent as $N \rightarrow \infty$ in [57]. Using the entropy estimates, we can then estimate all pairwise mutual information values by using the relation, [167],

$$\begin{aligned}\widehat{w}(X_j, X_k) :=& \widehat{\mathrm{MI}}(X_j, X_k)\\ =& \widehat{\mathrm{H}}(X_j) + \widehat{\mathrm{H}}(X_k) - \widehat{\mathrm{H}}(X_j, X_k).\end{aligned} \tag{3.5}$$

Finally, we obtain the optimal topology $G$ (Fig. 3.2c) by calculating the maximum spanning tree [28] of the full mutual information graph $\widetilde{G}$ (Fig. 3.2b).

### 3.2.2 Learning the Local Models

Once the network structure $G$ is fixed, we can learn the local densities $p_{X_j}\big(x_j \mid x_{\mathrm{pa}(j)}\big)$ from the training set $\mathcal{D}_{\mathbf{x}}$. We estimate them independently and focus on a specific node $X$ with parent $Y := \mathrm{pa}_G(X)$ to keep the notation uncluttered:

Let $p_X(x \,|\, y)$ be a local model and $\mathcal{D}_{(x,y)} := \{(x^{(i)}, y^{(i)})\}_{i=1,\dots,N}$ observations of the corresponding joint distribution $p_{XY}(x, y)$. Our approach will be to compute the conditional density of a kernel density estimate (CKDE), so that we can use non-parametric KDE's as local models of the Bayesian network $\mathcal{B}$. For two variables $(X, Y)$, the general form of an unconditional kernel density estimate (Eq. (2.10)) reduces to

$$p_{XY}(x, y) := \frac{1}{N|\mathbf{B}|} \sum_{i=1}^{N} \kappa\Big(\mathbf{B}^{-1}\big((x, y) - (x^{(i)}, y^{(i)})\big)\Big). \tag{3.6}$$

We use plug-in estimation as introduced in Eq. (2.12) to estimate $\mathbf{B}$ from the sample covariance and an isotropic Gaussian kernel $\kappa = \mathcal{N}\Big(\vec{\mathbf{0}}, \mathbf{I}\Big)$, in which case Eq. (3.6) simplifies to

$$p_{XY}(x, y) := \frac{1}{N} \sum_{i=1}^{N} \mathcal{N}\big((x, y) \mid (x^{(i)}, y^{(i)}), \mathbf{B}^2\big). \tag{3.6a}$$

The conditional density for a given value $Y = y$ is

$$p_X(x \mid Y = y) = \frac{p_{XY}(x, y)}{\int_x p_{XY}(x, y)\, \mathrm{d}x}, \tag{3.7}$$

where the evidence term in the denominator requires integration over all non-evident dimensions. For a Gaussian kernel, this marginal density has the analytic solution

$$\int_x p_{XY}(x, y)\, \mathrm{d}x = \frac{1}{N} \sum_{i=1}^{N} \mathcal{N}\big(y \mid y^{(i)}, \mathbf{B}^2_{|yy}\big), \tag{3.7a}$$

where $\mathbf{B}^2_{|yy}$ denotes the part of $\mathbf{B}^2$ describing the covariance of $Y$. In summary, we can compute the CKDE $p_X(x \,|\, Y = y)$ efficiently and at the same asymptotic complexity as the joint KDE $p_{XY}(x, y)$.

### 3.2.3 Exact Log-likelihoods and Sampling

There are two important operations to perform in applications of our model as a pose prior: computing the *likelihood* of a given pose and *sampling* a new pose. Both operations are efficient as we now show.

**Exact Log-likelihoods.** Given a Bayesian network with Chow-Liu structure and CKDE models, the log-likelihood $\log p_{\mathbf{X}}(\mathbf{x})$ of a new observation $\mathbf{x} \in \mathbb{R}^d$ is

$$\sum_{j=1}^{n} \left( \log p\big(x_{j,\mathrm{pa}(j)}\big) - \log \int_{x_j} p\big(x_{j,\mathrm{pa}(j)}\big) \, \mathrm{d}x_j \right), \tag{3.8}$$

where we use the shorthand notation $x_{j,k} := (x_j, x_k)$. Both parts of the $j$'th summand have a closed-form solution. Note that the global log-likelihood is composed of many local log-likelihoods, so that we can distinguish likely from unlikely angles/positions on the joint level. This allows a detailed analysis of a pose not possible in global methods.

**Sampling.** Thanks to the closed-form solution for a conditional Gaussian, we can employ standard ancestral sampling [81], *i.e.*, we find a topological ordering $\tau$ for the network structure $G$ and draw samples from $p_{X_{\tau(j)}}(x_{\tau(j)} | x_{\mathrm{pa}(\tau(j))})$, for $j = 1, \ldots, n$. The only technicality we need to take care of is a conditional reweighting operation of the Gaussian components: A standard kernel density estimate in the form of Eq. (3.6a) can be interpreted as a Gaussian mixture with uniform weights and sampling boils down to sampling from a uniformly selected component. In ancestral sampling, on the other hand, we have to deal with conditional distributions. Splitting up the enumerator in Eq. (3.7) shows that we get again a Gaussian mixture model,

$$p_X(x \mid Y = y) = \sum_{i=1}^{N} \alpha_i \cdot \mathcal{N}\Big(x \,\Big|\, \mu_{|y}^{(i)}, \Sigma_{|y}\Big), \tag{3.9}$$

but this time with non-uniform weights,

$$\alpha_i := \frac{\mathcal{N}\Big(y \,\Big|\, y^{(i)}, \mathbf{B}_{|yy}^2\Big)}{\sum_{i=1}^{N} \mathcal{N}\Big(y \,\Big|\, y^{(i)}, \mathbf{B}_{|yy}^2\Big)}. \tag{3.10}$$

Here, $\mu_{|y}^{(i)}$ and $\Sigma_{|y}$ are the mean and covariance of the $i$'th Gaussian conditioned on $y$. Sampling from a local distribution thus consists of two steps: We first select a

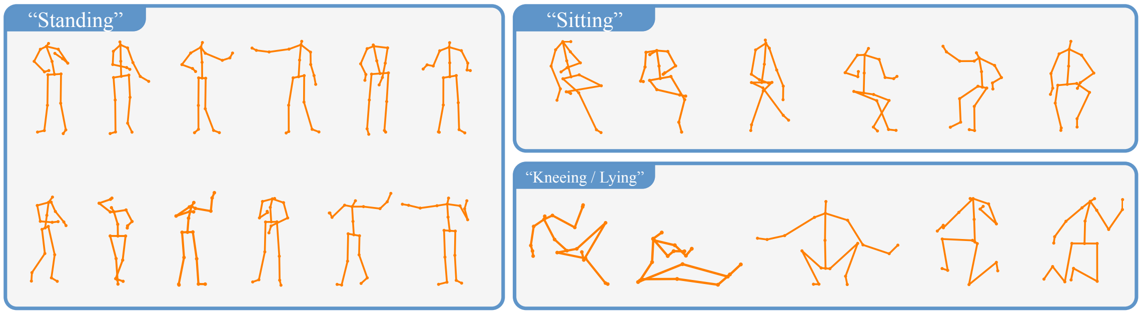


**Figure 3.3: Samples from Human Pose Prior.** Our non-parametric prior of human pose is completely generative and allows for simple and efficient sampling. We show a collection of samples that were generated from a single model with Chow-Liu topology and CKDE models; the assignment to different categories was done manually.

Gaussian component according to the discrete distribution induced by the weights and then draw a sample from the selected Gaussian conditioned on $y$.

We see that, despite its flexibility, computations in our model are efficient, exact, and simple to implement. We now validate our model experimentally.

## 3.3 Experiments

Our experiments are based on two different datasets: the Human 3.6M (H36M) dataset [71] for large-scale experiments and our own Kinect recordings to showcase more specific aspects of our model. For the H36M experiments we use all 7 actors for whom pose data are available and employ a leave-one-person-out scheme: We use all 30 categories of actors $5, 6, 7, 8, 9, 11$ and randomly subsample 15% of all frames without replacement to obtain more independent samples; this constitutes our training set ($\approx 7.0 \cdot 10^4$ poses). We use all frames of actor 1 to construct the test set ($\approx 6.2 \cdot 10^4$ poses). The H36M skeleton includes some spurious joints that we delete, which results in the same 20 joints present in the Kinect skeleton [143] (see also Fig. 2.6a). All frames are given in relative Cartesian coordinates centered at the hip node, unless otherwise stated.

### 3.3.1 Pose Model

We start by learning a pose model on the H36M training set according to the techniques introduced in section 3.2. The resulting network structure is displayed in

Fig. 3.2c and it is worth noting some of its properties: 1. Three edges connect the left half of the body with the right half, thereby enforcing coherent positions for the feet, hips and shoulders. Note that this does not apply to the kinematic chain. 2. The uninformative pairs of nodes present in the kinematic chain (red edges in Fig. 3.2a) are circumvented in the Chow-Liu tree, thus guaranteeing, from an information-theoretic point of view, optimal conditional distributions under the given constraint of a sparse structure. 3. Subgraphs containing joints with high entropies (Fig. 3.2d), such as the arms and legs, largely follow the kinematic chain. This confirms the intuitive belief that joints with high uncertainty should be conditioned on nearby joints, as they provide the maximum information about a joints position in this case.

One of the advantages of a generative model is its ability to produce new hypotheses, *e.g.*, by drawing samples from the model. Using our Matlab implementation of the sampling scheme introduced in section 3.2.3, we are able to generate approximately 10 samples/second. Fig. 3.3 shows a selection of them. Note the variety of poses and their natural appearance, confirming that our model can indeed capture, represent and generate many of the characteristics unique to a human pose.

### 3.3.2 Comparison of Hold-out Log-likelihoods

In this experiment, we evaluate how well our model fits to unseen test poses and how it compares to competing methods. One way to do this in an unsupervised setting is to compute expected hold-out log-likelihoods (ELLs) on the H36M test set. As described in section 3.2, our model consists of two components: estimation of the graph structure (non-parametric Chow-Liu tree) and estimation of the local distributions (conditional kernel density estimation).

We compare this approach to combinations of different topologies/local models and evaluate ELLs on both training and test data. Specifically, we consider two different ways of estimating the local models and six different graph structures. The options for the conditional distributions are our CKDE approach and a Gaussian linear (GL-) network [81]. Cases of badly conditioned covariance matrices are handled by enforcing a lower bound on the eigenvalues.

For the graph structures, we consider a global graph with only a single node, a fully independent graph with $n$ nodes but no edges, the kinematic chain, a higher-order kinematic chain, and two variants of the Chow-Liu tree, one with parametric and one with non-parametric estimation of mutual information. In the higher-order kinematic chain each joint is additionally conditioned on its parents' parents. Parametric MI-estimation is based on the entropy of fitted Gaussians. We use parametric

**Table 3.1: Expected Log-likelihoods.** Results for GL- and CKDE-networks with different graph structures and a comparison to global methods.

| Method | Graph structure | Training | Testing |
|---|---|---|---|
| Gaussian | Global | −266.84 | −271.15 |
| KDE | Global | −239.61 | −263.77 |
| GPLVM* | Global | −327.85 | −341.89 |
| GL-network | Independent | −352.80 | −345.94 |
| | Kinematic chain (order 1) | −311.54 | −310.98 |
| | Kinematic chain (order 2) | −305.54 | −307.88 |
| | Chow-Liu tree | −283.82 | −284.03 |
| CKDE-network | Independent | −322.64 | −322.25 |
| | Kinematic chain (order 1) | −260.04 | −270.52 |
| | Kinematic chain (order 2) | −247.35 | −263.83 |
| | Chow-Liu tree **(ours)** | −242.24 | **−254.98** |

*25% subsampling; FITC

MI-estimation for the parametric GL-network and non-parametric MI-estimation (Eqs. 3.4–3.5) for the non-parametric CKDE-network.

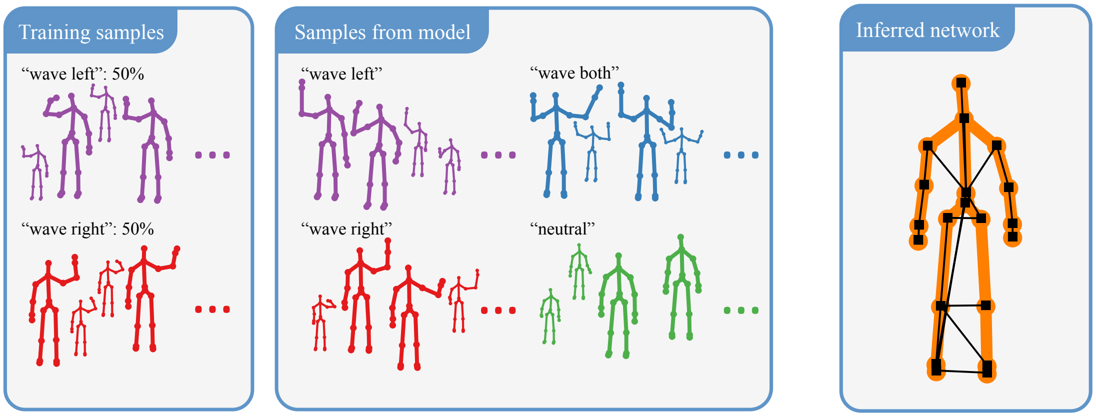
The network approaches are complemented by a comparison to the global GPLVM [95], where we employ the popular FITC approximation [148] together with subsampling to achieve tractability. We use a reference implementation[3] and consider embeddings in 1, 3, and 5 latent dimensions, reporting the best ELL.

**Results.** Our results are shown in Table 3.1. Among the global methods, the test ELL of a KDE (−264) outperforms both a global Gaussian (−271) and a GPLVM (−342; 5 latent dimensions), despite a spread between training and testing due to overfitting. Although the GPLVM performance could probably be further improved, either by developing better approximations or by fine-tuning the parameters, the application of a GPLVM to such a large dataset is an inherently approximate procedure involving a non-convex optimization problem prone to initialization and local minima, which is presumably the cause for its poor performance.

Let us now turn to the network approaches and analyze their graph structures. Not surprisingly, a network modeling the joints independently performs worst, with test ELLs of −346 (GL) and −322 (CKDE). Using graph structures based on the kinematic chain increases the test performance to −311 (GL) and −271 (CKDE). Higher-order kinematic chains improve on the results by another +6.7 (CKDE) and +3.1 (GL) nats. The best graph structure in this comparison are Chow-Liu trees.

---

[3] `http://staffwww.dcs.sheffield.ac.uk/people/N.Lawrence/fgplvm/`



**(a)** Pose Classes during Training and Testing. **(b)** Topology.

**Figure 3.4: Compositionality. (a)** Samples from the "wave" training set (left, 2 pose classes) and samples drawn from the learned model (right, 4 pose classes). Our model has learned two pose classes not available in the training set: "wave both" and "neutral". **(b)** The inferred network structure. Note that the arms are conditionally independent and can thus be freely combined, whereas the legs are dependent on each other.

Their usage results in a big leap in performance, increasing the results again by +8.9 nats in case of the CKDE-network and +23.9 nats in case of the GL-network. The direct comparison of CKDE- to GL-networks is unambiguous: CKDE networks perform consistently better, independent of the graph structure. The combination of a Chow-Liu topology and CKDE models also performs better than all 3 global methods, making it the best performing approach in this comparison.

### 3.3.3 Compositionality

One of the major disadvantages of global non-parametric models is their susceptibility to overfitting; they basically represent and reproduce the training samples. On the other hand, parametric networks based on the kinematic chain are too flexible in the sense that they allow arbitrary combinations of the positions of different limbs. This is because different limbs are conditionally independent once their lowest common ancestor w.r.t. the hierarchical tree structure is observed. This is the case, for example, when performing ancestral sampling. At the same time, Gaussian linear networks are not flexible enough in the sense that their local distributions cannot cope with multimodality, which is essential when modeling human pose.

Ideally, we would like to have flexibility and compositionality only where it is adequate and needed. In order to check this property under controlled conditions, we record two different gestures in front of a Kinect: either waving with the left hand only or waving with the right hand only (1000 frames each; Figure 3.4a (left)). We then learn a pose model according to section 3.2, draw 5000 samples from it and cluster them into 4 clusters using $k$-means.

**Results.** The non-parametric Chow-Liu tree of the model is shown in Fig. 3.4b. Since the arms do not share much information in our training sequences, they are automatically modeled conditionally independent of each other, *i.e.*, we can freely combine their positions. In contrast to that, the position of the right leg tells us a lot about the position of the left leg, since both are parallel to one another throughout the sequences. Consequently, the joint positions of the latter are all modeled conditional on the corresponding joint positions of the former.

The samples generated by this model (Fig. 3.4a (right)) fall into 4 distinct pose classes. Two of the four clusters (coloured in purple and red) correspond to poses also present in the training set. The other two clusters (coloured in blue and green) represent newly learned poses that do not appear in the training data: a neutral pose (both hands lowered) and a pose with both hands raised. The key here is that we do have samples with the left and right hand raised, just never in the same frame. During the sampling process, our model combines the available data to form a new sample that possibly does not resemble any training sample.

In summary, our formulation allows to freely combine substructures, but only if they do not share a lot of information. Joint positions that heavily depend on each other, for instance due to physical constraints, will always be modeled conditional on each other. Examples include positions of the feet (gravity) and the hips (rigidity), *i.e.*, we get compositionality exactly where we need it.

## 3.4 Real-time Scoring

Time is a critical factor in applications such as tracking or pose estimation. Our presentation in section 3.2 has shown that the computation of exact log-likelihoods in our model is tractable, *i.e.*, we do not have to resort to MCMC or similar methods. However, for large datasets the number $N$ of terms in the summations (3.6a) and (3.7a) will also become large, resulting in longer runtime. We can speed up inference by considering approximate log-likelihoods. There exist many fast methods for the evaluation of kernel density estimates, but the popular approaches are either not suited for high-dimensional data [37], do not lead to a speed-up for sequential

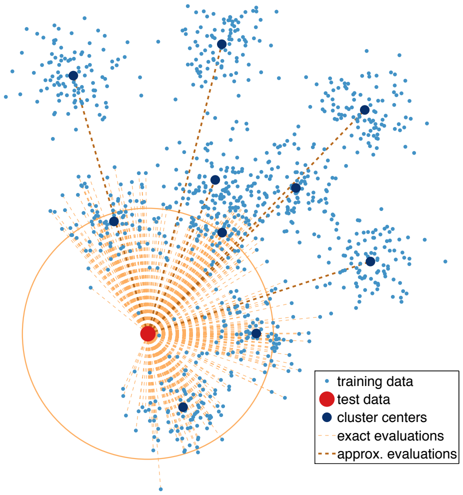
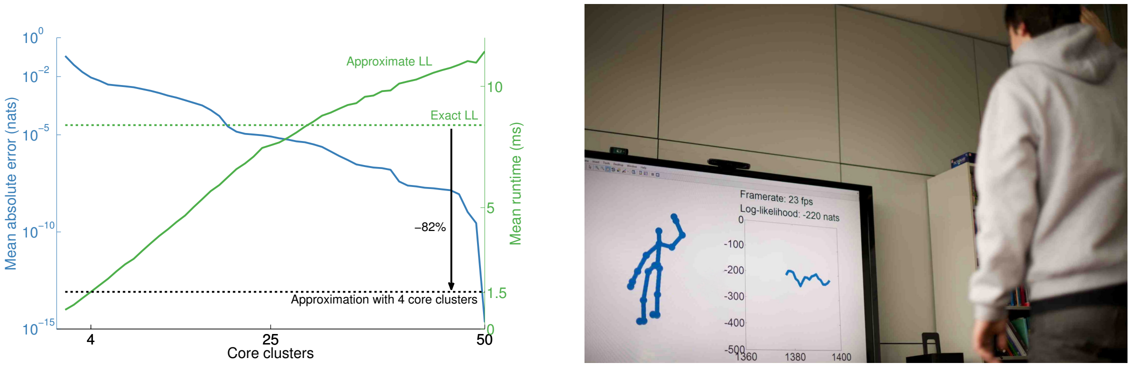


**Figure 3.5: Approximate Log-likelihoods.** Training points (light blue) that belong to a cluster whose center (dark blue) is close to the test point (red) are evaluated individually. Those farther away are approximated by their respective cluster centers.

data [172] or are hard to implement due to their complexity [58]. Here, we want to propose a simple alternative to these complex methods that will be sufficient for our purpose. Our experiments will show that the additional decrease in runtime is sufficient to allow the application of our approach in real-time, without having to sacrifice accuracy.

At training time, we cluster all training points into clusters $C_1, \dots, C_k$ using $k$-means and build a kd-tree for the cluster centers. At test time, we partition the clusters into a set of core clusters $\mathbf{C}^e$ and a set of approximate clusters $\mathbf{C}^a$ based on the following scheme: Given a test point $\mathbf{x}$, we use the kd-tree to determine the $k' \ll k$ clusters whose centers lie closest to $\mathbf{x}$. These make up the core clusters. All remaining clusters are considered approximate clusters. We then evaluate all training points within the core clusters exactly. All the other clusters are evaluated by multiplying the log-likelihood w.r.t. the center by the size of the cluster (Fig. 3.5). The sum in Eq. (3.6) thus decomposes into an exact and an approximate sum,

$$p_{\mathbf{X}}(\mathbf{x}) = \frac{S_e + S_a}{N|\mathbf{B}|}, \tag{3.11}$$



**(a)** Accuracy vs. Runtime. **(b)** Live Scoring in Front of a Kinect.

**Figure 3.6: Approximation Trade-off and Real-time Inference.** **(a)** Mean accuracy and mean runtime of approximate local log-likelihoods as a function of the number of core clusters. For comparison we also include the runtime for the computation of exact local log-likelihoods. The black line illustrates the situation for a tolerable $\epsilon$-error of $10^{-2}$, corresponding to 4 core clusters. The reduction in computational cost of 82% is big enough to allow real-time applications, shown in **(b)**.

with

$$S_e = \sum_{C \in \mathbf{C}_e} \sum_{j \in C} \kappa\big(\mathbf{B}^{-1}\big(\mathbf{x} - \mathbf{x}^{(j)}\big)\big), \tag{3.12}$$

$$S_a = \sum_{C \in \mathbf{C}_a} |C| \kappa\big(\mathbf{B}^{-1}\big(\mathbf{x} - \overline{C}\big)\big), \tag{3.13}$$

where $\overline{C}$ and $|C|$ denote the center and size of cluster $C$, respectively. In this formulation, those training points contributing most to the log-likelihood are evaluated exactly and those farther away are approximated by their corresponding cluster centers. As the number of core clusters approaches the total number of clusters (or as the number of total clusters approaches the total number of training points), our approximate method converges to the exact log-likelihood.

Since the contribution of a training point to the log-likelihood decreases exponentially with its distance from the test point, a few core clusters should suffice to achieve a high level of accuracy. In order to prove this point, we cluster the entire Human 3.6M training set into 50 clusters, evaluate approximate log-likelihoods for 100 randomly sampled points from the Human 3.6M test set and compare them to their exact counterparts. Fig. 3.6a shows the results in terms of accuracy and speed for a local log-likelihood: If an absolute error of $10^{-2}$ nats is acceptable, we need as few as 4 core clusters and the runtime is 1.5ms per frame. This compares to 8.4ms for the computation of an exact local log-likelihood. Adding more core clusters further decreases the error, while the runtime increases sublinearly. As the evaluation of a log-likelihood for a Bayesian network in our case requires computation of $2n = 40$ local log-likelihoods (see Eq. (3.8)), we achieve a total speed of approx. 61ms per frame (16 fps) on a dataset containing about $70,000$ training points.

This work is accompanied by an open source Matlab suite for Kinect data. Our framework supports training and real-time scoring of motion capture and Kinect poses (Fig. 3.6b), making the integration of the proposed model as part of a larger pipeline very easy.

# Chapter 4

# Non-parametric Markov Models for Human Motion

Tree-structured non-parametric Bayesian networks are good priors for static tasks, *e.g.*, human pose estimation in images. However, they are not well-suited to assess the likelihood of a time series, such as Kinect or motion capture data, because they lack temporal coherence and consistency. Such dynamic context is typically modeled using the hidden state sequence of a state-space model. An efficient and flexible alternative to these traditional models is a non-parametric Markov model without explicit latent space but the possibility to choose from different conditional models in a deterministic way.

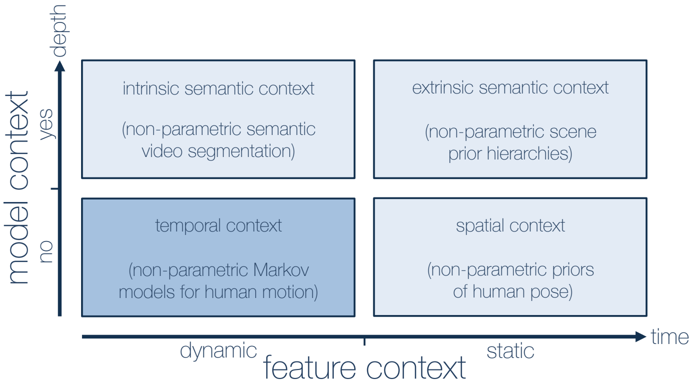

model context
depth
yes
no
intrinsic semantic context
(non-parametric semantic video segmentation)
extrinsic semantic context
(non-parametric scene prior hierarchies)
temporal context
(non-parametric Markov models for human motion)
spatial context
(non-parametric priors of human pose)
time
dynamic
static
feature context

**Figure 4.1: Key Message: Dynamic Context.** A temporal extension of decision forests can be used as a non-parametric $\delta$-approximation of a state-space representation like a hidden Markov model to encode dynamic dependencies in human motion data.

## 4.1 Introduction

Statistical models for human motion are important in many areas of computer vision and graphics. In addition to being interesting in their own right [164, 153, 115], their applications include areas as diverse as animation, robotics, tracking [144, 159], and activity recognition [13]. While great progress has been made in the past decade, the problem remains challenging because of the high dimensionality, nonlinearity and multimodality of natural human motion. Ideally, a good probabilistic model should account for all of those challenges, but unfortunately expressive models often result in intractable estimation and inference problems (see chapter 2). We now review the most popular approaches.

### 4.1.1 Dynamic Bayesian Networks

In recent years, modelling of human motion has been tackled from several different perspectives. Among the most popular methods are time-homogeneous latent variable models following the state-space equations (see Fig. 4.3a),

$$\begin{aligned} \mathbf{z}^{(t)} &= f\big(\mathbf{z}^{(t-1)}, \boldsymbol{\epsilon}_{\mathbf{z}}\big) \\ \mathbf{x}^{(t)} &= g\big(\mathbf{z}^{(t)}, \boldsymbol{\epsilon}_{\mathbf{x}}\big) \end{aligned} \qquad \longleftrightarrow \qquad \begin{aligned} &p_{\mathbf{Z}}\big(\mathbf{z}^{(t)} \,\big|\, \mathbf{z}^{(t-1)}\big), \\ &p_{\mathbf{X}}\big(\mathbf{x}^{(t)} \,\big|\, \mathbf{z}^{(t)}\big), \end{aligned} \tag{4.1}$$

where $\mathbf{x}^{(t)}$ are observable variables, such as joint positions or joint angles, $\mathbf{z}^{(t)}$ are hidden variables, and $\boldsymbol{\epsilon}_{\mathbf{x}}, \boldsymbol{\epsilon}_{\mathbf{z}}$ are random perturbations. Although filtering and smoothing distributions for $\mathbf{z}^{(t)}$ are within this framework, they are only tractable for either a discrete state space (forward(-backward) algorithm) or linear functions $f, g$ and additive Gaussian noise (Kalman filter/smoother) [47]. Efficient and exact solutions for the general nonlinear and/or non-Gaussian cases do not exist [21]. In order to perform inference, one therefore needs to resort to approximate methods, such as the extended Kalman filter and its derivates [77, 73], or to sequential Monte Carlo (SMC) methods like particle filters [56, 21, 20]. Augmenting the state-space model with discrete switching variables leads to a switching linear dynamical system (SLDS). Exact inference in this model is intractable even for the Gaussian-linear case [7] and learning has turned out to be a challenge in itself [46].

Eq. (4.1) and its extensions form the basis for a number of statistical models for human motion: Wang *et al.* [164] assume a linear combination of nonlinear basis functions for $f, g$ and additive Gaussian noise for $\boldsymbol{\epsilon}_{\mathbf{x}}$, $\boldsymbol{\epsilon}_{\mathbf{z}}$. Marginalization over $f, g$ leads to a Gaussian process dynamical model (GPDM), whose latent trajectories have to be learned by a combination of scaled conjugate gradient and a version of

EM using hybrid Monte Carlo techniques. Urtasun *et al.* [160] extend the GPDM framework by introducing a prior for latent positions that preserves local topological structure. Pavlović *et al.* [115] use approximate variational inference to learn an SLDS and perform inference in it. Taylor *et al.* [153] consider a latent space consisting of binary variables and use a conditional RBM to model human motion. While inference tasks are easy in this model, learning relies on approximations like contrastive divergence [62]. Models inspired by physics and biology include [165] and [166].

### 4.1.2 Contributions

In this work, we propose to leave out the latent space altogether. Instead, we follow the principles introduced in chapter 2 and model the dynamic context in human motion data by means of an expressive Markov model that is both simple enough to allow for efficient and exact inference yet flexible enough to accurately model real human motion data. Due to our non-parametric representation, we achieve a high level of realism. Specifically, our work makes the following four contributions: 1. We introduce dynamic forest models (DFMs) and describe their training and regularization, building upon work on autoregressive trees [103]; 2. We present a formulation of our approach in terms of latent variables, thereby allowing a direct comparison to hidden Markov models (HMMs); 3. We show how DFMs can be used as accurate and efficient models of human motion data; 4. We empirically validate DFMs on challenging action recognition and motion completion tasks, outperforming both HMMs and GPDMs.

## 4.2 Nonlinear Markov Models

In this section we present our model and discuss its training. A *Markov model* describes a conditional distribution of the present state $\mathbf{x}^{(t)}$ given a limited number of past observations $\mathrm{pa}(\mathbf{x}^{(t)}) := \mathbf{x}^{(t-K):(t-1)}$, *i.e.*, at each time step $t$ a fixed number of $K$ previous observations are combined to form a prediction $\mathbf{x}^{(t)}$. The prediction is then an order-$K$ Markov process,

$$p\big(\mathbf{x}^{(t)} \,\big|\, \mathrm{pa}\big(\mathbf{x}^{(t)}\big)\big) = p\big(\mathbf{x}^{(t)} \,\big|\, \mathbf{x}^{(t-K):(t-1)}\big). \tag{4.2}$$

If the mean of this distribution can be written as a fixed linear combination of the previous observations, the Markov model is said to be linear [39]. When this is not the case, the model is a *nonlinear* Markov model.

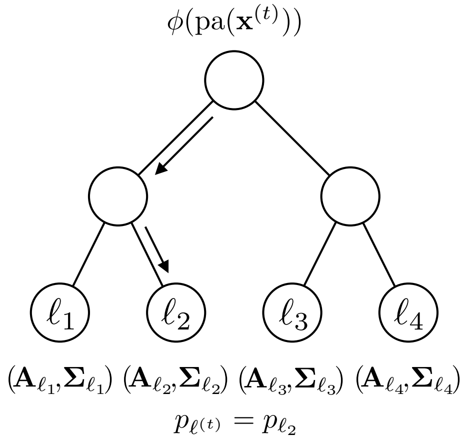

$\phi(\mathrm{pa}(\mathbf{x}^{(t)}))$
$\ell_1$
$\ell_2$
$\ell_3$
$\ell_4$
$(\mathbf{A}_{\ell_1},\boldsymbol{\Sigma}_{\ell_1})$
$(\mathbf{A}_{\ell_2},\boldsymbol{\Sigma}_{\ell_2})$
$(\mathbf{A}_{\ell_3},\boldsymbol{\Sigma}_{\ell_3})$
$(\mathbf{A}_{\ell_4},\boldsymbol{\Sigma}_{\ell_4})$
$p_{\ell^{(t)}} = p_{\ell_2}$

**Figure 4.2: Autoregressive Tree.** A decision tree $\mathcal{T} = (V_I \cup V_L, E)$ is evaluated on a set of features extracted from $K$ previously observed frames $\phi\big(\mathrm{pa}\big(\mathbf{x}^{(t)}\big)\big)$. At each leaf $\ell \in V_L$ of the tree a linear autoregressive model $(\mathbf{A}_\ell, \boldsymbol{\Sigma}_\ell)$ is stored and the predictive filtering distribution for $\ell^{(t)} := \ell(\phi(\mathrm{pa}(\mathbf{x}^{(t)})))$ is defined as $p_{\ell^{(t)}}\big(\mathbf{x}^{(t)} \mid \mathrm{pa}\big(\mathbf{x}^{(t)}\big)\big) = \mathcal{N}(\mathbf{A}_{\ell^{(t)}}\phi\big(\mathrm{pa}\big(\mathbf{x}^{(t)}\big)\big), \boldsymbol{\Sigma}_{\ell^{(t)}})$.

### 4.2.1 Autoregressive Trees

Autoregressive (AR-) trees [103] are a type of probabilistic AR-model for time-series data [39] in which the regression function is given by a regression tree $\mathcal{T} = (V_I \cup V_L, E)$ (see section 2.3.2).

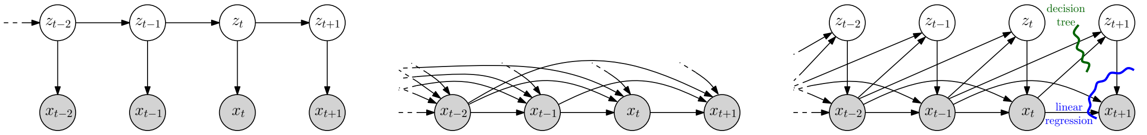
In order to represent the distribution in Eq. (4.2), the regression tree evaluates features $\phi\big(\mathrm{pa}\big(\mathbf{x}^{(t)}\big)\big) \in \Phi$ extracted from the previous $K$ frames and, using this information, decides among a set of simpler distributions stored at its leaf nodes. We store in each leaf $\ell$ one multivariate normal distribution with linearly regressed mean but fixed covariance matrix. This is illustrated in Fig. 4.2. In this case, the predictive expectation can be obtained by the linear dynamics

$$\mathbb{E}\big[\mathbf{X}^{(t)} \mid \mathrm{pa}\big(\mathbf{x}^{(t)}\big)\big] = \mathbf{A}_{\ell(\phi(\mathrm{pa}(\mathbf{x}^{(t)})))}\phi\big(\mathrm{pa}\big(\mathbf{x}^{(t)}\big)\big). \tag{4.3}$$

Although this is a simple linear prediction, the selection of a matrix $\mathbf{A}_{\ell(\bullet)}$ is governed by the decision tree's leaf function $\ell(\bullet) : \Phi \to V_L$ and thus a function of $\phi\big(\mathrm{pa}\big(\mathbf{x}^{(t)}\big)\big)$.[1]Autoregressive trees are therefore not only nonlinear but also nonparametric Markov models. For instance, by testing for statistics such as average joint velocities in the past $K$ frames, the tree may easily distinguish a running from a walking motion, and hence is able to select the appropriate linear dynamics.

---

[1] In general one can use different features for linear prediction and leaf node selection.



**(a)** Hidden Markov Model. **(b)** Marginalized Hidden Markov Model. **(c)** Latent Space View of Dynamic Forest.

**Figure 4.3: Dynamic Forest Model. (a)** Discrete-time hidden Markov models represent a probability distribution of a sequence of observations $(\mathbf{x}^{(t)})_t$ by a Markov model on a sequence of hidden variables $(\mathbf{z}^{(t)})_t$ and a conditional observation distribution $p\big(\mathbf{x}^{(t)} \mid \mathbf{z}^{(t)}\big)$. **(b)** Marginalization over the hidden variables yields a joint distribution $p(\mathbf{x}^{(\bullet)})$ over the observed variables, effectively coupling all variables. **(c)** Latent space formulation of our proposed non-parametric Markov model for order $K = 2$: A decision tree implicitly selects a latent state. We can view this non-parametric Markov model as an order-$K$ approximation to (b) in which filtering inference and computation of marginal observation likelihoods is very efficient.

In the original work on autoregressive trees, a single tree is learned by greedily optimizing a penalized likelihood objective [103]. The authors show applications to short-term forecasting of univariate economic data but note that 95% of their trees do not contain any splits, *i.e.*, they are common AR-models. Here, we propose extensions to AR-trees that enable us to take advantage of deeper trees and to cope with high-dimensional inputs, eventually allowing their use for classification, synthesis and upscaling of complex human motion data.

### 4.2.2 Dynamic Forests

Because a single tree is prone to overfitting and limited in its expressiveness due to its unimodal predictive distribution, we will instead learn an ensemble of $C > 1$ trees, $\{\mathcal{T}_c\}_{c=1,\dots,C}$. Each tree is trained separately using bagging [15], that is, resampling the training set framewise with replacement. The predictions of the individual trees are then *averaged* to produce the prediction of the forest. Since each tree has a Gaussian posterior, the forest posterior is given by a multimodal mixture of Gaussians,

$$p\big(\mathbf{x}^{(t)} \mid \mathrm{pa}\big(\mathbf{x}^{(t)}\big)\big) = \frac{1}{C}\sum_{c=1}^{C} \mathcal{N}\big(\mathbf{x}^{(t)} \mid \boldsymbol{\mu}_{\ell(t,c)}, \boldsymbol{\Sigma}_{\ell(t,c)}\big). \tag{4.4}$$

Here, $\ell^{(t,c)} := \ell^{(c)}(\phi(\mathrm{pa}(\mathbf{x}^{(t)})))$ denotes the leaf node that is selected by the $c$'th tree at the $t$'th time step and each mixture component has the mean vector $\boldsymbol{\mu}_{\ell(t,c)} := \mathbf{A}_{\ell(t,c)}\,\phi(\mathrm{pa}(\mathbf{x}^{(t)}))$. We call this new approach *dynamic forest model (DFM)* and continue with a description of its training.

### 4.2.3 Training

We provide a bottom-up description of the training procedure, *i.e.*, we start with the estimation of a leaf model and then consider the task of learning the tree structure.

The general setup is as follows: At training time, we are given a set of $M$ training sequences $\mathcal{D}_{\text{raw}} = \left(\mathbf{x}^{(\bullet,i)}\right)_{i=1,\dots,M}$. Each training sequence $\mathbf{x}^{(\bullet,i)}$ is a concatenation of $T_i$ frames, hence $\mathbf{x}^{(\bullet,i)} := \left(\mathbf{x}^{(t,i)}\right)_{t=1,\dots,T_i}$. The $t$'th frame in the $i$'th sequence is represented by a fixed-length vector $\mathbf{x}^{(t,i)} \in \mathbb{R}^d$. In contrast to chapter 3, we are now dealing with a supervised task, *i.e.*, we need to define a supervised training set $(\mathcal{D}_{\mathbf{x}}, \mathcal{D}_{\mathbf{y}})$ based on the raw data $\mathcal{D}_{\text{raw}}$. To this end, we set $\mathcal{D}_{\mathbf{x}} = \left(\phi(\text{pa}(\mathbf{x}^{(t,i)}))\right)_{\substack{K+1 \leqslant t \leqslant T_i \\ 1 \leqslant i \leqslant M}}$ and $\mathcal{D}_{\mathbf{y}} = \left(\mathbf{x}^{(t,i)}\right)_{\substack{K+1 \leqslant t \leqslant T_i \\ 1 \leqslant i \leqslant M}}$. We will often drop the sequence index $i$ to keep the notation uncluttered.

#### Training the Leaf Model

Learning a model for a leaf node $\ell$ amounts to estimating a regression matrix $\mathbf{A}_\ell$ and a covariance matrix $\mathbf{\Sigma}_\ell$. Each leaf accommodates a subset $(\mathcal{D}^\ell_{\mathbf{x}}, \mathcal{D}^\ell_{\mathbf{y}})$ of the training data, namely those feature vectors and regression targets that reach it. We use the available data points to estimate $\mathbf{A}_\ell$ using ridge regression [63]. To this end, let $\boldsymbol{\phi}$ denote all column-wise concatenated feature vectors $\phi(\text{pa}(\mathbf{x}^{(t)}))$ that are assigned to the leaf. Likewise, and in the same order, we concatenate all the desired predictions column-wise into a matrix $\mathbf{U}$. The matrix $\mathbf{A}_\ell$ now has the closed-form solution

$$\mathbf{A}_\ell := \mathbf{U}\boldsymbol{\phi}^\top \left(\boldsymbol{\phi}\boldsymbol{\phi}^\top + \gamma \mathbf{I}\right)^{-1}, \tag{4.5}$$

where $\gamma > 0$ is the ridge regularization parameter.

To determine the covariance matrix $\mathbf{\Sigma}_\ell$, we first use the ground truth to compute the residual matrix $\mathbf{R}_\ell := \mathbf{U} - \mathbf{A}_\ell \boldsymbol{\phi}$. The set of all the residual vectors is then used to estimate a matrix $\widehat{\mathbf{\Sigma}}_\ell$ by means of the sample covariance. While the estimate of $\mathbf{A}_\ell$ is generally quite accurate, we observed that the covariance estimate $\widehat{\mathbf{\Sigma}}_\ell$ may become inaccurate for high dimensions and small sample sizes. In some cases the estimated matrices are even singular. We therefore regularize our initial estimate by projecting $\widehat{\mathbf{\Sigma}}_\ell$ to an isotropic target with full rank,

$$\mathbf{\Sigma}_\ell := d^{-1} \operatorname{tr}\left(\widehat{\mathbf{\Sigma}}_\ell\right) \mathbf{I}, \tag{4.6}$$

a measure that proved to be important for the success of our approach. We also experimented with a convex combination of the sample covariance matrix and a diagonal shrinkage target [136], but it did not improve the results.

### Training the Tree Structure

To determine the tree structure, we use a greedy training procedure, as is commonly used in the literature [16, 29]. We start with a single node and recursively split leaf nodes by selecting the best among a set of hyperplane splits (see section 2.3.2). Each candidate split is sampled from a proposal distribution $p_s$, *e.g.*, by uniformly sampling a data point and a coordinate. A candidate split $s$ at leaf $\ell$ introduces child nodes $u$ and $v$, each of which receives a subset of the data present at $\ell$. After estimating a leaf model for $u$ and $v$ according to section 4.2.3, we determine the quality of the proposed split by measuring the resulting reduction in residual error,

$$Z_s := E_\ell - (E_u + E_v), \tag{4.7}$$

where $E_\bullet$ is given as the squared Frobenius norm of the residual matrix,

$$E_\bullet := \|\mathbf{R}_\bullet\|_F^2. \tag{4.8}$$

The split that achieves the largest score $Z_s$ is selected and the training proceeds recursively.

Implementation of DFMs is easy and analytical solutions for the score function and the least squares regressor make learning very efficient. The required training time can be controlled by the number of trees, their depth and the number of tested splits. Algorithm 1 summarizes DFM training.

## 4.3 Latent Space View

In this section, we compare the class of dynamic forest models with the class of hidden Markov models. In order to facilitate a direct comparison, we reformulate DFMs as (pseudo) latent variable models and draw connections between leaf nodes in a DFM and latent states in an LVM.

Consider a tree $\mathcal{T}_c$ with $l$ leaf nodes. By introducing latent variables $z^{(t)}$ with $l$ states we arrive at the latent variable model depicted in Fig. 4.3c for order $K = 2$. In this formulation, the joint distribution of an observed sequence $\mathbf{x}^{(\bullet)} := (\mathbf{x}^{(t)})_{t=1,\dots,T}$ and their corresponding latent variables $z^{(\bullet)} := (z^{(t)})_{t=1,\dots,T}$ is given by

$$p\big(\mathbf{x}^{(\bullet)}, z^{(\bullet)}\big) = \prod_{t=K+1}^{T} p\big(\mathbf{x}^{(t)} \mid \mathrm{pa}\big(\mathbf{x}^{(t)}\big)\big) \cdot p\big(z^{(t)} \mid \mathrm{pa}\big(z^{(t)}\big)\big). \tag{4.9}$$

Identifying latent variable states with decision tree leaves, the distribution of $z^{(t)}$ is a deterministic prediction from previous observations and can thus be understood as

**Algorithm 1** Probabilistic DFM Training

```
input: time-series data $(\mathcal{D}_{\mathbf{x}}, \mathcal{D}_{\mathbf{y}})$
input: number of ensemble trees $C > 1$
input: number of split tests $M \geqslant 1$
input: split proposal distribution $p_s$
output: dynamic forest $\{\mathcal{T}_c\}_{c=1,\ldots,C}$
procedure TrainDFM($(\mathcal{D}_{\mathbf{x}}, \mathcal{D}_{\mathbf{y}}), C, M, p_s$)
    for $c = 1, \ldots, C$ do
        Bootstrap resample the supervised training set $(\mathcal{D}_{\mathbf{x}}, \mathcal{D}_{\mathbf{y}})$
        $\mathcal{T}_c \leftarrow$ a growable root node $\rho$
        while there is a growable leaf $\ell$ in $\mathcal{T}_c$ do
            $Z^* \leftarrow -\infty$
            for $m = 1, \ldots, M$ do
                Sample split $s \sim p_s$
                Partition data at $\ell$ w.r.t. $s$
                $Z_s \leftarrow$ Compute score for $s$
                if $Z_s > Z^*$ then
                    $(s^*, Z^*) \leftarrow (s, Z_s)$
                end if
            end for
            Split leaf $\ell$ using split $s^*$
        end while
        for leaf $\ell$ in tree $\mathcal{T}_c$ do
            $(\mathbf{A}_\ell, \mathbf{\Sigma}_\ell) \leftarrow$ Build least squares regressor at $\ell$
        end for
    end for
    return Ensemble $\{\mathcal{T}_c\}_{c=1,\ldots,C}$
end procedure
```

a delta function,

$$p\big(z^{(t)} \mid \mathrm{pa}\big(z^{(t)}\big)\big) = p\big(z^{(t)} \mid \mathrm{pa}\big(\mathbf{x}^{(t)}\big)\big) = \delta\big(z^{(t)} - \ell^{(t,c)}\big). \tag{4.10}$$

In DFMs the information between the latent variables flows through the observed variables, whereas in HMMs the latent variables have direct interactions. Because the latent states are determined from observations only, they are conditionally independent given the observed sequence. This is different to hidden Markov models. More precisely, DFMs encode the following conditional independence statements for all $t > K + 1$:

$$\begin{aligned} Z^{(t)} &\perp\!\!\!\perp \big\{Z^{(1)}, \dots, Z^{(t-1)}\big\} \mid \mathrm{pa}\big(Z^{(t)}\big), \\ \mathbf{X}^{(t)} &\perp\!\!\!\perp \big\{\mathbf{X}^{(1)}, \dots, \mathbf{X}^{(t-K-1)}\big\} \mid \mathrm{pa}\big(\mathbf{X}^{(t)}\big). \end{aligned} \tag{4.11}$$

It is this factorization that allows us to compute the marginal observation likelihood in a single summation,

$$\begin{aligned} \log p\big(\mathbf{x}^{(\bullet)}\big) &= \log \sum_{z^{(K+1)}} \cdots \sum_{z^{(T)}} p\big(\mathbf{x}^{(\bullet)}, z^{(\bullet)}\big) \\ &= \sum_{t=K+1}^{T} \log p\big(\mathbf{x}^{(t)} \mid \mathbf{x}^{(t-K)}, \dots, \mathbf{x}^{(t-1)}, \ell^{(t,c)}\big). \end{aligned} \tag{4.12}$$

Under the assumption of balanced trees, the cost for the computation of log-likelihoods in DFMs is $\mathcal{O}(\log(l)T)$, which is sublinear in the number of latent states. The time complexity of hidden Markov models for the same task scales according to $\mathcal{O}(l^2T)$, which is quadratic.

The additional efficiency in our model relies on two implicit assumptions: 1. We assume that we can identify the correct latent state. At time $t$, we put the entire probability mass on a single latent state that we select based on the feature vector $\phi\big(\mathrm{pa}\big(\mathbf{x}^{(t)}\big)\big)$. Our approach thus stands and falls with the design of this feature vector and the information it encodes. A hidden Markov model, on the other hand, can incorporate prior knowledge from the application domain (*e.g.*, occlusion reasoning, object-object interactions, or compositionality [12]) by refining the model used for the hidden state sequence; 2. We assume that long-range dependencies are negligible. Whereas we do not model interactions beyond order $K$, hidden Markov models do have a long-term memory due to the Markov process on the hidden state sequence, thereby rendering the observation sequence non-Markov ([47], section 1.3.3).

## 4.4 Experiments

We demonstrate the usefulness of DFMs on three different tasks: action recognition, motion completion and prediction of 3D motion from 2D inputs. Our experiments are based on the MSRC-12 dataset [45] and a modified version of the CMU dataset[2] as used by Wang *et al.* [164].

As before, we represent a human motion sequence of length $T$ as a temporal sequence of $d$-dimensional body poses. This yields a matrix $\mathbf{x}^{(\bullet)} = \big(\mathbf{x}^{(t)}\big)_t \in \mathbb{R}^{d \times T}$

---

[2] Dataset obtained from `mocap.cs.cmu.edu`.

which can be used as part of an ensemble training according to section 4.2. Depending on the dataset, the individual poses $\mathbf{x}^{(t)}$ in the columns of $\mathbf{x}^{(\bullet)}$ are given in either joint angles or world coordinates. In our experiments, the feature mapping $\phi\big(\mathrm{pa}\big(\mathbf{x}^{(t)}\big)\big)$ concatenates all vectors in the subsequence $\mathrm{pa}\big(\mathbf{x}^{(t)}\big)$ and adds a constant that models an intercept and allows for affine regression functions.

### 4.4.1 Human Action Recognition

The MSRC-12 dataset comprises sequences of people performing a total of 12 iconic and metaphoric gestures. Every sequence is the output of the Kinect's human body tracker, which gives a noisy estimate of 20 joints (see Fig. 2.6a). All joints are defined in terms of $xyz$-world coordinates, resulting in a $d = 60$ dimensional vector $\mathbf{x}^{(t)}$ per frame.

We use the iconic gestures from this dataset, which amounts to 296 sequences of about 1000 frames length each. The task is to assign those sequences to one of the six iconic gesture classes

$$\mathcal{G} := \{\text{Duck}, \text{Goggles}, \text{Shoot}, \text{Throw}, \text{Change Weapon}, \text{Kick}\}. \tag{4.13}$$

Each class comprises approximately 50 sequences coming from 30 different persons. In order to assess the generalization capabilities of our approach across persons, we employ 5-fold leave-person-out cross-validation, *i.e.*, each fold consists of 24 persons for training and 6 persons for testing. We train $|\mathcal{G}|$ different DFMs $\{\boldsymbol{\mathcal{T}}_g\}_{g\in\mathcal{G}}$ per fold, one for each gesture, according to Algorithm 1. We report classification accuracies obtained with dynamic forests consisting of 12 trees, all of which were trained with $2^{10}$ tested splits. We use a fixed ridge regularization parameter $\gamma := 10^{-5}$ but vary the tree depth and Markov order.

In order to assign an unseen test sequence $\mathbf{x}^{(\bullet)}$ to its corresponding class $g^*$, we compute the log-likelihood of the sequence under each gesture model and assign it to the class maximizing this quantity,

$$g^* := \operatorname*{arg\,max}_{g'\in\mathcal{G}} \log p\big(\mathbf{x}^{(\bullet)} \mid \boldsymbol{\mathcal{T}}_{g'}\big), \tag{4.14}$$

where the log-likelihoods are given according to Eq. (4.12).

**Baseline Methods.** We compare DFMs to four different baseline methods. Two of them, $k$-nearest neighbours ($k$-NN) and support vector machines (SVM), are standard classification methods, the other two, hidden Markov models (HMM) and dynamic time warping (DTW), are more specialized dynamic models and tailored to time-series data.

**Table 4.1: Action Classification. (a)** Accuracies and runtimes of DFMs and four baseline models. DFMs outperform all other evaluated methods. **(b)** Classification accuracies of DFMs as a function of depth, order, and number of trees. The result in bold is shown in more detail in (a).

| Gesture | $k$-NN | SVM | HMM | | DTW | DFM |
|---|---|---|---|---|---|---|
| | | | | +PCA | | **(ours)** |
| Duck | 88.0 | 88.0 | 94.0 | **98.0** | **98.0** | 96.0 |
| Goggles | 70.0 | 84.0 | 54.0 | 70.0 | **88.0** | **88.0** |
| Shoot | 71.4 | 79.6 | 36.7 | 73.5 | 46.9 | **85.7** |
| Throw | 84.0 | 76.0 | 64.0 | **90.0** | 86.0 | **90.0** |
| Change weapon | 60.4 | 81.3 | 35.4 | 77.1 | 79.2 | **87.5** |
| Kick | 95.9 | 89.8 | **98.0** | **98.0** | 87.8 | **98.0** |
| Accuracy | 78.4 | 83.1 | 63.6 | 84.4 | 81.1 | **90.9** |
| Runtime (sec./seq.) | 9.24 | 189.23 | 0.50 | 0.45 | 107.55 | **0.23** |

**(a)** Comparison to Baselines.

| Trees | Depth | Markov order | | |
|---|---|---|---|---|
| | | 1 | 2 | 3 |
| 1 | 4 | 69.6 | 69.6 | 62.5 |
| 12 | 1 | 86.5 | 88.2 | 87.8 |
| | 2 | 76.4 | 88.5 | 84.1 |
| | 3 | 87.8 | 90.2 | 89.2 |
| | 4 | 87.2 | **90.9** | 90.5 |

**(b)** DFM Results.

For the $k$-NN and SVM approaches, we classify all length-four subsequences in a given test sequence. The class label of the entire sequence is determined by taking the majority vote over all of those subsequences. In order to classify a subsequence using $k$-NN, we compare it to all $\approx$ 256,000 subsequences of the training set and find the $k$-nearest neighbours with respect to the Euclidean distance. The parameter $k$ is set to 6, which yields best performance on the test set in the range $k = 1, \dots, 8$. The SVM classifier is also trained on all $\approx$ 256,000 subsequences. We use the implementation [22] with a one-vs.-rest classifier and a Gaussian RBF kernel.

DTW is a powerful time-series classifier that aligns two sequences by computing a possibly nonlinear warping path between them, thereby ignoring variations in duration and speed. We use the reference implementation of the FastDTW authors [135] and vary the search radius between $2^4$ and $2^8$. A test sequence is classified by calculating the warp distance to all training sequences (normalized by path length) and assigning it to the class of the training sequence with minimal warp distance.

For the HMM experiments, we use the implementation of [110] and train one HMM per gesture class. The original authors tuned their implementation for the same type of Kinect skeletal data we use. The implementation supports the use of raw features and PCA-reduced features (+PCA) and we report results for both options. The PCA features are obtained by constructing subsequences of length four and using the coefficients of the first 12 principal components. This is a powerful preprocessing step that is necessary for the HMM to work. To find the number of hidden states, we perform model selection, testing 5, 10, 20, and 40 hidden states, and report the best test performance. At test-time we classify a sequence by evaluating the marginal observation log-likelihood for each HMM and assign the sequence to the class of highest likelihood.

**Results.** Table 4.1a shows the quantitative results of our DFM and a comparison to the baseline methods. The worst performance was delivered by the $k$-NN approach (78.4%), which is not surprising given its naïve exhaustive search. Overall the time-series models tend to perform better than the simple classification baselines. That said, DTW (81.1%) still falls short of the SVM accuracy (83.1%), but mostly due to its weak performance on the "Shoot" class (46.9%). Varying the search radius does not eliminate this problem and the maximum is already attained for $2^4$. For the HMM the best result is obtained with 10 hidden states. Note that the HMM requires strong preprocessing: Using raw data the performance is inferior at 63.6% and only the PCA features make the HMM perform at 84.4%. Our DFM achieves an accuracy of 90.9% and outperforms all other evaluated methods, notably without relying on any form of preprocessing. This indicates that DFMs are very robust with respect to the features used. In Table 4.1b we show DFM results for different tree depths and Markov orders. For comparison, we also include accuracies for a single tree of depth 4. The results of the DFM are both better and more stable, proving the point that our approach reduces overfitting and increases predictive power.

### 4.4.2 Motion Completion

In this experiment, we apply the DFM to a motion completion task on walking sequences and compare our results to a recent Gaussian process dynamical model (GPDM). Furthermore, we demonstrate the suitability of the presented framework for more complex actions and provide some general insights into training of DFMs. All of our motion completion experiments use the CMU motion capture database, a rich and noiseless data source of people performing different activities. All sequences are given by 56 joint angles and an additional 6 parameters governing the global translation and orientation.

Motion completion refers to the task of recovering consecutively missing frames from a motion sequence. More specifically, given a complete motion sequence

$$\mathbf{x}^{(\bullet)} = \left(\mathbf{x}^{(1:i)}\ \mathbf{x}^{(i+1:i+j)}\ \mathbf{x}^{(i+j+1:T)}\right) \in \mathbb{R}^{d \times T}, \tag{4.15}$$

we remove the subsequence $\mathbf{x}^{(i+1:i+j)}$ of length $j$ in the middle and compute an estimate $\widehat{\mathbf{x}}^{(i+1:i+j)}$ from the remaining frames based on our model.

**Table 4.2: Motion Completion. (a)** Walking: Joint angle $\overline{\text{RMSE}}$ of our DFM approach and two baseline models. LI = linear interpolation; GPDM = Gaussian process dynamical model. **(b)** Jump forward and golf swing: World coordinate $\overline{\text{RMSE}}$ of our DFM approach as a function of tree depth and Markov order. Note the decrease of the error with increasing tree depth and/or Markov order.

| Seq. no. | LI | GPDM | | DFM |
|---|---|---|---|---|
| | | B-GPDM | 2-MAP | **(ours)** |
| 07-01 | 88.72 | 65.38 | 68.69 | **28.94** |
| 07-02 | 90.13 | 64.47 | 64.43 | **33.87** |
| 08-01 | 113.71 | 70.05 | 72.61 | **30.47** |
| 08-02 | 112.26 | 72.11 | 90.80 | **28.59** |
| 12-02 | 62.02 | 37.06 | **26.60** | 35.83 |
| 12-03 | 58.87 | 40.40 | **23.16** | 29.79 |
| 16-21 | 68.10 | 32.87 | 53.13 | **24.00** |
| 35-03 | 61.07 | **12.15** | 20.74 | 16.45 |
| Mean | 81.86 | 49.31 | 52.52 | **28.49** |

**(a) Joint Angle Representation.** $\overline{\text{RMSE}}$ for 8 walking sequences and different models. The best results are highlighted in bold.

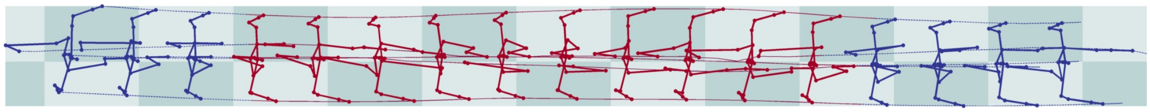
| Gesture | Depth | Order | | | |
|---|---|---|---|---|---|
| | | 1 | 2 | 3 | 4 |
| Golf swing | 1 | 10.93 | 7.44 | 7.06 | 6.89 |
| | 2 | 9.15 | 9.62 | 6.19 | 5.96 |
| | 3 | 4.82 | 5.16 | 5.79 | 4.35 |
| | 4 | 4.58 | **4.17** | 4.75 | 4.31 |
| Forward jump | 1 | 17.28 | 17.07 | 18.45 | 16.38 |
| | 2 | 39.59 | 12.01 | 11.70 | 10.61 |
| | 3 | 12.94 | 11.47 | 10.53 | 10.44 |
| | 4 | 11.08 | 11.27 | 11.41 | **9.19** |

**(b) World Coordinate Representation.** $\overline{\text{RMSE}}$ for two more complex gestures. Results are shown for different tree depths and Markov orders.

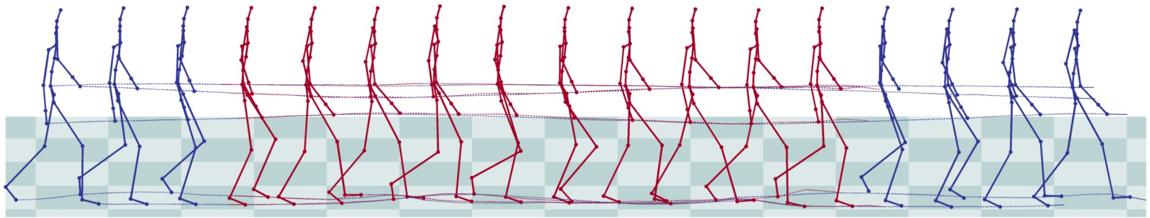
As in [164], we use the following three preprocessing steps for all CMU sequences: 1. A modified skeleton (reducing $d$ from 62 to 50); 2. Temporal downsampling by a factor of four; 3. Centering of all variables. After that, we take the first $T = 50$ frames of the 8 test sequences listed in Table 4.2a and consider 12 different starting positions $i \in S = \{4, \ldots, 15\}$ for the removal of $j = 31$ frames. The sequences come from 5 different subjects walking at different speeds and with different styles. Infilling of the missing frames involves training of a DFM $\{\mathcal{T}_c\}_{c=1,\ldots,C}$ with a total of 29 sequences, all of them preprocessed in the same way as described above and none of them part of the test set. The estimate for missing frame $j' \in \{1, \ldots, 31\}$ in run $i$ is then given by the conditional expectation

$$\begin{aligned}\widehat{\mathbf{x}}^{(i+j',i)} &:= \mathbb{E}\Big[\mathbf{X}^{(i+j',i)}\,\Big|\,\mathrm{pa}\Big(\widehat{\mathbf{x}}^{(i+j',i)}\Big)\Big]\\ &= \frac{1}{C}\sum_{c=1}^{C}\mathbf{A}_{\ell(i+j',c)}\,\phi\Big(\mathrm{pa}\Big(\widehat{\mathbf{x}}^{(i+j',i)}\Big)\Big).\end{aligned} \tag{4.16}$$

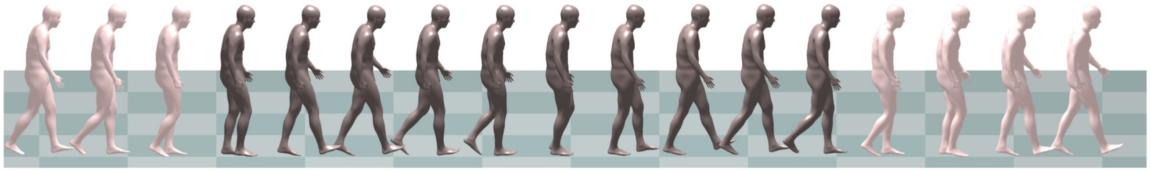
The estimates $\widehat{\mathbf{x}}^{(i+1:i+j,i)}$, $i \in S$, of all 12 runs are subsequently combined to give an



**(a)** Input: 2D trajectory (top view).



**(b)** Output: Stick model visualization of predicted 3D trajectory.



**(c)** Output: SCAPE model visualization of predicted 3D trajectory.

**Figure 4.4: Visualization of a 2D→3D Task. (a)** Every third frame of a 2D input sequence for which we want to reconstruct the red subsequence in 3D. **(b,c)** Two animations of the output from our model: A stick model (center) and a SCAPE model (bottom) that was fitted based on our reconstruction. Ground truth data is shown in blue/light skin, reconstructions in red/dark skin.

average RMSE per frame

$$\overline{\mathrm{RMSE}}(\hat{\mathbf{x}}^{(\bullet)}) := \frac{1}{\sqrt{j}|S|}\sum_{i\in S}\sqrt{\sum_{j'=1}^{j}\left\|\hat{\mathbf{x}}^{(i+j',i)} - \mathbf{x}^{(i+j')}\right\|^2} \tag{4.17}$$

as a measure of reconstruction quality.

Although a direct comparison to the GPDM is illustrative, our model is not limited to simple actions such as walking. In fact, complex actions benefit more from our nonlinear and non-parametric approach. To fortify this claim, we train two additional DFMs on more challenging gestures: forward jump and golf swing. In particular, we use the same set of preprocessing steps as before and train DFMs on 7 (forward jump) and 8 (golf swing) training sequences, with 1 sequence in each category reserved for testing. Both test sequences are missing 31 frames and the

RMSE is again averaged over 12 runs.

While our experiments on walking sequences use a representation in joint angles to allow a comparison with the results in [164], our experience has shown that DFMs perform better when trained on a representation in $xyz$-world coordinates. In particular, the ability to benefit from deep trees and to map distinct parts of a motion sequence to distinct linear systems works well in the $xyz$-domain. We therefore convert the involved sequences to $xyz$-coordinates and use this new representation for our two additional motion completion experiments.

**Results.** Table 4.2a summarizes the quantitative results of the experiments on walking sequences and compares our findings with simple linear interpolation (LI) and the Gaussian process dynamical model (GPDM). As expected, linear interpolation performs worst, with an average $\overline{\text{RMSE}}$ of 81.86. The original authors of the GPDM [164] consider four different learning procedures and we restate the numbers for the two best performing approaches: B-GPDM, which adds a balancing term to MAP estimation, and 2-MAP, which is a two-stage process involving a hybrid Monte Carlo version of EM and scaled conjugate gradient. The former performs slightly better (49.31 vs. 52.52). We compare these results with a DFM consisting of 12 trees. Our method achieves a lower $\overline{\text{RMSE}}$ on 5 out of 8 test sequences.

Table 4.2b shows our results for the actions ‘forward jump’ and ‘golf swing’. DFMs are suited for these more complex actions just as well. Both gestures take full advantage of our distributed approach and the minimum error is reached for a tree depth of 4. The table also suggests that depth is more important than order: While we do see some improvement with increasing Markov order, the error decreases much more substantially with increasing tree depth. Our assumption that long-range dependencies are only of minor importance thus seems to hold.

### 4.4.3 Predicting 3D from 2D

For our last demonstration, we consider the task of predicting 3D motion from 2D inputs. In combination with the output of a pose estimation pipeline, we see potential applications like automatic character animation from videos.

In particular, we use the same walking sequences and preprocessing steps as we did in section 4.4.2 but train our model with pairs of 2D inputs and 3D outputs, where we obtain the 2D data from orthographic projections to the $xz$-plane (top view) and the $yz$-plane (side view).

Our results are summarized in Table 4.3. When using 2D views from the top, our findings are in accordance with those in the previous section, *i.e.*, the $\overline{\text{RMSE}}$

**Table 4.3: 3D from 2D.** $\overline{\text{RMSE}}$ of predicted 3D trajectory given a 2D input. Deeper trees perform consistently better, while the effect of the Markov order varies.

| 2D Input | Depth | Markov order | | | |
|---|---|---|---|---|---|
| | | 1 | 2 | 3 | 4 |
| Top | 1 | 6.08 | 6.18 | 8.17 | 7.72 |
| | 2 | 3.55 | 3.53 | 3.44 | 3.71 |
| | 3 | 2.19 | 1.92 | 1.71 | 1.55 |
| | 4 | 1.87 | 1.37 | 1.24 | **1.23** |
| Side | 1 | 7.31 | 8.03 | 8.96 | 9.37 |
| | 2 | 4.11 | 4.23 | 4.62 | 5.30 |
| | 3 | 3.13 | 3.23 | 3.35 | 3.76 |
| | 4 | **2.97** | 3.19 | 3.39 | 3.65 |

decreases considerably when using deeper trees and higher Markov orders, although the former seems to have a higher influence. Note, for instance, the decrease in error of 84% between the best performing model, a tree of depth 4 and Markov order 4, and its counterpart of depth 1. For 2D views from the side, the test sequences cannot take advantage of higher orders, but the benefit from deep trees still leads to a substantial improvement of 59% compared to trees of depth 1.

Finally, we want to complement our numerical evaluation with a visual inspection of the reconstructed sequences: Fig. 4.4a shows a stickman visualization of the available 2D input data (projections to $xz$-plane). Based on the predicted 3D motion trajectories from our model (Fig. 4.4b), we can produce a realistic looking animation of a SCAPE model [4] (Fig. 4.4c).

## 4.5 Computational Aspects

One of the major benefits in using DFMs instead of latent variable models lies in their combination of being both flexible and tractable. The calculation of an exact log-likelihood for our action classification experiment takes only 0.04 sec. when using a dynamic forest with 12 trees of depth 4. We can thus classify a sequence in around 0.23 sec. Likewise, one synthesis step in our motion completion experiment takes 0.05 sec. All numbers refer to our Matlab implementation. Table 4.1a shows how that compares to the other methods. The DFM is almost twice as fast as an HMM and magnitudes faster than the remaining methods.

# Chapter 5

# Non-parametric Scene Prior Hierarchies

So far, we have presented non-parametric models for single tasks: human pose in chapter 3 and human motion in chapter 4. However, related tasks depend on and provide semantic context for each other. A special case occurs if the considered tasks constitute a coarse-to-fine hierarchy with respect to the information they comprise, in which case we can order them in a linear chain. The individual models along such a chain can be conditioned on the information provided by earlier models. We present and discuss an incarnation of such a hierarchical system for object quantification and detection and show how to incorporate powerful CNN-based image likelihoods.

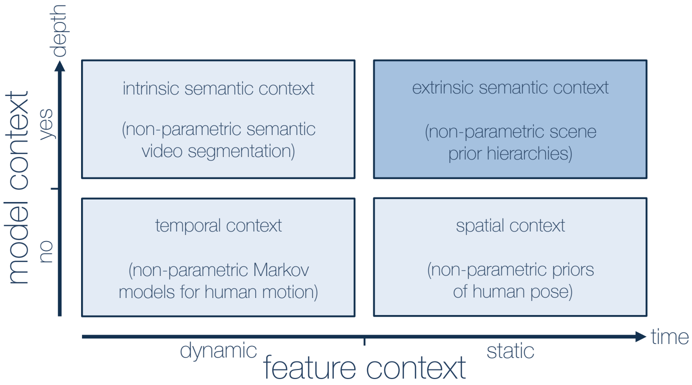

depth
model context
yes
no
intrinsic semantic context
(non-parametric semantic video segmentation)
extrinsic semantic context
(non-parametric scene prior hierarchies)
temporal context
(non-parametric Markov models for human motion)
spatial context
(non-parametric priors of human pose)
time
dynamic
static
feature context

**Figure 5.1: Key Message: Extrinsic Semantic Context.** A first-order hierarchy of conditional Bayesian networks with dynamic topology can be used to model coarse-to-fine dependencies in natural scenes, *e.g.*, between object occurrences and object locations.

## 5.1 Introduction

Scene understanding frameworks use image features comprising a visual scene to categorize, quantify, detect, and segment objects in an image. While the advancement of feature learning methodologies (*e.g.*, convolutional neural networks [87]) and availability of large-scale datasets (*e.g.*, Microsoft COCO [99]) have lead to a tremendous increase in performance of local classifiers in multi-class categorization and detection tasks [55, 23], structured approaches like graphical models will continue to play major roles in computer vision to encode dependencies and enforce consistency between those independent predictions. First, because independent predictions based on local evidence, *e.g.*, SVM predictions based on CNN features derived from bounding box proposals, are unlikely to ever surpass a certain level of accuracy; and second, because generative priors can produce image- and context-conditioned hypotheses and likelihoods, which enables reasoning in scenarios with partial evidence or incomplete data.

The importance of a generative prior for *scene understanding* can be motivated in many ways: Without any evidence, it can serve as a regularization term or hypothesis generator for any probabilistic model of natural scenes. Given partial evidence in the form of context information, it can answer *"what if?"* questions, *e.g.*, about likely quantities, positions and scales of an object category given occurrence or location information of other object categories or instances ("Where is the cutlery given the location of the table and the information that there are two plates?"). Applications for this type of queries range from artificial image composition [91] to making inferences about an object's role, its interaction with other objects or the atypicality of its use in a given context [25]. Finally, given image evidence, a scene prior can detect and correct inconsistent scene layouts produced by independent classifiers (*e.g.*, detections of persons or cars above the ground [65]).

We argue that a good scene prior should have the following properties: (1) First and foremost, it should be able to represent the true generating distribution and answer (conditional) queries about it. Most importantly, it should be able to evaluate the (context- or image-conditioned) likelihood of a specific configuration of objects in a scene; (2) It should allow for efficient sampling of plausible scenes, either conditioned on evidence or fully generative; (3) It should be able to take any and all available information into account when performing (1) and (2), which suggests a set of interrelated priors specializing in dedicated tasks, so that extrinsic information provided by one can steer intrinsic behaviour of another; (4) It should allow inference at different levels of granularity, so that less granular tasks (*e.g.*, object occurrence) do not require inference over more granular variables (*e.g.*, object location).

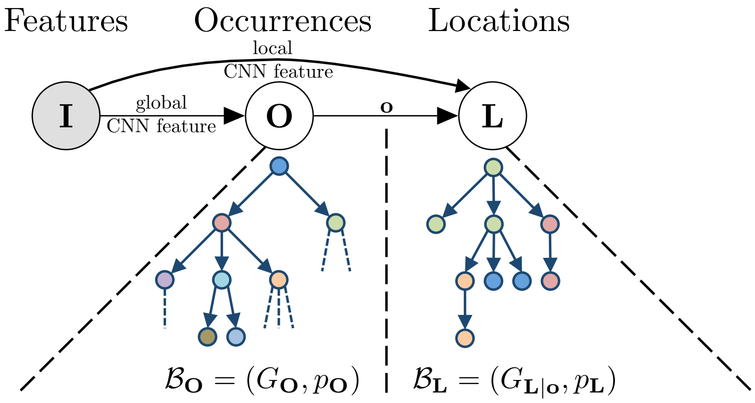

Features
Occurrences
Locations
local
CNN feature
global
CNN feature
I
O
o
L
$\mathcal{B}_{\mathbf{O}} = (G_{\mathbf{O}}, p_{\mathbf{O}})$
$\mathcal{B}_{\mathbf{L}} = (G_{\mathbf{L}|\mathbf{o}}, p_{\mathbf{L}})$

**Figure 5.2: Scene Prior Hierarchy.** A first-order hierarchy of conditional Bayesian networks encodes information from coarse to fine. Global topology and local models are dynamically estimated based on preceding hierarchical layers.

### 5.1.1 Approach

Guided by the motivation above, we propose a scene prior architecture consisting of a novel first-order hierarchy of conditional Bayesian networks with structured global topology and parametric as well as non-parametric local models at the individual layers (Fig. 5.2).

**Global Architecture.** Our choice of a first-order hierarchy is lead by the observation that the prototypical challenges in scene understanding form a matryoshka-like system of nested dependencies,

$$\text{categorization} \rightarrow \text{quantification} \rightarrow \text{detection} \rightarrow \text{segmentation}, \tag{5.1}$$

*i.e.*, each level comprises a superset of the information present at the previous level. Our approach mimics those natural dependencies and allows their joint modeling in a causal framework, so that fine-grained tasks can benefit from coarse evidence. In this work, we illustrate this idea by building a bilayered hierarchy over object occurrence (quantification) and object location (detection). Both layers can optionally be conditioned on an observed image, effectively resulting in a prior-based rescoring of independent image likelihoods. To make a prediction, we run a forward-pass of level-dependent inference schemes, from coarse to fine, and stop as soon as we reach a desired level of granularity. Each task along the chain is modeled by a separate (conditional) Bayesian network whose dynamic global structure and local models depend on the previous output. The choice for Bayesian networks in favor of undirected graphical models is motivated by the former's generative principles, which allow for efficient sampling and a simple encoding of conditional independencies.

**Local Architecture.** At its statistical core, a generative scene prior is a high-dimensional density that maps parameterizations of scenes to likelihoods. Ideally, we would therefore like to model the individual Bayesian networks along our hierarchical chain using a fully-connected graph structure (≙ no topological independence assumptions) and non-parametric local models (≙ no distributional shape assumptions), as discussed in section 2.6. Unfortunately, this is not possible due to insufficient training data (as indicated by discrete PAC bounds [49] and continuous AMIAE analyses [141]) and intractable inference procedures (due to high treewidth [127]). Instead, we follow our design principles and let the topological and distributional structure of each network be a trade-off between high flexibility and high efficiency, *i.e.*, a compromise between an unstructured non-parametric model with high variance and intractable inference (*e.g.*, a kernel density estimate) and a structured parametric model with high bias and low expressiveness (*e.g.*, a Gaussian linear network). More specifically, we obtain a bias-variance trade-off by combining the efficiency of low-treewidth topologies with the flexibility of non-parametric local models. The former are given by conditional arborescences, which we learn from Microsoft COCO [99] by generalizing our previous approach based on [26] to continuous data in dynamic settings. The latter are given by conditional kernel density estimates whose bandwidths we estimate according to Scott's rule [141].

Our large-scale experiments aim at showing two things: (1) A structured non-parametric scene prior hierarchy is superior to both independent non-parametric priors and structured parametric priors; (2) It improves the predictive performance of a generative object detection framework based on state-of-the-art convolutional neural network features (fast/er R-CNN [54, 124]).

### 5.1.2 Related Work

Over the years, there have been numerous attempts to build contextual and hierarchical scene models of various forms for recognition:

**Contextual Scene Models.** Conceptually simple approaches focus on the co-occurrence [123] or mutual exclusion [33] of object categories in a given image. Others look at the enhancement or inhibition of detections using contextual relations based on both co-occurrence and co-location, *e.g.*, through the use of structured image labels [34], visual phrases [132], or discovered object groups [92, 98]. Sequential detection, such that weaker detectors (*e.g.*, keyboard) can benefit from stronger ones (*e.g.*, monitor), was investigated in [52, 155]. Some models look at context across granularities, for instance, by using texture patches (*e.g.*, trees or sky) to enhance the performance of object detectors with fixed extent [61]. We take inspiration from [52, 61, 155] in that we also believe that certain recognition tasks are easier than others and can provide context for more complex tasks. However, instead of modeling such contextual relationships only at the object(–texture patch) level, we model them across recognition tasks (quantification → detection).

**Hierarchical Scene Models.** Hierarchical models attempt to model scenes at different levels of granularity. Well-known examples include the discriminative joint scene model of [175], which encodes consistency of scene classification, object detection, and object segmentation through a structured output space, and the generative scene–object–part model of [152]. Our model is inherently generative, hence closer to [152], though we employ MAP inference in order to predict object locations. Unlike [175] and [152], we perform inference only over the variables that are of interest for a specific task; [175] can only return a MAP solution over *all* variables. In contrast to [152], we are not limited to pure MCMC schemes, because inference over object occurrences can be done efficiently using standard BP in our model. As such, our model is most similar to [25], where a hierarchical context model over object occurrences and locations is proposed. However, our model has two notable distinctions: First, we use non-parametric densities to model pairwise location relationships, as opposed to unimodal Gaussian distributions in [25], which allows us to capture much more complex spatial relationships. Second, and more importantly, we do not assume that the graph topology of object occurrences and object locations is one and the same. Instead, we generate the latter dynamically from continuous mutual information tables. This is an important distinction: For instance, cars and people may co-occur very often, but they do not have strong spatial relations with respect to one another; our model can encode such task-dependent differences.

## 5.2 Non-parametric Scene Priors

In this section, we introduce our non-parametric scene prior, describe its training, and provide a simple formulation for its integration with a general-purpose object detection framework. We follow a top-down description, first describing the general hierarchical architecture and then discussing the details of each layer in the subsequent sections.

**Prior Hierarchies.** A Bayesian network $\mathcal{B} = (G, p_{\mathbf{X}})$ is fully specified in terms of two components: A directed, acyclic graph $G = (\mathbf{X}, E)$, where $\mathbf{X} = \{X_i\}_{i=1}^{n}$ is a node set of random variables and $E \subseteq \mathbf{X} \times \mathbf{X}$ is an edge set, and a joint probability distribution $p_{\mathbf{X}}$ that factorizes over $G$,

$$p_{\mathbf{X}}(\mathbf{x}) = \prod_{i=1}^{|\mathbf{X}|} p_{X_i}(x_i \mid \mathrm{pa}_G(x_i)), \tag{5.2}$$

where $\mathrm{pa}_G$ maps a variable to its parent variables with respect to $G$, *i.e.*, $(X_i, X_j) \in E$ if and only if $X_i \in \mathrm{pa}_G(X_j)$.

In this work, we consider a first-order hierarchy of *conditional* Bayesian networks, that is, a linear chain of Bayesian networks $\mathcal{B}^{(1)}, \ldots, \mathcal{B}^{(L)}$ in which each network $\mathcal{B}^{(i)} = (G^{(i)}, p_{\mathbf{X}^{(i)}})$ along the chain is conditioned on its parent network $\mathcal{B}^{(i-1)}$ and (optionally) an observed RGB image $I$. Writing $\mathcal{X} := \bigcup_{i=1}^{L} \mathbf{X}^{(i)}$ and setting $\mathbf{X}^{(0)} := \varnothing$, we have

$$p_{\mathcal{X}}\big(\mathbf{x}^{(1:L)} \mid I\big) = \prod_{i=1}^{L} p_{\mathbf{X}^{(i)}}\big(\mathbf{x}^{(i)} \mid \mathbf{x}^{(i-1)}, I\big), \tag{5.3}$$

where $p_{\mathbf{X}^{(i)}}$ factorizes over $G^{(i)}$ and both may depend on $\mathbf{x}^{(i-1)}$. The hierarchical chain $\mathbf{X}^{(1)} \to \ldots \to \mathbf{X}^{(L)}$ encodes information from coarse to fine, which means that $\mathcal{B}^{(i)}$ encodes a superset of the information encoded by $\mathcal{B}^{(i-1)}$. Intuitively, each network is thus only responsible for the differential information between its parent and itself. We can use this property of *information enclosure* to reason about an image at different levels of granularity, because each intermediate level $L' < L$ constitutes a proper joint distribution over $\bigcup_{i=1}^{L'} \mathbf{X}^{(i)}$.

**Scene Prior Hierarchies: Occurrence $\mathcal{B}_{\mathbf{O}} \to$ Location $\mathcal{B}_{\mathbf{L}}$.** As a specific instantiation of the general prior hierarchies formulation above, we build a bilayered scene prior hierarchy over object occurrence and object location. As such, our scene prior consists of a parametric occurrence network $\mathcal{B}_{\mathbf{O}} = (G_{\mathbf{O}}, p_{\mathbf{O}})$ over occurrence variables $\mathbf{O}$ and a non-parametric location network $\mathcal{B}_{\mathbf{L}} = (G_{\mathbf{L}}, p_{\mathbf{L}})$ over location variables $\mathbf{L}$. $\mathbf{O}$ models each category's quantity and $\mathbf{L}$ models each instance's position. Following Eq. (5.3), both networks are not only conditioned on an image $I$ but also exhibit a structural dependency on each other,

$$p_{\mathbf{OL}}(\mathbf{o}, \mathbf{l} \mid I) = p_{\mathbf{O}}(\mathbf{o} \mid I) \cdot p_{\mathbf{L}}(\mathbf{l} \mid \mathbf{o}, I). \tag{5.4}$$

We will sometimes write $G_{\mathbf{L}|\mathbf{o}}$ or $p_{\mathbf{L}|\mathbf{o}}$ to highlight this dependence of $\mathcal{B}_{\mathbf{L}}$ on $\mathcal{B}_{\mathbf{O}}$. Note that the second layer $p_{\mathbf{L}}$ is based on more granular information than the first layer

$p_{\mathbf{O}}$: If we know the locations of the object instances across all classes, we also know their occurrences. Extensions in both directions (finer and coarser) are possible[1] but beyond the scope of this work. Fig. 5.2 depicts a graphical model of this high-level structure and the following two sections describe both layers in more detail.

### 5.2.1 Occurrence Model

The occurrence network $\mathcal{B}_{\mathbf{O}} = (G_{\mathbf{O}}, p_{\mathbf{O}})$ is a parametric, image-conditioned Bayesian network that models both the presence/absence of object categories as well as the number of their instances. Formally, we introduce one occurrence variable $O_k$ per category $k$, *i.e.*, $\mathbf{O} = \{O_k\}_{k=1}^{K}$, where $K$ is the total number of categories. To guarantee maximum flexibility, we assume categorical marginal distributions $O_k \sim \mathrm{Cat}(\mathbf{p} = (p_i)_{i \in \mathbb{Z}_{m_k}})$ with probability mass function $p_{O_k}(o_k; \mathbf{p}) = \prod_i p_i^{[o_k = i]}$.[2] The maximum number of instances $m_k - 1$ varies across categories $k$ and is determined at training time. The joint distribution is assumed to factorize over $G_{\mathbf{O}}$ according to Eq. (5.2),

$$p_{\mathbf{O}}(\mathbf{o} \mid I) = \prod_{k=1}^{K} p_{O_k}\big(o_k \mid \mathrm{pa}_{G_{\mathbf{O}}}(o_k), I\big). \tag{5.5}$$

An examplary instantiation $\mathbf{o}$ of $\mathbf{O}$ is illustrated in Fig. 5.5a. We incorporate image evidence by factoring the conditional distributions into log-convex combinations with $\beta$-weighted contributions of prior and likelihood,

$$p_{O_k}\big(o_k \mid \mathrm{pa}_{G_{\mathbf{O}}}(o_k), I\big) \propto p_{O_k}\big(o_k \mid \mathrm{pa}_{G_{\mathbf{O}}}(o_k)\big)^{\beta} \cdot p_{O_k}(o_k \mid I)^{(1-\beta)}. \tag{5.6}$$

For $\beta = 1$, we obtain a generative, image-independent occurrence prior and for $\beta = 0$, we get back the original, priorless occurrence likelihoods that operate independently on each object category $k$.[3] The likelihood can be modeled in a very general way as the function composition of a global image feature $\phi_o : \mathcal{I} \to \Phi$ with a category-specific classifier $c_o^{(k)} : \Phi \to \mathcal{P}$,

$$p_{O_k}(o_k \mid I) = \big[\big(c_o^{(k)} \circ \phi_o\big)(I)\big](o_k), \tag{5.7}$$

where $\mathcal{I}$ is the set of all images, $\Phi$ is a feature space and $\mathcal{P}$ is the set of probability mass functions on $\mathbb{Z}_{m_k}$.

---

[1] A binary presence/absence class categorization network is coarser than $\mathcal{B}_{\mathbf{O}}$, an instance segmentation network is finer than $\mathcal{B}_{\mathbf{L}}$.

[2] $[\cdot]$ is the Iverson bracket and $\mathbb{Z}_n = \{0, \ldots, n-1\}$.

[3] The distribution in Eq. (5.6) is not normalized, but the local partition function can be readily obtained as the (weighted) inner product of the two parameter vectors.

**Global Topology $G_{\mathbf{O}} = (\mathbf{O}, E_{\mathbf{O}})$ of $\mathcal{B}_{\mathbf{O}}$.** Eqs. (5.5) and (5.6) reference the global topology $G_{\mathbf{O}}$. There are many reasonable ways to define an edge set $E_{\mathbf{O}}$ over the node set $\mathbf{O}$, *e.g.*, based on our intuitive understanding of the semantic relationships between object categories, their distances in well-established semantic object hierarchies like WordNet [41], or their attributes. However, they might not lead to an optimal graph structure in an information-theoretic sense, which is given by the projection of the true generating distribution $p^*_{\mathbf{O}}$ onto a target class that allows for reliable learning and tractable inference. We project $p^*_{\mathbf{O}}$ onto the set of distributions that factorize over the robust and efficient class of arborescences $\mathcal{T}_{\mathbf{O}}$, so that our objective function is

$$G_{\mathbf{O}} = \underset{G'_{\mathbf{O}} \in \mathcal{T}_{\mathbf{O}}}{\arg\min}\, \mathrm{KL}\left( p^*_{\mathbf{O}} \,\middle\|\, \prod_{k=1}^{K} p_{O_k}\left( o_k \,\middle|\, \mathrm{pa}_{G'_{\mathbf{O}}}(o_k) \right) \right). \tag{5.8}$$

As discussed in chapter 3, this seemingly complex optimization problem has a closed-form solution in terms of the maximum spanning tree (MST) of a complete graph over $\mathbf{O}$ whose edge weights $w(O_k, O_l)$ are given by the mutual informations $\mathrm{MI}(O_k, O_l)$ between the corresponding random variables [26]. The MST is undirected, but since any two arborescences with the same undirected skeleton encode the same set of conditional independence assumptions, we can choose an arbitrary root node and direct all edges away from it to obtain an arborescence.

Now that the local models and global topology are fully specified, we can turn to the question of how to estimate them from a training set.

### Learning

Let $\mathcal{D}_{\mathbf{o}} = (o_k^{(i)})_{ki} \in \mathbb{N}_0^{K \times N}$ be a training set of occurrences, where $N$ is the number of annotated training images, $K$ is the number of object classes and $o_k^{(i)}$ is the number of occurrences of category $k$ in image $i$. Learning $\mathcal{B}_{\mathbf{O}}$ from $\mathcal{D}_{\mathbf{o}}$ consists of learning (1) the graph topology $\widehat{G}_{\mathbf{O}}$; (2) the local occurrence priors $\widehat{p}_{O_k}(o_k \mid o_l)$, with $o_l := \mathrm{pa}_{\widehat{G}_{\mathbf{O}}}(o_k)$; and (3) the local occurrence likelihoods $\widehat{p}_{O_k}(o_k \mid I)$.

**(1) Learning the Global Topology $\widehat{G}_{\mathbf{O}}$.** We estimate all pairwise marginal distributions $\widehat{p}_{O_k O_l}(o_k, o_l)$, $k \neq l$, and compute the mutual information between the variables in question using

$$\widehat{\mathrm{MI}}(O_k, O_l) = \widehat{H}(O_k) + \widehat{H}(O_l) - \widehat{H}(O_k, O_l), \tag{5.9}$$

where $\widehat{H}(\bullet) = \mathbb{E}[-\ln \widehat{p}_{\bullet}]$ denotes entropy. These give rise to an estimate of the full mutual information graph whose maximum spanning tree (Fig. 5.4a) specifies an optimal topology $\widehat{G}_{\mathbf{O}} = (\mathbf{O}, \widehat{E}_{\mathbf{O}})$.

**(2) Learning the Local Priors $\widehat{p}_{O_k}(o_k \mid o_l)$.** Given $\widehat{G}_{\mathbf{O}}$, we can reuse the pairwise marginals to define the local models

$$\widehat{p}_{O_k}(o_k \mid o_l) = \frac{\widehat{p}_{O_k O_l}(o_k, o_l)}{\sum_{o_k} \widehat{p}_{O_k O_l}(o_k, o_l)} \quad \forall (O_l, O_k) \in \widehat{E}_{\mathbf{O}}. \tag{5.10}$$

In theory, estimating the required conditional distributions is straightforward, but a simple maximum likelihood estimator, which would just count the conditional occurrences, might suffer from high variance and overfitting due to limited training data. In practice, we therefore employ the following smoothing scheme: Let $\mathbf{p} = (p_i)_i$ be the unknown parameter vector of some conditional occurrence distribution $p_{O_k}(o_k \mid O_l = l')$ and $\mathcal{D}_{o_k|o_l=l'} = \{o_k^{(i)} \mid o_l^{(i)} = l'\}$. We assume a Dirichlet conjugate prior $\mathbf{p} \sim \mathrm{Dir}(\boldsymbol{\alpha} = (\alpha_i)_i)$ and compute an estimate $\widehat{\mathbf{p}}$ of $\mathbf{p}$ by minimizing the Bayes risk with respect to a mean squared error loss, which has the closed-form solution

$$\widehat{p}_i = \frac{\sum_{o \in \mathcal{D}_{o_k|o_l=l'}} [o = i] + \alpha_i}{|\mathcal{D}_{o_k|o_l=l'}| + \|\boldsymbol{\alpha}\|_1}. \tag{5.11}$$

Note that this is just an asymmetric form of additive smoothing. We incorporate the prior belief of an exponential decline of occurrence probabilities by letting the relative pseudocounts $\frac{\alpha_i}{\|\boldsymbol{\alpha}\|_1}$ follow a truncated geometric distribution whose single parameter we set based on maximum likelihood estimation.

**(3) Learning the Local Likelihoods $\widehat{p}_{O_k}(o_k \mid I)$.** Possible choices for $\phi_o$ include standard image features like HoG [30] or SIFT [100] as well as the fully-connected layers of a CNN [87, 54]. The only requirement concerning the classifiers $c_o^{(k)}$ is that they return scores that can be (re-)interpreted in a probabilistic sense, *e.g.*, logistic regression, a calibrated SVM [177], or a softmax layer.

### 5.2.2 Location Model

A principled architecture for the location network $\mathcal{B}_{\mathbf{L}} = (G_{\mathbf{L}}, p_{\mathbf{L}})$ is a challenging task due to a number of reasons, including high-dimensional and continuous data, dynamic dependencies on the occurrence network, and complex image dependencies; its design is among the core contributions of this work.

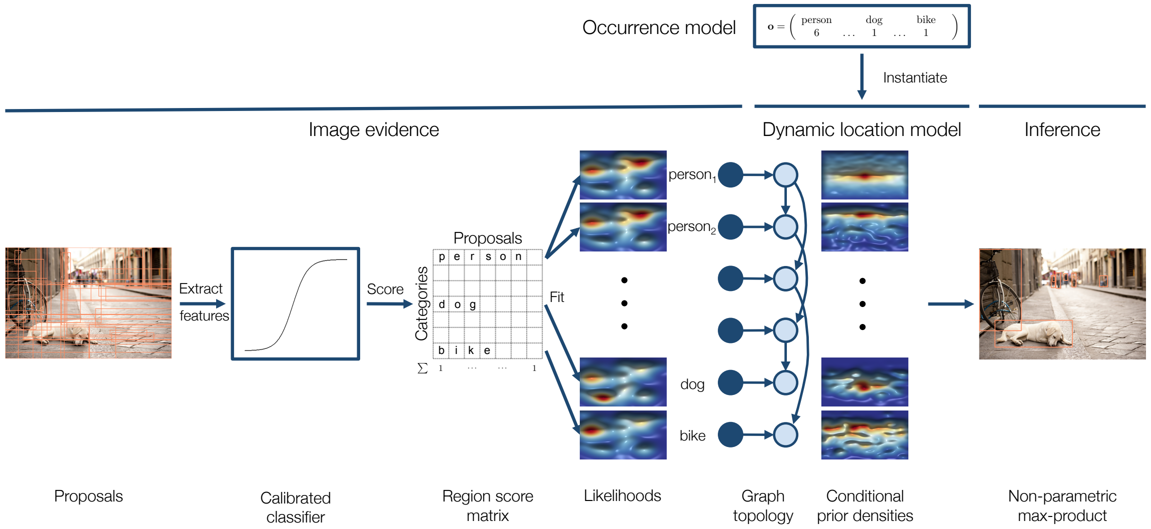


**Figure 5.3: Location Network Architecture.** We produce a fixed number of object proposals and obtain for each proposal a normalized distribution over its class membership using a calibrated classifier. In parallel, we create a dynamic location topology based on the output of the occurrence model. The local models consist of a weighted product of two densities: an image density that we fit based on the classifier scores and a conditional prior density that we learn from data. A final result is obtained through non-parametric max-product inference.

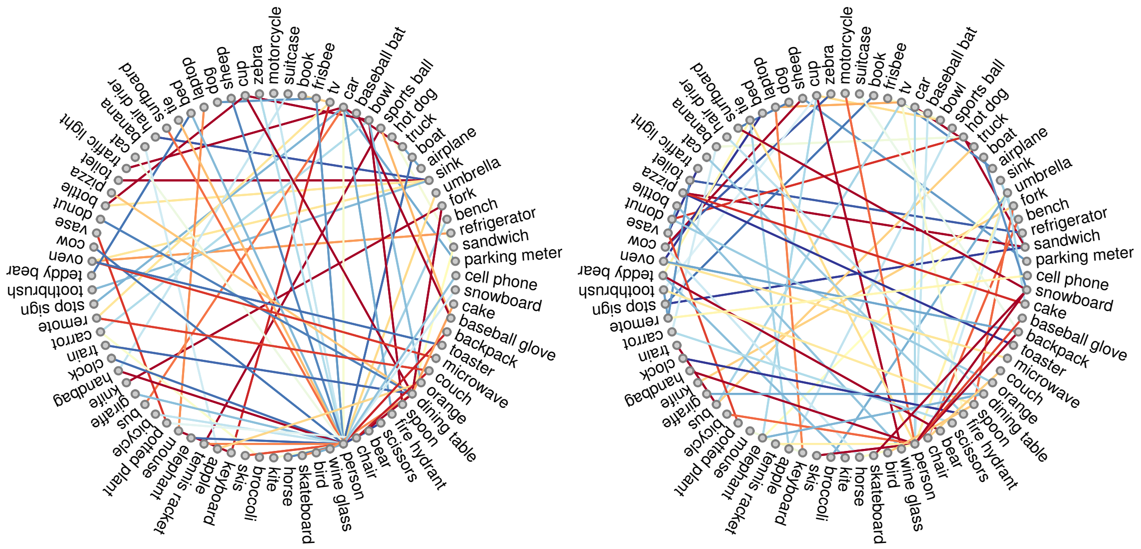
Due to the continuous domain of bounding box parameterizations, our task has shifted from discrete parameter estimation to the notoriously difficult problem of multivariate density estimation. As noted throughout the statistics literature, there is little hope to reliably estimate an unstructured density over locations of multiple 4-dimensional bounding boxes, even for a moderate number of object instances. For instance, Scott [142] estimates that approximately $5.0 \times 10^6$ training samples are needed to learn a density over 2 object instances that is comparable in asymptotic mean integrated absolute error (AMIAE) to a single-instance density estimated from just $5.6 \times 10^3$ training samples. We therefore take an approach similar to the previous section and factor $p_{\mathbf{L}}$ into an arborescence. While enforcing conditional independencies through topological constraints inevitably introduces a global estimation bias, density estimation of the low-dimensional local models becomes feasible and modeling of spurious interactions is avoided. Following the hierarchical scene prior architecture of Eq. (5.4), the location network $\mathcal{B}_{\mathbf{L}}$ depends on both the image $I$ and the occurrence network $\mathcal{B}_{\mathbf{O}}$. We make use of those dependencies to dynamically adjust the location network with respect to the number of active object instances as well as their categories, distributions, and optimal dependencies. At a high level, we use a formulation that bears some resemblance to the one used in the occurrence network,

$$p_{\mathbf{L}}(\mathbf{l} \mid \mathbf{o}, I) \propto \prod_{k=1}^{K} \prod_{j=1}^{m_k - 1} p_{L_{kj}}\Big(l_{kj} \,\Big|\, \mathrm{pa}_{G_{\mathbf{L}|\mathbf{o}}}(l_{kj})\Big)^{\beta} \cdot p_{L_{kj}}(l_{kj} \mid I)^{1-\beta}, \tag{5.12}$$

where $L_{kj} \in \mathbf{L}$ models the 4-dim. bounding box of the $j$-th instance of the $k$-th class in the image $I$ and $\mathbf{L} = \{L_{kj} \mid 1 \leqslant k \leqslant K, 1 \leqslant j \leqslant m_k - 1\}$. Our approach reflects the intuition that a good location model should balance prior assumptions about spatial dependencies between object classes on the one hand and likelihoods based on image evidence on the other hand in a coherent way. Although similar in spirit to Eq. (5.6), the underlying characteristics are very different. Most notably, we incorporate occurrence information $\mathbf{o}$ to obtain a dynamic global topology $G_{\mathbf{L}|\mathbf{o}}$ and dynamic local models $p_{L_{kj}}\Big(l_{kj} \,\Big|\, \mathrm{pa}_{G_{\mathbf{L}|\mathbf{o}}}(l_{kj})\Big)$. Fig. 5.3 provides an overview of this process and the following sections discuss the necessary details.

### Dynamic Global Topology

Even though differential entropy differs in fundamental aspects from the classic Shannon entropy, Eq. (5.9) carries over to the continuous case. At training time, we can therefore use a non-parametric entropy estimator [85] to estimate the continuous



**(a)** Occurrence network. **(b)** Location network.

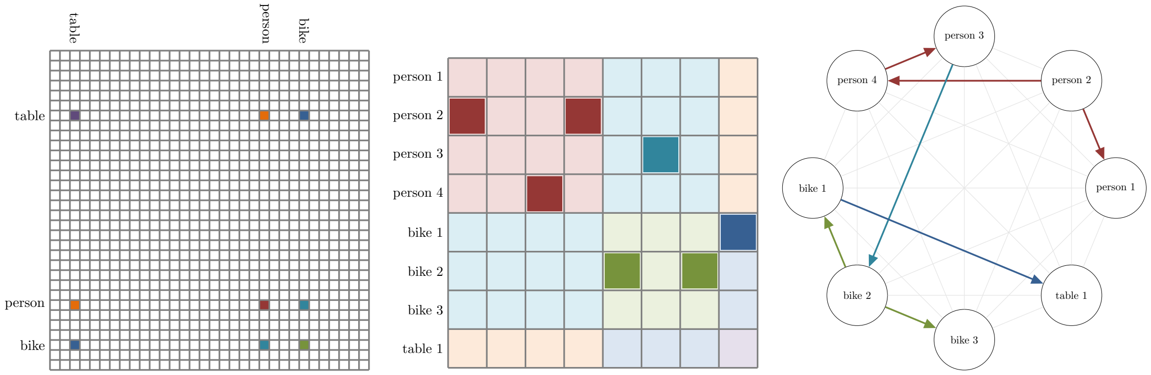
**Figure 5.4: Object Category Topologies.** We compare **(a)** occurrence MI and **(b)** location MI [red $\mathrel{\widehat{=}}$ high MI; blue $\mathrel{\widehat{=}}$ low MI]. The observed differences fortify the need for different topologies in $\mathcal{B}_{\mathbf{O}}$ and $\mathcal{B}_{\mathbf{L}}$. Note that, while (a) shows the actual occurrence topology, the values in (b) are used to generate a dynamic topology on demand.

mutual information between all categories, including self-information. Our training set for the joint entropy estimation of categories $k, k'$ consists of pairs that we find via bipartite matching [66] between the object instances of both classes, where the edge weights are given by Euclidean distance. We collect all MI estimates in a matrix $\mathbf{M} = \left(\widehat{\mathrm{MI}}(L_{k1}, L_{k'1})\right)_{kk'} \in \mathbb{R}^{K \times K}$. Note that the entries of $\mathbf{M}$ consist of mutual informations between bounding box parameterizations, which are different from the closed-form mutual informations in Eq. (5.9) for the parametric occurrence graph. For instance, even though 'dining table' and 'fork' frequently co-occur, their relative locations are largely unstructured, which results in a high occurrence MI and low location MI. See Figure 5.4 for an overview.

Instead of learning a full location model over all possible combinations of categories and instances, we construct a dynamic location topology that is optimal with respect to the available occurrence information. Specifically, given an occurrence vector $\mathbf{o} = (o_k)_{k=1}^{K}$, we split the location variables into an active set $\mathbf{L}_A$ and an inactive set $\mathbf{L}_I := \mathbf{L} \backslash \mathbf{L}_A$. For each category $k$, we add the $o_k$ location variables $\{L_{k1}, \ldots, L_{ko_k}\}$ to the active set, *i.e.*, if the occurrence model identifies three persons, we activate the first three person location variables. We can now efficiently

$$\mathbf{o} = \left(\begin{array}{ccccccccccccc} & & \text{table} & & & & \text{person} & & & & \text{bike} & & \\ \cdots & 0 & 1 & 0 & \cdots & 0 & 4 & 0 & 0 & 0 & 3 & 0 & 0 \end{array}\right)$$

**(a)** Occurrence vector $\mathbf{O} = \mathbf{o}$.



**(b)** Full MI table $\mathbf{M}$. **(c)** MST construction. **(d)** Dynamic topology.

**Figure 5.5: Dynamic Location Topology.** **(a)** The occurrence vector $\mathbf{o}$ contains quantities for all $K$ categories. ($\mathbf{b}, \mathbf{c}$) After locating the MI values between classes with non-zero occurrences in $\mathbf{M}$, we compute an MST over the active variables $\mathbf{L}_A$. **(d)** The active variables $\mathbf{L}_A$ form a dynamic location topology $G_{\mathbf{L}|\mathbf{o}}$.

construct a non-parametric, occurrence-dependent topology $G_{\mathbf{L}|\mathbf{o}}$ by computing a maximum spanning tree over $\mathbf{L}_A$ only. The required MI values are readily available in $\mathbf{M}$. Note that this is not equivalent to computing a maximum spanning tree of $\mathbf{M}$ (Fig. 5.4b) and connecting the active location variables based on such a tree's edges. Since the occurrence network is image-conditioned, the dynamic location topology depends on the image as well. Fig. 5.5 shows an illustration of this process.

### Dynamic Local Models

Instead of fitting unimodal Gaussian distributions to the active location variables, which, despite their limitations, are still a quasi-standard for estimating continuous Bayesian networks [25], we account for the rich and multimodal dependencies in natural scenes by adapting the non-parametric Bayesian network of chapter 3 to our problem.

**Structured Non-parametric Density Estimation.** As a reminder, a non-parametric Bayesian network describes a factorization of a density into a product of conditional kernel density estimates, *i.e.*, it is a special case of Eq. (5.2) in which the local models are continuous and non-parametric. To account for the non-uniform structure of location likelihoods and to pave the way for kernel reduction approaches (section 5.3), we use a slighlty more general version of a conditional KDE than previously.

More specifically, given a training set $\mathcal{D}_{\mathbf{x}} = (\mathbf{x}^{(i)})_{i=1}^{N} = (x_j^{(i)})_{ji} \in \mathbb{R}^{d \times N}$ and an ensemble of bandwidth matrices $\mathcal{H} = \{\mathbf{H}^{(i)}\}_{i=1}^{N}$, we extend our original definition of an unconditional kernel density estimate to

$$p_{\mathcal{H}}^{\mathcal{D}_{\mathbf{x}}}(\mathbf{x}) = \sum_{i=1}^{N} \frac{w_i}{|\mathbf{H}^{(i)}|} \cdot \kappa_d\Big(\big(\mathbf{H}^{(i)}\big)^{-1}\big(\mathbf{x} - \mathbf{x}^{(i)}\big)\Big), \tag{5.13}$$

where $\kappa_d$ is a $d$-dimensional kernel and $\mathbf{w} = (w_i)_{i=1}^{N}$ is a non-negative weight vector such that $\|\mathbf{w}\|_1 = 1$. In case of an isotropic Gaussian kernel $\kappa_d = \mathcal{N}(\mathbf{0}, \mathbf{I}_d)$, we can interpret Eq. (5.13) as a Gaussian mixture model with $\mathbf{w}$-weighted components. The $i$-th Gaussian is then centered at $\mathbf{x}^{(i)}$ and has covariance $(\mathbf{H}^{(i)})^2$. As before, we can use such a KDE as a local model by conditioning Eq. (5.13) on an observed value $\mathbf{y}$,

$$p_{\mathcal{H}}^{\mathcal{D}_{(\mathbf{x},\mathbf{y})}}(\mathbf{x} \mid \mathbf{y}) = \sum_{i=1}^{N} w_i(\mathbf{y}) \cdot \mathcal{N}\Big(\mathbf{x} \,\Big|\, \mu_{|\mathbf{y}}^{(i)}, \mathbf{\Sigma}_{|\mathbf{y}}^{(i)}\Big), \tag{5.14}$$

where $\mu_{|\mathbf{y}}^{(i)}, \mathbf{\Sigma}_{|\mathbf{y}}^{(i)}$ are the mean and covariance of the $i$-th Gaussian component in the joint KDE $p_{\mathcal{H}}^{\mathcal{D}_{(\mathbf{x},\mathbf{y})}}(\mathbf{x}, \mathbf{y})$ conditioned on $\mathbf{y}$ and

$$w_i(\mathbf{y}) \propto w_i \cdot \mathcal{N}\Big(\mathbf{y} \,\Big|\, \mathbf{y}^{(i)}, (\mathbf{H}_{|\mathbf{yy}}^{(i)})^2\Big) \tag{5.15}$$

are the conditional weights ($\mathbf{H}_{|\mathbf{yy}}^{(i)}$ denotes the $\mathbf{y}$-part of $\mathbf{H}^{(i)}$).

With these prerequisites at hand, we are ready to describe the factors of Eq. (5.12) in more detail. To keep the notation uncluttered, we fix a node $L \in \mathbf{L}_A$ with category $k$ and its parent node $L' := \mathrm{pa}_{G_{\mathbf{L}|\mathbf{o}}}(L) \in \mathbf{L}_A$ with category $k'$.

**Local Priors.** Due to its non-parametric nature, a training set specifies Eq. (5.14) up to the weights and bandwidth matrices. We create a training set $\mathcal{D}_{(l,l')}$ by accumulating, over all images but for each image separately, all pairs of instances belonging to categories $k$ and $k'$. Using $\mathcal{D}_{(l,l')}$, we set the prior term in Eq. (5.12) to[4]

$$p_L(l \mid L' = l') = p_{\mathcal{H}}^{\mathcal{D}_{(l,l')}}(l \mid L' = l'). \tag{5.16}$$

---

[4] In practice, one should also consider conditioning Eq. (5.16) on the space of bounding boxes to ensure a finite support region.

If there is no reason to favor one training point over the other, we use a uniform weight vector $\mathbf{w}$ but note that the conditional weights in Eq. (5.16) are nonetheless non-uniform due to the reweighting operation in Eq. (5.15). Furthermore, we set $\mathbf{H}^{(i)} := \mathbf{H}$ and estimate the shared bandwidth matrix $\mathbf{H}$ according to Scott's rule [141], that is, proportional to the square root of the sample covariance matrix.

**Local Likelihoods.** The density $p_L(l \mid I)$ at a particular point should reflect the likelihood that the corresponding bounding box contains an object of category $k$ in image $I$. We obtain a training set representing this density by following a path similar to Eq. (5.7), the only difference being that we account for the sample space $\mathcal{B}$ of $L$ by extracting and scoring bounding box features instead of global image features. Since a sliding window evaluation of the entire space is computationally intractable, we use selective search proposals [158] to cover high-density regions as efficiently as possible. This allows us to approximate the likelihood over the image with a kernel density estimate with relatively few components centered on likely object locations.

Formally, we extract a fixed number $P$ of object proposals $\mathcal{D}_{(\boldsymbol{\zeta},\varnothing)} = \{\boldsymbol{\zeta}^{(i)} \in \mathcal{B}\}_{i=1}^{P}$ using selective search, compute features $\phi_\ell(\boldsymbol{\zeta}^{(i)}) \in \Phi$ for them, and classify each feature with a pre-trained and calibrated multi-class classifier $c_\ell : \Phi \to \mathbb{R}^K$. This results in a set of $P$ normalized score vectors $\mathbf{s}^{(i)} := [c_\ell \circ \phi_\ell](\boldsymbol{\zeta}^{(i)}) \in \mathbb{R}^K$ that we collect in a score matrix $\mathbf{S} = (\mathbf{s}^{(i)})_{i=1}^{P} \in \mathbb{R}^{K \times P}$, *i.e.*, $s_k^{(i)}$ is the probability that $\boldsymbol{\zeta}^{(i)}$ contains an object of class $k$. To approximate the likelihood over the image, we consider each object proposal $\boldsymbol{\zeta}^{(i)}$ as a training point with weight $w_i := s_k^{(i)}$ and bandwidth $\mathbf{H}^{(i)} := \text{Area}(\boldsymbol{\zeta}^{(i)})\mathbf{H}$, where $\text{Area}(\bullet)$ denotes normalized bounding box area and $\mathbf{H}$ is set as above, leading to

$$p_L(l \mid I) = p_{\mathcal{H}}^{\mathcal{D}_{(\boldsymbol{\zeta},\varnothing)}}(l \mid \varnothing). \tag{5.17}$$

## 5.3 Inference

Given a trained scene prior hierarchy $\mathcal{S}$, we obtain a prediction by conditioning $\mathcal{S}$ on a test image $I$ and running level-dependent inference schemes.

**Parametric Sum-Product Inference in $\mathcal{B}_{\mathbf{O}}$.** Due to our tree-structured topology and the fact that most images do *not* contain a given object class, many conditional distributions in $\mathcal{B}_{\mathbf{O}}$ are peaked at 0. As a consequence, we would need strong image evidence to avoid an empty scene as MAP solution. An approach that proved to perform better in practice is therefore a run of standard sum-product inference followed by a computation of the expected marginals for each node $O_k$.

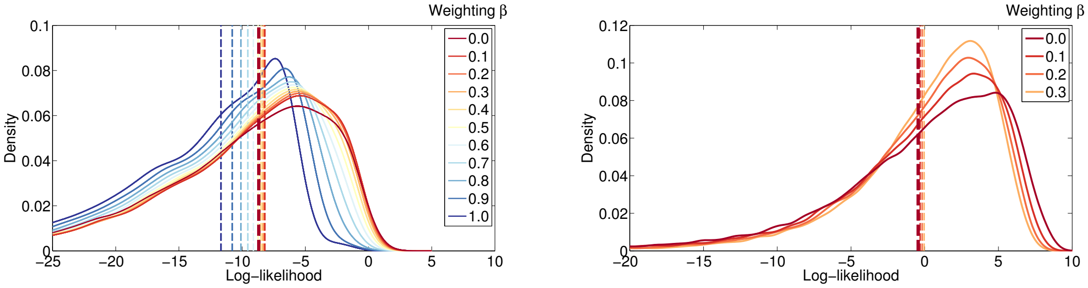
**Non-parametric Max-Product Inference in $\mathcal{B}_{\mathbf{L}}$.** The true generating distribution $p^*_{\mathbf{L}}$ is inherently multimodal, a fact that not only causes the need for non-parametric local models but also drives the way we perform inference. In particular, we employ a recent version of non-parametric max-product BP [112] that is designed to preserve modes, which makes it ideally suited for our purpose. Note that our generative approach allows for very effective proposals in the form of likelihood samples from $p_{L_{kj}}(l_{kj} \mid I)$. In all our experiments, we use 40 particles and run the algorithm for 40 iterations at a smoothing temperature of 10.

**Speed-Accuracy Trade-Off.** MAP inference in non-parametric Bayesian networks is computationally expensive, because particle-based inference schemes need to evaluate the complex model potentials (*i.e.*, the conditional kernel density estimates) millions of times. To perform those evaluations as efficiently as possible, we could use an approach similar to section 3.4. Another possibility, which we employ in this chapter, is to approximate the original KDE with a mixture consisting of fewer components. A popular way to achieve this is to successively replace two similar kernels by their moment-preserving merge, either by minimizing the change in intra-component variance (Eq. 2.19) [134] or by minimizing an upper bound of the KL-discrimination before and after the merge [130]. The quality of such an approximation can be ensured by keeping track of the MSE between the reduced and reference density.

## 5.4 Experiments

Our approach is made possible by the recent availability of the large-scale Microsoft COCO dataset [99], which contains, inter alia, bounding box annotations for 80 semantic object classes of common everyday objects. The full dataset comprises a total of $9.1 \times 10^5$ instances in $1.2 \times 10^5$ images, which have been officially split into a training set and a validation set at a ratio of 2 : 1. COCO exhibits a large variability with respect to both annotations per object class ($1.1 \times 10^4 \pm 3.0 \times 10^4$) and per image ($7.39 \pm 7.40$), which poses a serious challenge and requires a flexible approach.

We use COCO to illustrate the benefits of the proposed non-parametric scene prior hierarchy in two large-scale experiments: (1) Expected log-likelihoods and (2) precision in object quantification and detection tasks. All experiments are based on the same universal model that we train on the COCO training set and test on a subset of the COCO validation set proposed by the fast-RCNN author [54]. We make use of recent advances in the area of deep architectures and instantiate $\phi_o, \phi_\ell$



**(a)** Occurrence Network. **(b)** Location Network.

**Figure 5.6: Distribution of Test Log-likelihoods.** We show the distribution of test log-likelihoods in **(a)** the occurrence network $\mathcal{B}_{\mathbf{O}}$ and **(b)** the location network $\mathcal{B}_{\mathbf{L}}$. Models with moderate weights up to $\beta = 0.3$ achieve a better expected test log-likelihood (vertical lines) than the baseline with $\beta = 0.0$.

as rectified FC-7 features from a state-of-the-art CNN [54]. The classifiers $c_o^{(k)}, c_\ell$ are obtained using logistic regression [40] and the CNN's softmax layer, respectively. In case of the location network, we will therefore sometimes refer to the pure image likelihoods Eq. 5.17 ($\beta = 0.0$) as *generative fast-RCNN (gfRCNN)*.

## 5.4.1 Expected Log-likelihood

A standard way to compare $\theta$-parameterized density models $\widehat{p}^{(\theta)}$ is to measure their closeness in terms of KL-divergence to the unknown generating distribution $p^*$. In the specific case of our scene prior hierarchy (Eq. (5.4)), this leads to

$$\mathrm{KL}\left(p^*_{\mathbf{OL}} \,\middle\|\, \widehat{p}^{(\beta)}_{\mathbf{OL}}\right) = -\mathbb{E}_{p^*_{\mathbf{OL}}}\left[\log\, \widehat{p}^{(\beta)}_{\mathbf{OL}}(\mathbf{O}, \mathbf{L})\right] + \text{const.} \tag{5.18}$$

Since we do not have direct access to $p^*_{\mathbf{OL}}$, we estimate

$$\widehat{\mathbb{E}}_{p^*_{\mathbf{OL}}}\left[\log\, \widehat{p}^{(\beta)}_{\mathbf{OL}}(\mathbf{O}, \mathbf{L})\right] = N^{-1} \sum_{i=1}^{N} \log\, \widehat{p}^{(\beta)}_{\mathbf{OL}}(\mathbf{o}^{(i)}, \mathbf{l}^{(i)}), \tag{5.19}$$

which is just the negative empirical risk with respect to a log-loss. In our experiments, we evaluate Eq. (5.19) for a number of different weightings between $\beta = 0.0$ (no prior) and $\beta = 1.0$ (no image evidence). Figs. 5.6a and 5.6b summarize this quantitative evaluation: Moderate contributions of the prior up to a weighting of $\beta = 0.3$ perform better than the baseline without a prior in both layers. In particular, we accumulate substantially more mass around the modes of the likelihoods at approximately

$-5$ nats ($\mathcal{B}_{\mathbf{O}}$) and $+2.5$ nats ($\mathcal{B}_{\mathbf{L}}$), suggesting that our scene prior hierarchy pushes the underlying CNN architectures for object quantification and detection closer to the ground truth density. For higher values of $\beta$, the model moves towards a generative, image-independent prior and it is no surprise that the performance falls below that of a baseline incorporating more image evidence.

### 5.4.2 Occurrence and Location Accuracy

In addition to the theoretical validation in the preceding section, we also analyze the predictive performance gains when using our scene prior. We look at both layers independently and compare their ground truth occurrences and locations to our model's expected marginals and MAP predictions.

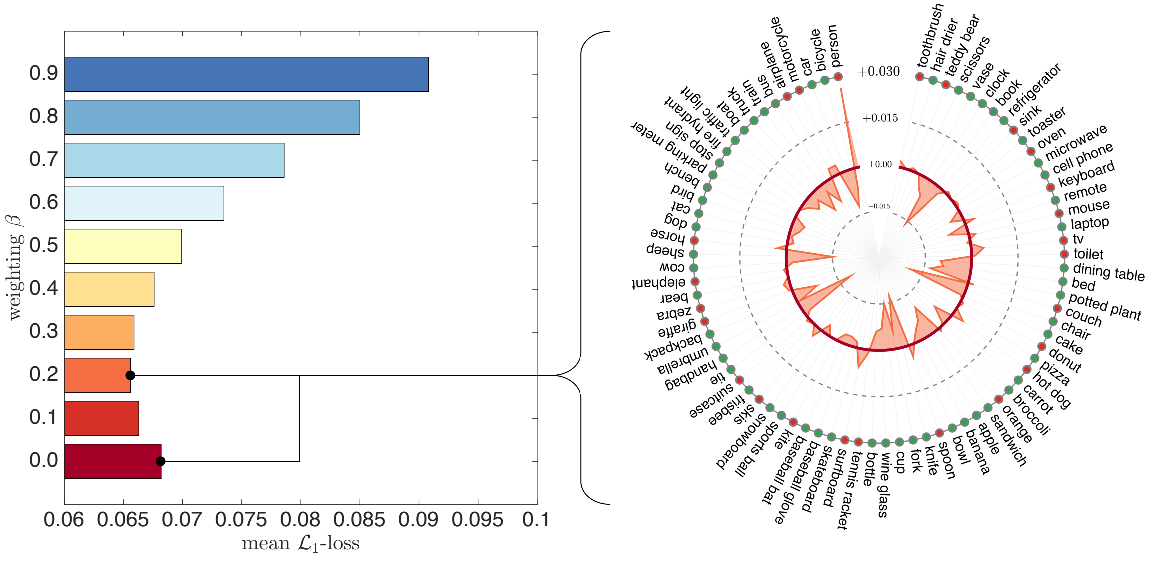
#### Occurrence network $\mathcal{B}_{\mathbf{O}}$.

For evaluation purposes, we employ a standard $\mathcal{L}_1$-loss, $\mathcal{L}_1(\mathbf{o}^{(i)}, \widehat{\mathbf{o}}^{(i)}) = \|\mathbf{o}^{(i)} - \widehat{\mathbf{o}}^{(i)}\|_1$, and report the expected test loss (ETL) across all test images and categories in Fig. 5.7a. The results confirm our previous findings: We achieve a minimal ETL of 0.0656 at $\beta = 0.2$, after which the performance decreases as expected and falls below the baseline score of 0.0682 at $\beta = 0.5$ due to the lack of strong image evidence. In Fig. 5.7b, we take a closer look at the performance gains across categories for the optimal weighting: Using our prior with $\beta = 0.2$ improves the performance in 56/80 categories (green dots). The performance decline in the remaining categories (red dots) is small, with the 'person' category being the only exception. We attribute this observation to the high variance (15.98) and high average number (4.04) of persons in the COCO images in which this class is present.

#### Location network $\mathcal{B}_{\mathbf{L}}$.

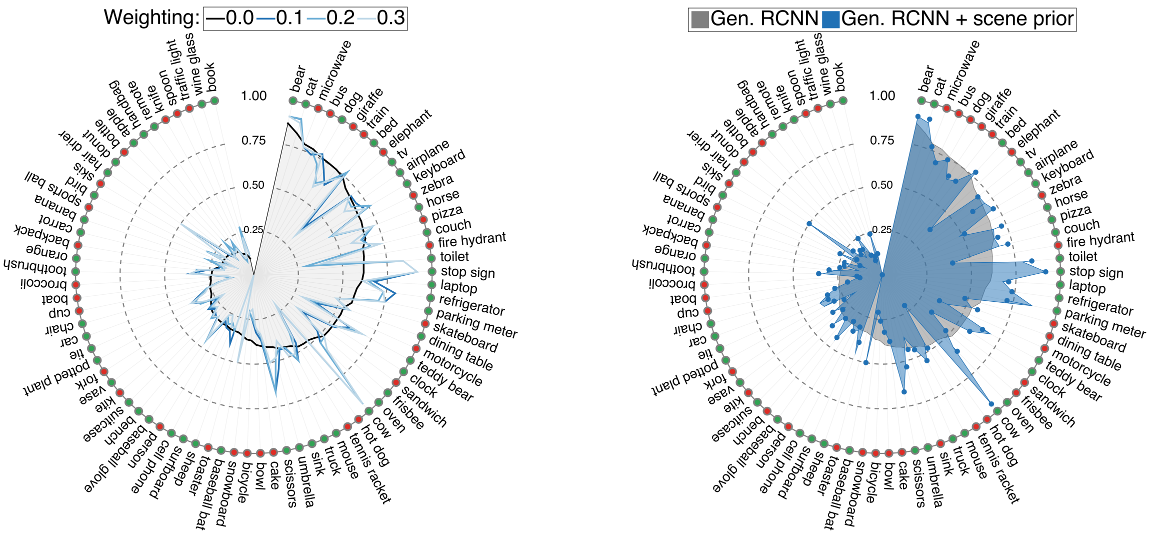
We report average precision (AP) at an intersection-over-union threshold of 0.5. Guided by our findings in the occurrence network, we evaluate this metric in the critical range between $\beta = 0.0 - 0.3$. Fig. 5.7c shows our results across COCO categories. The average improvement over the priorless baseline (mean AP: 41.85%) varies with $\beta$ and can be up to 3.1% (max. mean AP: 43.13%). Overall, we achieve performance benefits in 49/80 categories.

The ability to take advantage of the prior naturally varies from category to category, depending on the predictive power of a category's likelihoods and the strength of the contextual information provided by co-occurring objects. In practice, it can



**(a) Occurrence network.** Mean error on COCO for $\beta = 0.0 - 0.9$.

**(b) Occurrence network.** Mean error on COCO for $\beta = 0.2$ relative to $\beta = 0.0$.



**(c) Location network.** Average precision across COCO categories (*global* weighting).

**(d) Location network.** Average precision across COCO categories (*local* weighting).

**Figure 5.7: Quantitative Evaluation on COCO. (a)** Expected $\mathcal{L}_1$-loss in the occurrence network over all COCO categories and test images. Moderate weightings of the prior up to $\beta = 0.2$ improve the overall performance. **(b)** Category-by-category comparison of a prior with $\beta = 0.2$ to a system without a prior. Negative values indicate a relative improvement over the priorless baseline. **(c)** Average precision in the location network across COCO categories. **(d)** We report the same content as in (c), but using a custom weight $\beta_k$ per category. (**note**) Categories are ordered according to their baseline precision. A green dot indicates categories for which a positive weight (Fig. 5.7c)/the variable weight (Fig. 5.7d) outperforms the priorless baseline. A blue dot indicates an upper bound on the average precision if the optimal variable weights were known. Lower is better in (a-b), higher is better in (c-d).

therefore be advisable to use a custom weight per category. Simple ways of achieving this include selecting the weights based on the edge weights in $G_{\mathbf{L}|\mathbf{o}}$ or cross-validation. The next section discusses variable weights in some more detail.

**Variable Weights.** The scalar weight $\beta \in [0, 1]$ controls and balances the contributions of prior and likelihood (see Eqs. (5.6) and (5.12)); the larger $\beta$ the higher the contribution of the prior. However, with a single parameter $\beta$, the improvement due to our scene prior hierarchy varies from object category to object category, as mentioned in the previous section. This observation motivates choosing an individual weight $\beta_k$ for each object category: For object categories $k$ that are visually discriminant and have peaked likelihoods, $\beta_k$ should intuitively be smaller. On the other hand, for object categories $k$ that often appear small, non-discriminant, or occluded but have high location MI with other object categories that are more discriminant, $\beta_k$ should intuitively be larger.

We illustrate the potential that lies in variable weights by considering an oracle setting: We employ a greedy two-stage approach that selects an optimal local weight $\beta_k$ for each category $k$ by running the standard model with different global weights. More specifically, we fix a set $\boldsymbol{\beta} = \{\beta^{(i)}\}_i$ of global weights and select, for each category $k$ independently, the weight $\beta_k$ that maximizes the average precision of category $k$ over all $\beta \in \boldsymbol{\beta}$. Subsequently, we rerun max-product inference in the location network with those optimal values. Using this weighting strategy with the results shown in Fig. 5.7c, we obtain the precision scores shown in Fig. 5.7d. Separate weights for each object category can increase the baseline performance by up to 3.7% (mean AP: 43.40%). It is important to note that this analysis is a theoretical *upper bound*. In practice, we do not know those optimal weights and have to estimate them using cross-validation, in which case the performance gain is 2.4% (mean AP: 42.86%).

**Comparison to Alternative Topologies and Local Models.** The previous paragraphs have shown that our scene prior can improve the performance of a generative approach based on fast-RCNN predictions. We will now show that this could not have been accomplished with less flexible topologies and/or local models. In particular, we compare our approach to scene priors with an edgeless topology ('independent') and/or normally distributed local models ('Gaussian'). For all 4 combinations, we run MAP inference with 4 global ($\beta = 0.0 - 0.3$) and 2 local (cross-validation and oracle) weighting schemes. Table 5.1 summarizes our results on the test set. With global weights, an independent Gaussian prior performs worst, followed by a tree-structured Gaussian prior and an independent KDE prior. Our proposed combination of a tree-structured topology and non-parametric local models (last row)

| | Model | Global Weighting $\beta$ | | | | Local Weighting $\beta_k$ | |
|---|---|---|---|---|---|---|---|
| | | 0.0 | 0.1 | 0.2 | 0.3 | CV | Oracle |
| | Parametric Network | 41.85 | 40.95 | 39.42 | 37.37 | 41.62 | 41.60 |
| | (Gaussian, independent) | (±0.00%) | (−2.15%) | (−5.81%) | (−10.70%) | (−0.55%) | (−0.60%) |
| | Parametric Network | 41.85 | 41.21 | 39.82 | 37.03 | 42.15 | 42.21 |
| | (Gaussian, tree-structured) | (±0.00%) | (−1.53%) | (−4.85%) | (−11.52%) | (+0.72%) | (+0.86%) |
| | Non-parametric Network | 41.85 | 41.38 | 40.23 | 38.51 | 41.64 | 42.03 |
| | (KDE, independent) | (±0.00%) | (−1.12%) | (−3.87%) | (−7.98%) | (−0.50%) | (+0.43%) |
| (**ours**) | Non-parametric Network | 41.85 | **43.13** | 42.31 | 41.48 | **42.86** | 43.40 |
| | (KDE, tree-structured) | (±0.00%) | (**+3.06%**) | (+1.10%) | (−0.88%) | (**+2.41%**) | (+3.70%) |

**Table 5.1: Comparison to Baseline Priors.** Mean AP for two types of topologies (independent, tree-structured) and two types of local models (parametric Gaussian, non-parametric KDE). We run MAP inference on the test set for all 4 possible combinations and 6 different weighting schemes, 4 global ones ($\beta = 0.0 - 0.3$) and 2 local ones (cross-validation, oracle). The last row (**ours**) is shown in more detail in Figs. 5.7c and 5.7d.

is the only setup which is able to improve the baseline precision. The other priors are inferior due to their lack of context information (independent topology) or their limited expressiveness (Gaussian local models). With local weights, tree-structured topologies have an edge over the independent priors, but the non-parametric approach is again superior to the parametric model.

## 5.5 Qualitative Results

We will now support our quantitative evaluation with some qualitative impressions from our scene prior hierarchy.

### 5.5.1 Scene Layout Synthesis

One of the advantages of our fully generative prior is its ability to synthesize plausible scene layouts by drawing samples from the unconditional joint distribution $p_{\mathbf{OL}}(\mathbf{o}, \mathbf{l} \mid \varnothing)$. In order to give an impression of sample quality, we show some unconditional samples that we draw, layer-by-layer, from the hierarchy. In particular, we draw an unconditional sample $\mathbf{o}$ from the occurrence distribution $p_{\mathbf{O}}(\mathbf{o} \mid \varnothing)$, generate a dynamic location topology $G_{\mathbf{L}|\mathbf{o}}$ based on $\mathbf{o}$, and draw an unconditional (in the sense of 'not image-conditioned') sample $\mathbf{l}$ from $p_{\mathbf{L}}(\mathbf{l} \mid \mathbf{O} = \mathbf{o}, \varnothing)$.

### 5.5.2 Occurrence Network Predictions

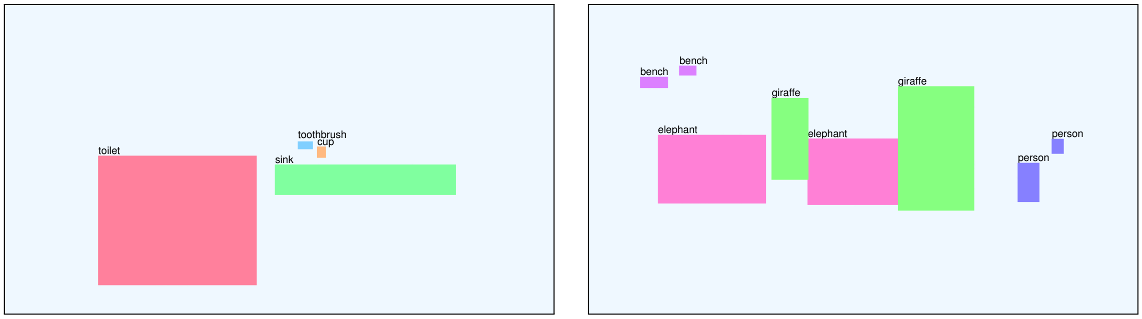
Given a trained scene prior hierarchy $\mathcal{S}$ and a test image $I$, we run parametric sum-product inference in $\mathcal{B}_{\mathbf{O}}$ to obtain all $K$ marginal distributions $p_{O_k}(o_k \mid I)$ and make a prediction $\mathbf{o} = (o_k)_{k=1}^{K}$ with $o_k = \mathbb{E}[O_k \mid I]$. Fig. 5.9 illustrates our model's ability to infer object occurrences even in complex images containing several categories (a) and to 'count' multiple instances of the same class (b,c). Also note how the strong image evidence for 'doughnut' (b) increases the likelihood for other foods (*e.g.*, 'pizza','cake') and a 'dining table', but the lack of further image evidence keeps these classes below the radar. The difficult class 'baseball glove' is detected successfully due to the presence of 'baseball bat' (d).

### 5.5.3 Location Network Predictions

Given a trained scene prior hierarchy $\mathcal{S}$ and a test image $I$, we generate a dynamic network topology according to the procedure introduced in section 5.2.2 and run non-parametric max-product inference [112][5] in $\mathcal{B}_{\mathbf{L}}$ to obtain a prediction for all active location variables $\mathbf{L}_A$. Fig. 5.10 shows some of those predictions.

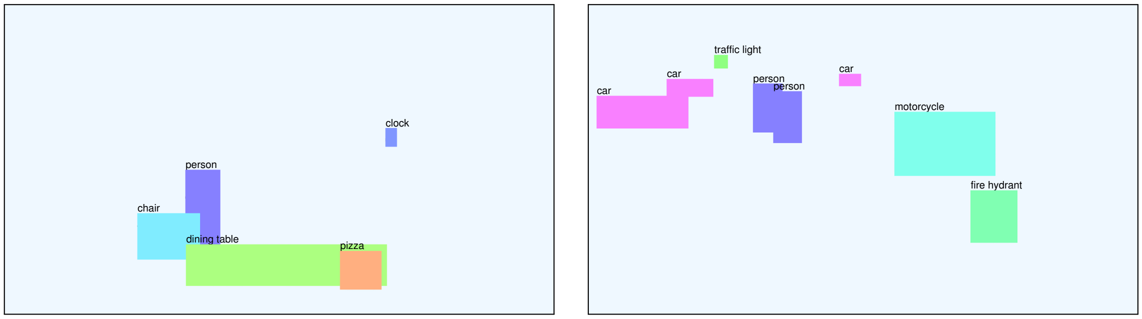
---

[5] Inference parameters: smoothing temperature = 10, particles = 40, iterations = 40.



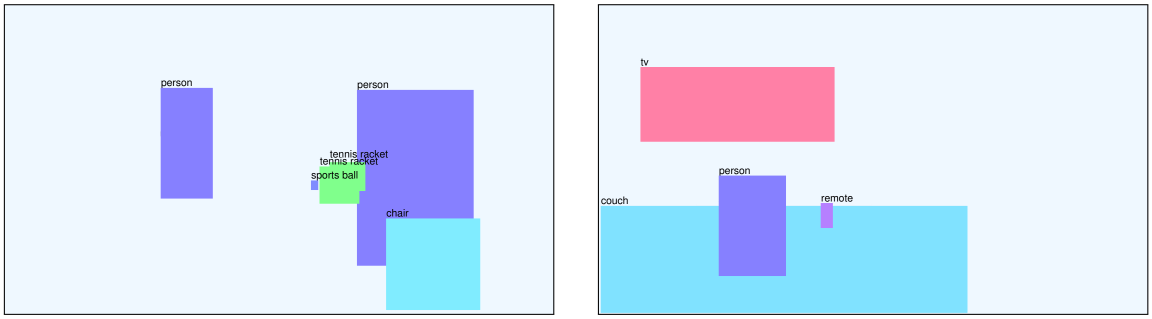
**(a)** Bathroom scene.

**(b)** Animal scene.



**(c)** Dining room scene.

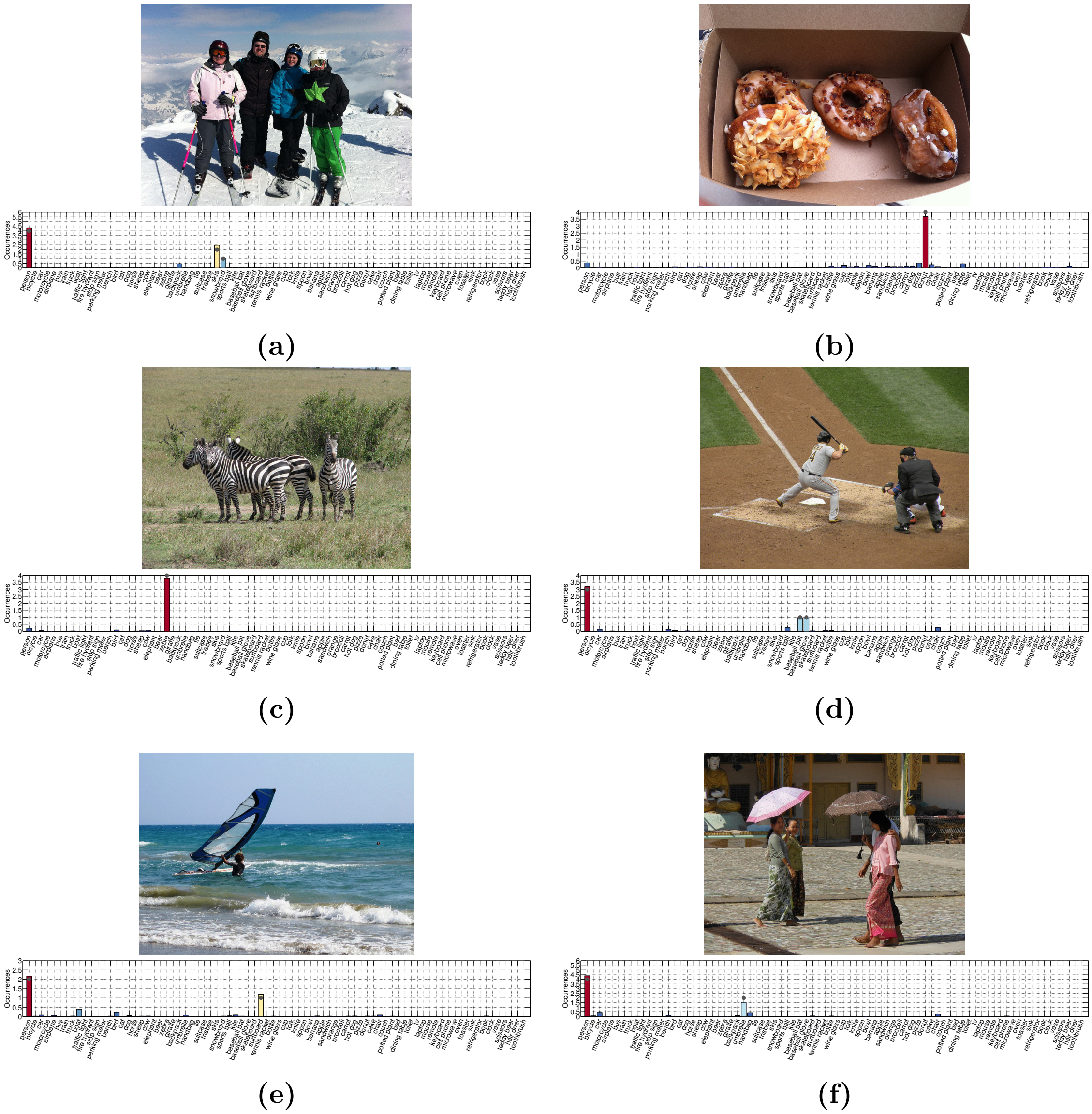
**(d)** Road scene.



**(e)** Sports scene.

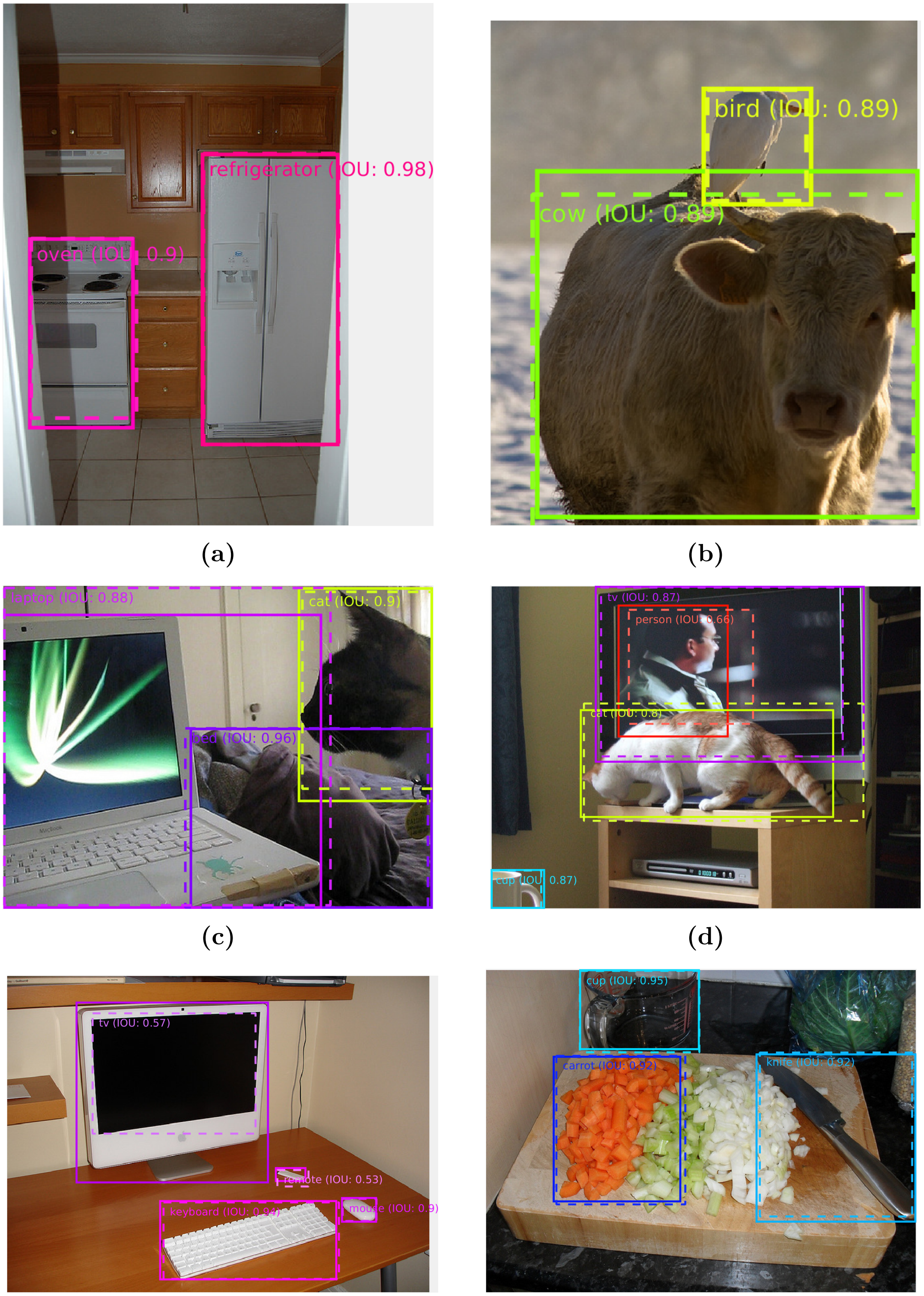
**(f)** Living room scene.

**Figure 5.8: Scene Layout Synthesis.** 6 unconditional samples from our prior. All scene layouts were drawn from the same universal model, illustrating that our approach can cover a wide variety of domains, produce sensible combinations of objects that are specific to a particular environment, and place them in appropriate positions (pizza on dining table; toothbrush next to cup; remote on sofa). In scene (d), we may even perceive a vanishing point. The naming of the scenes was done manually for illustrative purposes.



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

**Figure 5.9: Occurrence Network Predictions.** 6 test images together with the expected marginals of all 80 COCO categories (in practice, we would round these to the nearest integer). Filled gray circles mark ground truth values of all categories with non-zero occurrences. Note how the presence of a category with strong image evidence supports the presence of semantically related categories with less image evidence.



**(a)** **(b)**

**(c)** **(d)**

**(e)** **(f)**

**Figure 5.10: Location Network Predictions.** 6 test images together with their ground truth annotations (solid boxes), our predictions (dashed boxes), and the corresponding intersection-over-union (IOU) scores. Hue encodes categories, saturation encodes accuracy.

# Chapter 6

# Non-parametric Semantic Video Segmentation

Semantic context can be extrinsic (‘allocontext’), as in the previous chapter, or intrinsic (‘autocontext’), which we will investigate in the present chapter to model the foreground shape of people in video sequences. We employ a three-layered approach in which each layer models a different type of contextual information: Spatial context in the first layer, static autocontext in the second layer, and dynamic autocontext in the third layer. Each layer contains the same type of non-parametric base classifier but is trained on a dedicated set of features that may comprise information derived in previous layers, possibly at previous points in time, such as their labeling or their confidence.

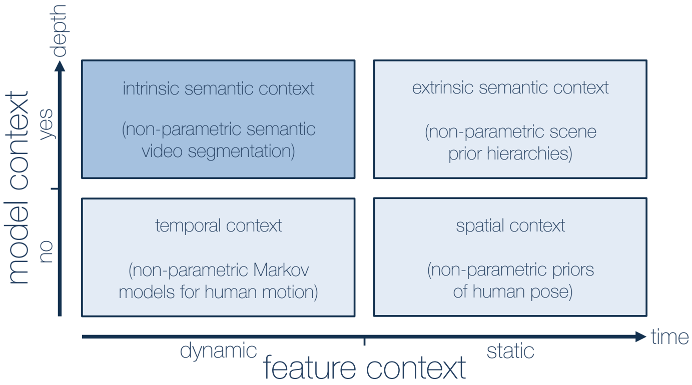

depth
model context
yes
no
intrinsic semantic context
(non-parametric semantic video segmentation)
extrinsic semantic context
(non-parametric scene prior hierarchies)
temporal context
(non-parametric Markov models for human motion)
spatial context
(non-parametric priors of human pose)
dynamic
static
time
feature context

**Figure 6.1: Key Message: Intrinsic Semantic Context.** The output of a non-parametric base classifier for a previous frame can be used as an additional feature in a semantic segmentation task to train a non-parametric base classifier for the present frame.

## 6.1 Introduction

The treatment of objects in computer vision comprises a coarse-to-fine hierarchy of principled tasks: categorization → quantification → detection → segmentation. We already encountered this hierarchy in the previous chapter, where we used coarse occurrence information to prime a finer location model. Although the individual links in this hierarchical chain are all useful in their own right, they should generally be understood as adjuvants for subsequent high-level tasks. In this chapter, we focus on segmentation, which is on the finest hierarchical level and therefore an effective tool for ensuing tasks requiring the actual outline of an object, such as the identification of object-object interactions (*e.g.*, between two people), the recognition of an object's state (*e.g.*, a person's activity), or the reconstruction of an object's 3D shape.

We are particularly interested in the semantic segmentation of video sequences, a task with a rich history that has been approached by both static models, which operate independently on single frames of a video sequence, and dynamic models, which condition their actions on other frames. While static models can exploit spatial dependencies, dynamic models have access to higher-order motion information. In contrast to feature context, where dynamic context in the form of temporal dependencies between pixels is well-studied, model context has been limited to static types of intrinsic semantic context between labels. We are therefore proposing a general-purpose *dynamic autocontext* model for semantic video segmentation, turning the three-dimensional spatio-temporal grid of common dynamic models into a four-dimensional tensor grid encoding spatio-temporal as well as semantic dependencies.

The integration of a temporal component for semantic cues into a segmentation model is beneficial from an information-theoretic perspective but challenging from a modeling point of view, especially with regard to memory, runtime, and inference:

**Memory.** Dynamic approaches are often non-causal offline methods, *i.e.*, they require that the entire video is available at processing time. This not only limits the maximum number of processable video frames due to memory constraints but also excludes their application from all real-world systems in which an online analysis of a video stream is necessary (*e.g.*, surveillance cameras).

**Runtime.** A related challenge in the online setting is the requirement to process the incoming video with low latency. This is especially relevant for high-level tasks, where segmentation serves as a preprocessing step for subsequent components, which come with computational costs of their own.

**Inference.** Many approaches for semantic video segmentation operate on the level of individual pixels or small superpixels, so that the corresponding graphical models can easily contain millions of random variables. Ensuring adequate conditional independencies is therefore key to guaranteeing efficient training and inference.

### 6.1.1 Contribution

In this chapter, we propose a causal extension to static autocontext [156]. Our contribution is a dynamic context model for semantic video segmentation which addresses the aforementioned challenges and has the following desirable properties:

1. General: We do not rely on application-specific features or priors.
2. Comprehensive: We integrate intrinsic semantic context in space and time.
3. Efficient: Our CPU implementation runs at multiple frames per second.
4. Effective: We achieve decent performance on multiple benchmark datasets.

### 6.1.2 Related Work

In recent years, semantic video segmentation has been tackled from several different perspectives. The available methods can be roughly classified by the type of input data they rely on, whether they fulfil their task in isolation or jointly with other tasks, and their processing time for unseen test frames.

Unsupervised approaches have the appealing property that they can operate on unlabeled RGB data. This is a big advantage considering the high cost for annotating even short video sequences. A clear downside is the unavailability of a likelihood function as an objective evaluation criterion. Furthermore, the semantic regions resulting from such approaches can be controlled only implicitly, *e.g.*, based on color or motion: Grundmann *et al.* [59] build upon the work of Felzenszwalb and Huttenlocher [43] and construct a hierarchy of 3D spatio-temporal graphs with varying granularity by successively merging graph regions based on their Lab histogram difference. Corso *et al.* [171] pursue a conceptually similar method but focus on a streaming mode consisting of hierarchies for non-overlapping chunks of $k$ frames. Region merging as in [43, 59] can be allowed or forbidden based on a semi-supervised evaluation scheme depending on hierarchical layers in the previous and current frame. Motion-based approaches include a popular method due to Brox and Malik [17], who use a flow-based tracker to generate dense point trajectories that can be clustered using spectral clustering [107]. Recently, Papazoglou *et al.* [113]

proposed an efficient approach to general object segmentation by first computing a rough inside-outside-map of the foreground object based on motion and then refining the initial estimate by enforcing temporal smoothness and building a dynamic appearance model. Schlesinger [137] proposed an unsupervised real-time video segmentation method for tracking foreground object segmentation masks. The model includes a spatial prior which needs to be initialized and is of a restrictive form in order to permit real-time inference.

Conditional random fields (CRFs) are flexible models in the sense that they provide a natural framework to process different types of input data (*e.g.*, RGB, depth and disparity), allow the inclusion of arbitrary features and can be optimized jointly for multiple tasks (*e.g.*, pose, segmentation and depth): The detector-CRF [89] unifies detection and segmentation in a common and principled framework. Vineet *et al.* [161] use an instance of this approach for human segmentation in which the detector potentials are supplied with body and face detections. In addition to that, video potentials in form of a Potts model [119, 170] on 3D mean-shift [27, 50] regions enforce interframe consistency and histogram matching ensures consistency over scale and pose. Alahari *et al.* [2] consider the task of jointly optimizing for segmentation, pose and disparity of humans in 3D movies. Their CRF contains spatial as well as temporal pairwise terms and is solved in a three-step approach: After running disparity-augmented versions of the person detector in [42] and the pose estimation proposed in [173], they fix the pose and first optimize for the disparity and finally for the segmentation using $\alpha$-expansion [11]. The obvious drawback of supervised video segmentation approaches is their need for labeled data. At the moment, only very few datasets exist for this task, among them the Buffy dataset [44], the FBMS dataset [111], the J-HMDB dataset [75] and the more recent VSB100 dataset [51].

From a technical point of view, semantic video segmentation has been successfully addressed using spatio-temporal Markov random fields (MRFs, Eq. 2.8). In the early work of Luthon *et al.* [102], the authors use an MRF-prior on a spatio-temporal grid and propose a multi-resolution inference method based on iterative conditional modes [8] in a temporal sliding window. Although the type of dependencies expressible in this model is limited, the authors report an impressive runtime of 13 FPS for $256 \times 256$ images with their custom CPU implementation. More recently, Yin and Collins [176] used a similar model for segmenting moving objects in videos; their key improvements are in terms of inference, where they use loopy belief propagation, and in terms of incorporating a motion likelihood term into their model. While the spatio-temporal MRF model is principled, the dependencies expressible in a rigid spatio-temporal graph are clearly limited. Huang *et al.* [68] address this deficiency by using a flexible graph structure build from local motion information and by incorporating a global shape prior into the model. Although this complicates inference, they suggest an efficient expectation maximization method.

While the above successes demonstrate that spatio-temporal MRFs can be efficient, they are a joint model for the entire video sequence and therefore not suited for a causal estimation of segmentation masks. In contrast, our approach uses a directed model for the temporal dependencies and as such also supports online inference.

## 6.2 Autocontext

Tu [156, 157] proposed a *static autocontext* model for semantic image segmentation. The method works by iteratively improving a semantic labeling using a sequence of *base classifiers* learned from training data. In the first iteration, only the input image is used to predict, for each pixel, a distribution over semantic classes. In the following iterations, each model is provided as input not only the original input image but also the output of earlier models. Later models can therfore condition on intrinsic semantic context ("autocontext") that abstracts away irrelevant variation of the original input image. Conceptually, autocontext is generally applicable in the sense that any probabilistic model can be used as base classifier to predict the class distribution.

Because of its generality, the idea has been improved in many ways and incorporated into a broad range of models and applications. In the case of *entanglement forests* [105], a random forest is used to condition on context which characterizes the set of nodes in a random forest that have been traversed when classifying nearby pixels in the image. This idea has been improved in [83] by conditioning on features derived from smoothed probability maps. The original autocontext model [156] faces the problem of overfitting in later stages; Munoz *et al.* [106] address this problem using *stacking* [169] for their hierarchical model. Autocontext has also been adapted to continuous labeling tasks, for instance in the recent work of Schmidt *et al.* [138], where it was used for image deblurring. Overall, the idea of autocontext has been useful to enhance many basic machine learning models for semantic image labeling.

For video sequences, Burgos-Artizzu *et al.* [19] propose an extension of autocontext that takes temporal information into account. We base our method on this idea but are different in the following two aspects: (1) Burgos-Artizzu *et al.* use a context window centered around the current frame, *i.e.*, their model relies on future observation and works only in the offline setting. Instead, we exploit causal dependencies to enable online inference. (2) Whereas they predict one class label per frame, our model labels each pixel in each frame, integrating both spatial and temporal dependencies between semantic labels into one model.

## 6.3 Dynamic Autocontext

Our dynamic autocontext model incorporates intrinsic semantic context by conditioning on the predictions made by the static autocontext model, both for the current frame as well as the previous frame. At this stage, appearance variation that is not relevant to the semantic segmentation task has already been largely removed by the static autocontext model. As a result, the dynamic autocontext model can learn temporal dependencies between *semantic* labels; it can, for instance, learn an implicit motion model and shape prior between frames, independent of the image appearance. We will show in our experiments that this temporal model improves the predictive performance compared to a static autocontext model with access to the same features.

Next, we present our model and describe its training and inference. We follow a top-down presentation, first giving a high-level overview and then filling in the missing details.

A video stream $\mathbf{x}^{(\bullet)} = (\mathbf{x}^{(t)})_{t=1,\ldots,T}$ with spatial resolution $(w, h) \in \mathbb{N}^2$ and current length $T$ is a temporally ordered sequence of images $\mathbf{x}^{(t)} := (x_j^{(t)})_{j=1,\ldots,w \cdot h}$, where we assume w.l.o.g. that the pixels are enumerated by a linear index $j$. We will frequently drop the frame index $t$ if the reference to a specific point in time is not necessary. The actual representation $P$ of a pixel $x_j^{(t)}$ (*e.g.*, its color space or depth) is not relevant; even sophisticated data structures are possible, as long as the used feature map operates on $P$.

A dynamic autocontext classifier $c^{(\bullet)} = (c^{(k)})_{1 \leqslant k \leqslant 3}$ is a hierarchical model with three sequential layers: Base classification ($k = 1$), static autocontext ($k = 2$), and dynamic autocontext ($k = 3$). At each layer $k$ we predict a semantic label $y_j^{(k)} \in L$ for each pixel $x_j \in P$ in the current frame using a base classifier. We consider this to be any function $c^{(k)} : \Phi \longrightarrow \mathcal{P}$, in which $\Phi := \bigtimes_i \Phi_i$ is a product feature space with associated feature map $\phi : P \longrightarrow \Phi$ and $\mathcal{P}$ is a probability measure on a discrete probability space with sample space $L$. Our prediction is then simply

$$y_j^{(k)} = \arg\max_{l \in L} \left[ (c^{(k)} \circ \phi)(x_j) \right](l). \tag{6.1}$$

If all layers used the same features $\Phi_i$, the predictions $\mathbf{y}^{(k)} := (y_j^{(k)})_{j=1,\ldots,w \cdot h}$ would be the same for all $k$.[1] However, the features do change in our dynamic autocontext model and this is the key to our approach: Each new layer gains access not only to a new set of features, but to a whole new class of features. We distinguish three ordered

---

[1] Strictly speaking, this is only true for deterministically trained classifiers.

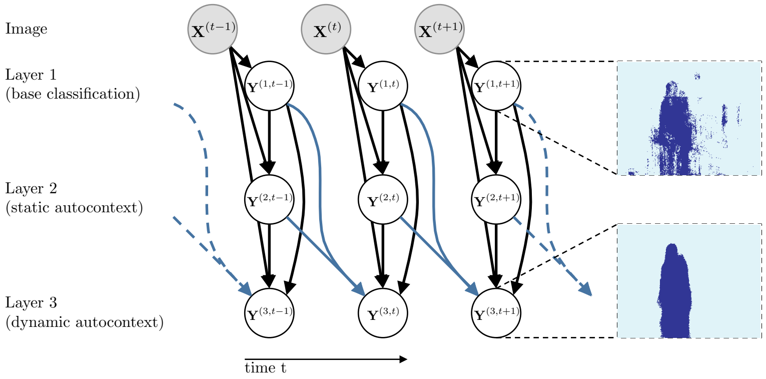


**Figure 6.2: Dynamic Autocontext.** Our approach consists of 3 conceptually different layers: Base classification (layer 1), static autocontext (layer 2), and dynamic autocontext (layer 3). Black lines indicate static autocontext [156], our contribution is displayed in blue.

and mutually exclusive feature classes, hereafter referred to as image features, static autocontext features, and dynamic autocontext features. Writing $\mathbf{y}^{(k,t)}$ for the output of layer $k$ at time $t$, we define:

- Image features are class 1 features and conditioned on the *current* input $\mathbf{x}^{(t)}$.
- Static autocontext features are class 2 features and conditioned on the *current* outputs $\mathbf{y}^{(1,t)}$ and $\mathbf{y}^{(2,t)}$.
- Dynamic autocontext features are class 3 features and conditioned on the *previous* outputs $\mathbf{y}^{(1,t-1)}$ and $\mathbf{y}^{(2,t-1)}$.

The base classifier of layer $k$ can access features whose class is at most $k$, *i.e.*, layer 1 can only use image features, layer 2 can additionally use static autocontext features, and layer 3 can additionally use dynamic autocontext features. Fig. 6.2 gives an overview of our method. Note that, theoretically, the base classifier could also differ from layer to layer, but for the purpose of this work we consider it to be fixed.

### 6.3.1 Instantiation

Up to this point, we have presented our approach in fairly general terms. In this section, we describe the specific instance of this framework, *i.e.*, the base classifier and the features, that we use for our experimental evaluation. All elements of our system are designed so as to give an optimal speed-accuracy trade-off, eventually allowing

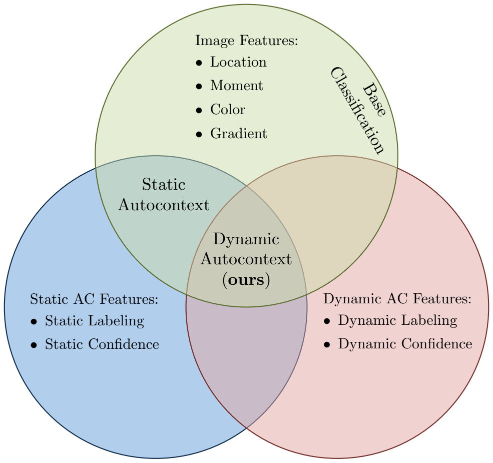


**Figure 6.3: Feature Hierarchy.** Visualization of the three feature classes, their members, and how they relate to the three layers of our dynamic autocontext model.

the application of our approach in time-constrained environments. We make frequent use of the mean response $\tau_R$ inside a rectangular image region $R$ with $|R|$ pixels,

$$\tau_R(\mathbf{x}) := |R|^{-1} \sum_{j \in R} x_j, \tag{6.2}$$

which we note can be computed efficiently using integral images [162].

**Base classifier.** We train a random forest with 12 trees using a training procedure similar to chapter 4, *i.e.*, we bootstrap resample the training data and recursively split leaf nodes by selecting the best among a set of hyperplane splits. Candidate splits are generated by sampling binary tests for random feature instances and scored based on the information gain of the resulting split. Note that this randomized training leads to the output $\mathbf{y}^{(k,t)}$ being a random variable itself. See [29] for more details.

The following paragraphs list only the features themselves; binary tests can be obtained by data-dependent thresholding of their values. For multi-dimensional features one dimension is selected uniformly at random.

**Image Features.** We use 4 simple but highly efficient image features based on the location, moments, color, and gradients of a single pixel $x_j$ or some associated rectangular image region(s) $R(x_j)$:

1. Location feature: We encode the relative location of the input pixel $x_j$ on the $x$-axis and $y$-axis, as well as its distance from the center of the image.
2. Moment feature: Given two image regions $(R_1(x_j), R_2(x_j))$, we record the difference in variance between those regions within an RGB channel and the difference in mean between those regions across RGB channels.
3. Color feature: We compute the inner product $\langle c, x_j \rangle$ between a color vector $c$ and the input pixel $x_j$.
4. Gradient feature: We use the oriented gradient mean response in an image region $R(x_j)$. [space-continuous histogram of gradients]

**Static Autocontext Features.** We employ two static autocontext features, a labeling feature and a confidence feature. In particular, the static labeling feature $f_L^{(k)}$ in layer $k > 1$ is defined as the mean prediction within a rectangular image region $R(x_j^{(t)})$ in layer $k' < k$,

$$f_L^{(k)}\left(x_j^{(t)} \,\middle|\, \mathbf{y}^{(k',t)}\right) := \tau_{R\left(x_j^{(t)}\right)}\left(\mathbf{y}^{(k',t)}\right) \in [0,1]. \tag{6.3}$$

The static confidence feature $f_C^{(k)}$ in layer $k > 1$ assesses the uncertainty associated with a prediction in layer $k' < k$. We use a variance-based measure of the forest posterior spread,

$$f_C^{(k)}\left(x_j^{(t)} \,\middle|\, \mathbf{y}^{(k',t)}\right) := 1 - 4 \cdot \sigma^2\left(y_j^{(k',t)}\right) \in [0,1], \tag{6.4}$$

which allows us to condition the actions of the classifier on its own (un)certainty in previous layers.

**Dynamic Autocontext Features.** We can easily turn the two static autocontext features into dynamic autocontext features by conditioning them on $\mathbf{y}^{(k',t-1)}$ instead of $\mathbf{y}^{(k',t)}$. By using the same fundamental type of features as both static and dynamic features, we move the focus of the evaluation from the predictive capabilities of the features to the effects of dynamic context. Fig. 6.3 summarizes all feature classes and illustrates their accessibility.

### 6.3.2 Hierarchical Training

The training of a dynamic autocontext hierarchy is simple and efficient. Without loss of generality, we assume that the supervised training set is given by a single video sequence $\mathcal{D}_{\mathbf{x}} = (x_j^{(t)})_{\substack{j=1,\ldots,w\cdot h\\ t=1,\ldots,T}}$ with ground truth segmentation $\mathcal{D}_{\mathbf{y}} = (y_j^{(\mathrm{gt},t)})_{\substack{j=1,\ldots,w\cdot h\\ t=1,\ldots,T}}$. Furthermore, let $\mathbf{Y}^{(k,\bullet)}$ be the $k$'th layer. Consider again Fig. 6.2 and note that $\mathbf{Y}^{(k,t)}$ is only dependent on variables in previous layers. By $d$-separation, we therefore have

$$\left(\mathbf{Y}^{(k,t)} \perp\!\!\!\perp \mathbf{Y}^{(k,\bullet)} - \mathbf{Y}^{(k,t)} \;\middle|\; \bigcup_{i=1}^{k-1} \mathbf{Y}^{(i,\bullet)}\right), \tag{6.5}$$

*i.e.*, the variables in layer $k$ are conditionally independent given the previous layers. Consequently, we can train the base classifier *layer by layer* on the supervised training set $(\mathcal{D}_{\mathbf{x}}, \mathcal{D}_{\mathbf{y}})$, in each layer adding the now accessible additional features. The asymptotic complexity for the training of a dynamic autocontext hierarchy is thus the same as that of the base classifier.

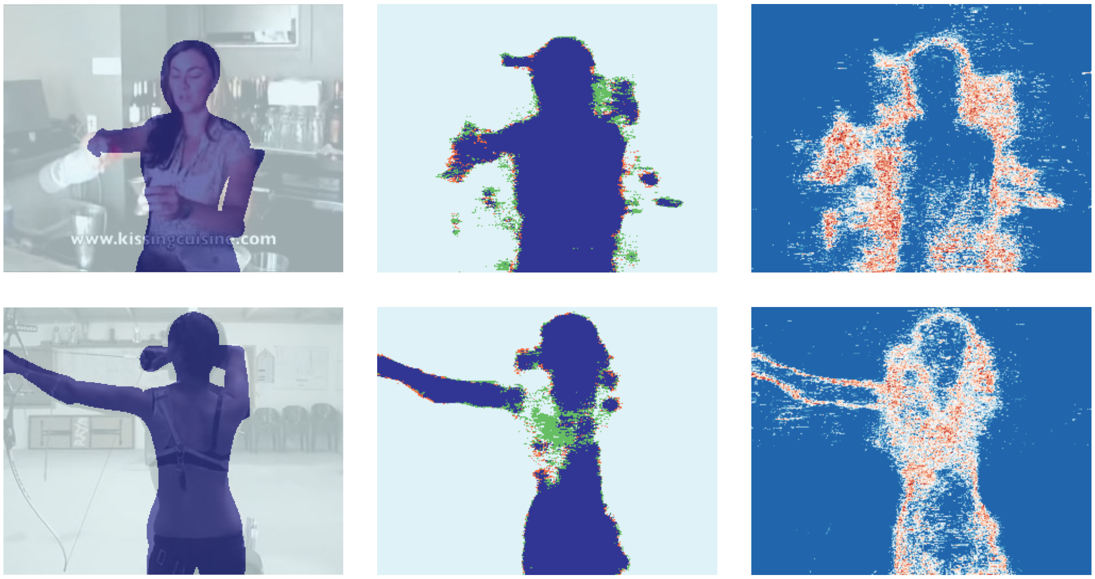
## 6.4 Experiments

People are among the most important objects in our lifes. Their analysis comprises all scales and touches virtually every scientific discipline. In computer vision, we are interested in where they are, what they do and who or what they interact with.

### 6.4.1 Datasets

We demonstrate the usefulness of our dynamic autocontext model for semantic labeling tasks on three challenging video datasets containing people, namely the Buffy dataset [44], the FBMS dataset [111], and the J-HMDB dataset [75]. The datasets vary greatly with regard to the number of training sequences, the number of instances per training sequence, and the type of annotation masks.

**J-HMDB.** The joint-annotated human motion database (J-HMDB) [75] is a large action recognition dataset containing extended annotations for the older HMDB51 dataset [88], which it is based on. The full dataset comprises a total of 928 short video clips taken from movies and YouTube. Each video clip is tagged with one out of 21 action classes and has an average length of approximately 34 frames, resulting in a total number of $\approx 3.2 \cdot 10^4$ frames. The authors provide dense ground truth puppet annotations for all frames.

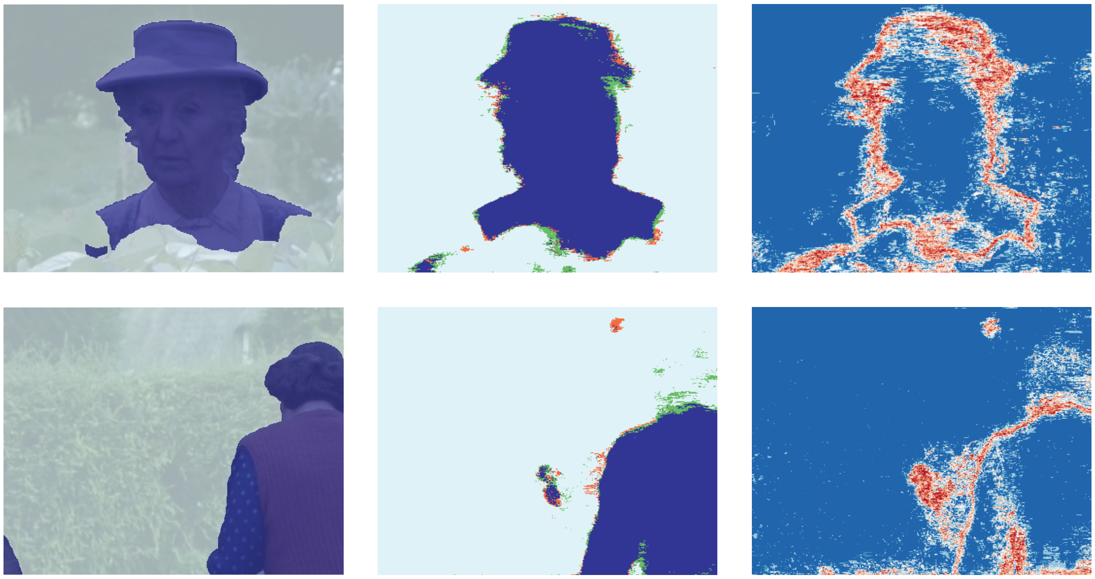


**(a)** Ground Truth **(b)** Prediction **(c)** Confidence

**Figure 6.4: Comparison of Static and Dynamic Autocontext (J-HMDB).** **(a)** Hand-labeled ground truth puppet annotations. **(b)** Illustration of the differences in the predictions of static and dynamic autocontext. Pixels classified correctly when using dynamic autocontext but incorrectly when using static autocontext are marked in green, the vice versa case is depicted in red. **(c)** Visualization of the second layer's confidence feature $f_C^{(2)}$ [blue $\widehat{=}$ high confidence; red $\widehat{=}$ low confidence]. Note that dynamic autocontext preferably acts in regions with low-confidence predictions in the second layer.

We select a representative subset of video frames in the following way: First, we randomly pick 7 categories and split the videos in each category evenly into training and testing. Next, each video sequence is subsampled by selecting 3 non-overlapping chunks of 4 consecutive frames, amounting to a total number of 1240 frames for training and another 1208 frames for testing.

**FBMS.** The Freiburg-Berkeley motion segmentation dataset (FBMS) consists of 720 annotated frames in 59 video sequences. We use the official split of a class containing video clips from Miss Marple ("marple"), consisting of 69 annotated frames for training and another 69 for testing. Some ground truth annotations in this class contain foreground objects other than people, which we manually reassign to background.



**(a)** Ground Truth **(b)** Prediction **(c)** Confidence

**Figure 6.5: Comparison of Static and Dynamic Autocontext (FBMS).** We show the same type of content as in Fig. 6.4. Dynamic autocontext does not alter correct high-confidence predictions made in previous layers but can improve on fine details or pick up new cues.

**Buffy.** The Buffy dataset contains a total of 748 fully annotated frames from the TV series “Buffy – The Vampire Slayer”. The frames are taken from 5 different episodes and include pixel-accurate segmentation masks for all people in the scene [90]. Most frames belong to local chunks of 4 non-consecutive but related frames. Since the original dataset does not include information about a frame’s predecessor, we add this information based on a temporal threshold $q$, *i.e.*,

$$\mathrm{pred}_q(\mathbf{x}^{(t)}) = \underset{\left\{\mathbf{x}^{(t')} \,\middle|\, 0<(t-t')\leq q\right\}}{\arg\min} \; t - t'. \tag{6.6}$$

For our experiments, we choose $q = 12$, which assigns a predecessor to roughly 50% of the frames. Note that this allows for large gaps of almost half a second between neighbouring frames, posing a serious challenge to any approach exploiting temporal information. We train our models on episodes $2 - 4$ (452 frames) and test on the remaining two episodes $5 - 6$ (160 frames).

### 6.4.2 Quantitative Analysis

We evaluate the quantitative performance on those datasets by calculating the average pixelwise accuracy, $\mathrm{F}_1$ score, and intersection over union (IoU) score, which are defined as follows:

$$\mathrm{Acc}_{\mathrm{pixel}} = \frac{\mathrm{TP} + \mathrm{TN}}{\#\mathrm{Pixel}}, \quad \mathrm{F}_1 = 2 \cdot \frac{\mathrm{precision} \cdot \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}}, \quad \mathrm{IoU} = \frac{|\mathrm{FG} \cap \mathrm{FG}_{\mathrm{gt}}|}{|\mathrm{FG} \cup \mathrm{FG}_{\mathrm{gt}}|}, \tag{6.7}$$

where TP are the true positives, TN are the true negatives, FG are the predicted foreground pixels, and $\mathrm{FG}_{\mathrm{gt}}$ are the ground truth foreground pixels.

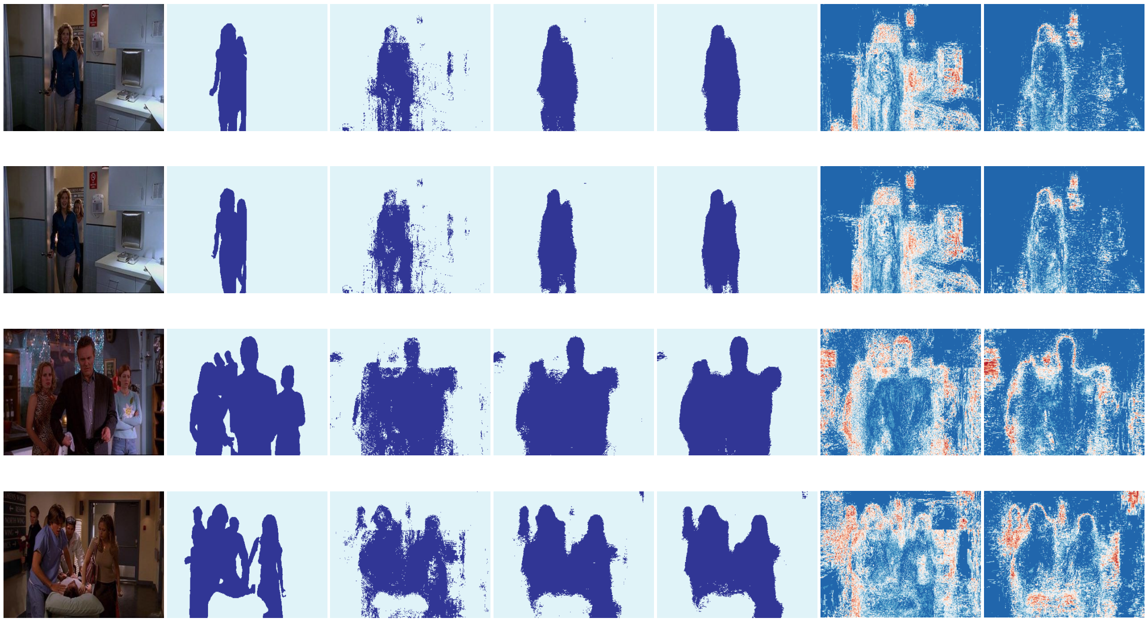
We compare our results to three popular baselines, namely random forests [143], approximate MAP inference in a fully-connected CRF with random forest unaries [86], and Tu's static autocontext with a random forest base classifier [156]. Table 6.1 shows our results. All evaluated methods exhibit the tendency to perform better when using deeper trees, independent of the dataset. At the same time, we observe small increases in runtime.

The pixelwise accuracies are on similar levels for all evaluated methods. While dynamic autocontext has an advantage on the Buffy dataset (79.99) and the J-HMDB dataset (90.00), static autocontext performs best on the FBMS dataset (71.61). However, we noticed only little correlation between pixelwise accuracy and visual quality, so that the actual predictions differ substantially.

The $\mathrm{F}_1$ score is the harmonic mean of precision and recall, that is, it aggregates both and penalizes differences between them. This may be a partial explanation for the weak performance of the dense CRF model, which suffers from oversmoothing effects resulting in a high precision and low recall. Our proposed model achieves the highest $\mathrm{F}_1$ score on all three datasets. The margin to the runner-up, the static autocontext model, is small on the J-HMDB dataset (+0.03%) but bigger on the other two (Buffy: +1.42%, FBMS: +2.09%).

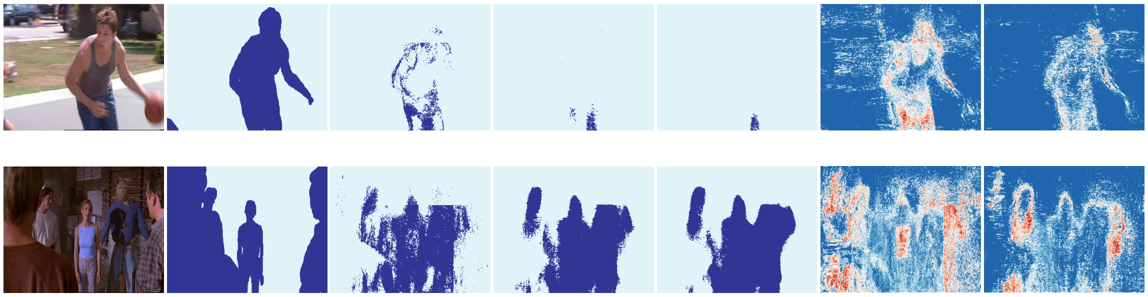
The intersection over union score is not only the strictest among the three evaluation criteria but also the most relevant one in terms of correlation with visual quality. In comparison to the other baselines, our dynamic autocontext model performs well again, outperforming its strongest competitor, the static autocontext model, by 1.51% (Buffy), 1.64% (FBMS), and 0.48% (J-HMDB).

We note that specialized models integrating multiple modalities and application-specific knowledge can achieve higher numbers on those datasets [161]. In case of human segmentation, they usually include person detections, pose estimation, shape priors, *etc.*, as discussed in section 6.1.2. This is outside the scope of this work, which is to propose a general and efficient dynamic autocontext model and compare its performance to static alternatives.



**(a)** Input **(b)** GT **(c)** RF **(d)** SAC **(e)** DAC **(f)** $f_C^{(3)}\left(x_j^{(t)}\middle|\mathbf{y}^{(1,t)}\right)$ **(g)** $f_C^{(3)}\left(x_j^{(t)}\middle|\mathbf{y}^{(2,t)}\right)$

**Figure 6.6: Positive Examples on the Buffy Dataset.** From left to right we show: **(a)** The RGB input frames. **(b)** The associated ground truth segmentations. **(c-e)** The predicted outputs when using random forests (RF), static autocontext (SAC) and dynamic autocontext (DAC). **(f-g)** Confidence maps for the predictions (c) and (d). [blue $\widehat{=}$ high confidence; red $\widehat{=}$ low confidence].



**(a)** Input **(b)** GT **(c)** RF **(d)** SAC **(e)** DAC **(f)** $f_C^{(3)}\left(x_j^{(t)}\middle|\mathbf{y}^{(1,t)}\right)$ **(g)** $f_C^{(3)}\left(x_j^{(t)}\middle|\mathbf{y}^{(2,t)}\right)$

**Figure 6.7: Failure Cases on the Buffy Dataset.** We display the same type of content as above. **First row:** The foreground uncertainty is too high, so that the person in the scene is falsely being removed by our approach. **Second row:** Our approach interprets the straw man as a person.

**(a)** Buffy Test Set.

| Method | Tree Depth | Performance | | | Runtime |
|---|---|---|---|---|---|
| | | Acc$_{\text{pixel}}$ | $F_1$ | IoU | (ms) |
| Random Forest [109] | 16 | 77.71 | 59.17 | 42.43 | 101 |
| | 20 | 77.64 | 59.37 | 42.63 | 131 |
| | 24 | 78.20 | 59.59 | 42.91 | 152 |
| Random Forest + Dense CRF [86] | 16 | 78.28 | 59.26 | 42.68 | 183 |
| | 20 | 78.42 | 59.92 | 43.52 | 215 |
| | 24 | 79.10 | 60.29 | 43.89 | 232 |
| Static Autocontext [156] | 16 | 79.23 | 61.02 | 44.57 | 189 |
| | 20 | 78.98 | 61.92 | 45.67 | 246 |
| | 24 | 79.40 | 62.60 | 46.50 | 288 |
| Dynamic Autocontext **(ours)** | 16 | 79.93 | 63.12 | 47.00 | 227 |
| | 20 | **79.99** | **64.02** | **48.01** | 290 |
| | 24 | 79.75 | 63.70 | 47.62 | 339 |

**(b)** FBMS Test Set.

| Method | Tree Depth | Performance | | | Runtime |
|---|---|---|---|---|---|
| | | Acc$_{\text{pixel}}$ | $F_1$ | IoU | (ms) |
| Random Forest [109] | 16 | 70.29 | 40.39 | 24.17 | 157 |
| | 20 | 70.53 | 43.28 | 26.81 | 178 |
| | 24 | 70.43 | 42.16 | 25.73 | 192 |
| Random Forest + Dense CRF [86] | 16 | 70.93 | 37.29 | 21.22 | 292 |
| | 20 | 71.14 | 40.83 | 24.57 | 321 |
| | 24 | 70.96 | 39.42 | 23.14 | 330 |
| Static Autocontext [156] | 16 | 69.45 | 48.79 | 29.91 | 306 |
| | 20 | 70.20 | 52.57 | 34.27 | 342 |
| | 24 | **71.61** | 53.60 | 34.36 | 362 |
| Dynamic Autocontext **(ours)** | 16 | 69.10 | 54.58 | 34.95 | 337 |
| | 20 | 69.58 | 54.19 | 35.60 | 387 |
| | 24 | 70.07 | **55.69** | **36.00** | 408 |

**(c)** J-HMDB Test Set.

| Method | Tree Depth | Performance | | | Runtime |
|---|---|---|---|---|---|
| | | Acc$_{\text{pixel}}$ | $F_1$ | IoU | (ms) |
| Random Forest [109] | 16 | 88.00 | 44.28 | 21.77 | 106 |
| | 20 | 88.80 | 52.60 | 28.78 | 127 |
| | 24 | 89.39 | 57.33 | 33.10 | 145 |
| Random Forest + Dense CRF [86] | 16 | 87.90 | 39.78 | 17.18 | 188 |
| | 20 | 89.05 | 49.77 | 23.74 | 210 |
| | 24 | 89.73 | 55.23 | 27.70 | 239 |
| Static Autocontext [156] | 16 | 88.48 | 54.80 | 28.99 | 179 |
| | 20 | 89.12 | 62.82 | 38.96 | 235 |
| | 24 | 89.58 | 66.22 | 43.42 | 269 |
| Dynamic Autocontext **(ours)** | 16 | 89.00 | 56.07 | 30.02 | 278 |
| | 20 | 89.63 | 63.09 | 39.13 | 321 |
| | 24 | **90.00** | **66.25** | **43.90** | 376 |

**Table 6.1: Quantitative Evaluation of Dynamic Autocontext.** We compare dynamic autocontext on 3 datasets against 3 baseline methods using 3 performance measures.

### 6.4.3 Qualitative Analysis

We support the numerical benefits of our dynamic autocontext model with some visual impressions. Fig. 6.4 and Fig. 6.5 contrast the predictions from the static and the dynamic autocontext model on the J-HMDB and FBMS dataset, respectively. Both figures show ground truth annotations (Fig. 6.4a, Fig. 6.5a) together with a visualization that highlights the differences between the two forms of autocontext (Fig. 6.4b, Fig. 6.5b). Different colors represent classifications that flip from being incorrect to being correct (green pixels) or vice versa (red pixels) when moving from the static to the dynamic framework.

Fig. 6.4 depicts two challenging images from the J-HMDB dataset in which the static autocontext result is noisy (top) or misses part of the foreground object (bottom). Dynamic autocontext can improve on some of those undesired effects, as indicated by the green pixels. Note that the dynamic autocontext layer revisits preferably those image regions that the static autocontext classifier is still uncertain about (Fig. 6.4c, Fig. 6.5c).

If the results obtained from the static autocontext classifier are already satisfying, there is only little room for improvement. In those cases, dynamic autocontext can still work out some fine details in the uncertain border regions, while leaving the remaining parts of the segmentation untouched. This is illustrated in Fig. 6.5 for two images from the FBMS dataset.

In Fig. 6.6, we show comprehensive visualizations of the proposed autocontext features for 4 images from the Buffy dataset. They may serve to illustrate progressive effects on the overall accuracy and confidence of the base classifier. We focus on complex scenes with multiple people and cluttered background. As supported by the images, our approach can deal well with these challenges. Note that, unlike the typical smoothing effects that can be observed in CRF-based models, dynamic autocontext can discover and eliminate new regions originally misclassified by the base classifier. Also note the general tendency of the base classifier to become more certain about its own predictions in later layers.

Finally, we complement our positive findings with an impression of some representative failure cases, which we show in Fig. 6.7. Typical causes of failure include a low-density foreground segmentation during base classification or ambiguous foreground objects.

# Chapter 7

# Conclusions and Directions

In this thesis, we investigated the use of non-parametric models for structured data. We made use of topological constraints, such as bounds on the indegree of a local node or the Markov order of the global network, to develop directed graphical models with low treewidth that allow for reliable learning and efficient inference. In order to compensate for inaccurate conditional independence assumptions, we used flexible yet efficient non-parametric local models like kernel density estimates and decision forests. Both techniques benefit greatly from our training on large-scale datasets, including Human 3.6M, CMU Motion Capture, and Microsoft COCO. The resulting priors can be easily integrated with rich, CNN-based image likelihoods.

In total, we introduced four such models to address challenging tasks in computer vision. In particular, we showed how to use non-parametric graphical models in applications related to the human body (*e.g.*, human pose, motion, and segmentation) and natural scenes (*e.g.*, object categorization, quantification, and detection). We classified our models according to the type(s) of context that they can exploit and distinguished between static, dynamic, and semantic context information (Fig. 1.1). Our static models exploit spatial context (chapter 3) and, additionally, extrinsic semantic context between recognition tasks (chapter 5). Our dynamic models take advantage of temporal context (chapter 4) as well as intrinsic semantic context between label sequences (chapter 6). Comprehensive experimental evaluations showed that our models compare favorably with both traditional approaches and state-of-the-art research. On the other hand, it also became clear that especially the integration of semantic context poses many challenges. Further research on the efficient propagation of information between multiple interacting modalities is required to unlock the full potential of such systems.

The remainder of this chapter is organized as follows: We begin by reviewing the theoretical aspects and practical results of our projects in some more detail. In the second part of the chapter, we outline possible modifications, extensions, and generalizations of the models presented in this thesis, which we hope will inspire future research in their respective areas.

## 7.1 Conclusions

In this section, we want to summarize contributions, outcomes, and open challenges that are specific to the individual projects and thus go beyond the general principles just mentioned.

**Non-parametric Priors of Human Pose.** We introduced a non-parametric Bayesian network serving as a prior of human pose. In order to learn the network structure, we used an extension of Chow-Liu trees to the continuous domain. The required pairwise mutual information values between the variables were obtained by means of a non-parametric estimator of differential entropy based on nearest-neighbor distances. We showed that our model admits the efficient calculation of exact log-likelihoods. Samples from our generative model can be obtained at a rate of multiple frames per second, which enables an easy generation of new pose hypotheses. In case of a large training set, our simple, fast, and accurate method for the computation of approximate log-likelihoods still allows the application of our approach in real-time.

In our experiments, we demonstrated that the proposed model achieves a better expected log-likelihood on the Human 3.6M test set than three global baselines and a Gaussian linear network. The comparison of different graph structures showed that our non-parametric approach to structure learning outperforms both the widely used kinematic chain and a higher-order variant thereof by a significant margin. We further illustrated the capabilities of our model to generalize to new poses not present in the training data (compositionality). We see widespread applicability in domains such as tracking, pose estimation, and pose denoising.

**Non-parametric Markov Models for Human Motion.** We presented Dynamic Forest Models (DFMs) as a novel approach to human motion modeling, generalizing autoregressive trees and introducing new training and regularization schemes. The use of bagging turned out to be an essential means against overfitting and complements other techniques like ridge regression and isotropic covariance estimation. Instead of relying on a latent space, we used a non-parametric and non-linear Markov model that allows exact and efficient inference, while at the same time being able to represent complex conditional mixture distributions. Our presentation of DFMs in terms of a pseudo latent variable model highlighted similarities and differences to HMMs. An analysis of the underlying conditional independence assumptions showed that DFMs decouple in time, which is the main reason for their computational advantage over HMMs.

The effectiveness of our approach was demonstrated in three different application scenarios: Action recognition, motion completion, and prediction of 3D trajectories from 2D inputs. The comparison with popular baselines and state-of-the-art latent variable models like HMMs and GPDMs showed that DFMs perform well in those areas, while still being computationally efficient. Given the positive findings, we believe that human motion models based on DFMs could be useful in many areas beyond the ones explored in this thesis, *e.g.*, character animation from videos.

**Non-parametric Scene Prior Hierarchies.** We transferred the basic ideas developed for human pose models to the domain of scene understanding and combined them with a flexible hierarchy and powerful image likelihoods. Our hierarchy consisted of a coarse-to-fine sequence of task-specific layers following information enclosure. We showed how this matryoshka-like approach can be used to build a first-order chain of conditional Bayesian networks in which each layer is conditioned on the information of its predecessor. In the special case of a bilayered occurrence→location hierarchy, this resulted in a dynamic topology for the non-parametric location network. Similar to ancestral sampling in traditional Bayesian networks, we can easily synthesize full scenes by sampling the hierarchy layer by layer and conditioning each layer on the previous sample.

Our large-scale experiments on Microsoft COCO validated the benefits of our approach for modeling high-dimensional densities and showed improvements in object categorization, quantification, and detection tasks. Possible extensions comprise the inclusion of amorphous background classes (*e.g.*, trees, sky), temporal information, and additional layers, such as a coarser class presence network and/or a finer instance segmentation network. Furthermore, we believe that the principles introduced in this work can be combined with orthogonal ideas [61, 131] to build a single, wholistic approach to scene understanding.

**Non-parametric Semantic Video Segmentation.** We proposed a dynamic autocontext model that integrates both spatial and temporal dependencies between semantic labels into a causal framework. After an initial base classification, two additional layers exploit intrinsic semantic context by considering (functions of) ouput labels of previous layers, both in the current frame as well as in the previous frame, as additional features.

Our experiments validated dynamic autocontext on three different datatsets: Buffy, FBMS, and J-HMDB. A comparison with widely used baselines showed that dynamic autocontext is on par with or slightly better than its static competitors. We think that dynamic autocontext is a useful extension of its static counterpart and that its value and performance will increase over time as better regularization techniques and larger training sets for semantic video segmentation become available.

## 7.2 Directions

The line of research discussed in this thesis may serve as a starting point for new projects. We want to mention a few promising ideas and hope that they will be picked up by the research community.

**Unification with CNNs.** Recently, there has been a growing research interest in common principles and possible unifications of probabilistic graphical models and convolutional neural networks [178, 140, 24]. It seems promising to extend this line of work and transfer those insights to non-parametric graphical models. For instance, a recent work by Kontschieder *et al.* [82] uses backpropagation through a decision tree to combine a standard CNN with a non-parametric layer. Similar concepts could be applied to kernel density networks as used in chapters 3 and 5, which are easily differentiable. Ideas from deep variational methods [80] and deep generative models [125] could possibly be used to mimic the expensive inference process in non-parametric graphical models and build a true end-to-end system acting as a deep non-parametric network.

**Alternative Topological Objectives.** Our adaptation of Chow-Liu trees to non-parametric Bayesian networks computes an optimal topology with respect to KL-divergence to the true underlying distribution. However, this is only one out of many structure learning objectives that can be pursued. An interesting counterpart to such a KL-tree is an efficiency-tree, *i.e.*, a tree whose topology is computed so as to allow for fast approximate inference. Moreover, we can form convex combinations of both types of trees, which effectively leads to a continuous spectrum (with discrete jumps) of trees with varying degrees of influence from each objective function. We briefly describe the principles of such a framework.

Let $\tau(\mathbf{x})$ be the runtime to compute the approximate likelihood of a point $\mathbf{x} \in \mathbb{R}^d$ in the joint density $f_{\mathbf{X}}(\mathbf{x})$. The expected runtime

$$\mathbb{E}[\tau(\mathbf{X})] = \int_{\mathbb{R}^d} \tau(\mathbf{x}) f_{\mathbf{X}}(\mathbf{x}) \, \mathrm{d}\mathbf{x} \tag{7.1}$$

can in general not be calculated analytically, because $\tau$ is a deterministic but complex function with no closed-form expression. However, we can easily draw a sample $S := \{\mathbf{s}^{(j)}\}_j$ from $f_{\mathbf{X}}$ and use Monte-Carlo integration,

$$\widehat{\mathbb{E}}[\tau(\mathbf{X})] = \frac{1}{|S|} \sum_{j=1}^{|S|} \tau(\mathbf{s}^{(j)}). \tag{7.2}$$

In order to obtain a runtime estimate for a non-parametric arborescence $\mathcal{B} = (G, f_{\mathbf{x}})$ with topology $G = (\mathbf{X}, E)$ and root $X_r$, we can apply Eq. (7.2) to its local models,

$$\widehat{\mathbb{E}}[\tau(\mathbf{X})] = \widehat{\mathbb{E}}[\tau(X_r)] + \sum_{\substack{i=1 \\ i \neq r}}^{|\mathbf{X}|} \Big( \widehat{\mathbb{E}}[\tau(X_i, X_{\mathrm{pa}(i)})] + \widehat{\mathbb{E}}[\tau(X_{\mathrm{pa}(i)})] \Big). \tag{7.3}$$

A possible edge-symmetric objective function is thus

$$\mathcal{T}_E := \arg\min_{\mathrm{pa}} \sum_{\substack{i=1 \\ i \neq r}}^{|\mathbf{X}|} \left[ \widehat{\mathbb{E}}[\tau(X_i, X_{\mathrm{pa}(i)})] + \frac{1}{2} \Big( \widehat{\mathbb{E}}[\tau(X_i)] + \widehat{\mathbb{E}}[\tau(X_{\mathrm{pa}(i)})] \Big) \right], \tag{7.4}$$

in which case the optimal efficiency-tree $\mathcal{T}_E$ is given by the maximum spanning tree of a complete graph with nodes $\mathbf{X}$ and edge weights $w(X_j, X_k) := \widehat{\mathbb{E}}[\tau(X_j, X_k)] + (\widehat{\mathbb{E}}[\tau(X_j)] + \widehat{\mathbb{E}}[\tau(X_k)])/2$.

The KL-tree $\mathcal{T}_{\mathrm{KL}}$ and the efficiency-tree $\mathcal{T}_E$ pursue orthogonal objectives. In practice, applications would benefit from a smooth trade-off. An ad-hoc solution to this problem is to choose $\alpha \in [0,1]$ and consider a convex combination of both trees,

$$\mathcal{T}_\alpha = \alpha \cdot \mathcal{T}_{\mathrm{KL}} + (1 - \alpha) \cdot \mathcal{T}_E, \tag{7.5}$$

where we assume that the trees are given in terms of a linear, undirected, and weighted edge list $\mathcal{T}_\bullet = (e_i^\bullet)_{i=1,\ldots,\binom{|\mathbf{X}|}{2}}$. Here, $e_i$ is the $i$-th edge w.r.t. an arbitrary but consistent order across all trees and $e_i^\bullet$ is its weight in $\mathcal{T}_\bullet$. We refer to the maximum spanning tree $\overline{\mathcal{T}_\alpha}$ of such a tree $\mathcal{T}_\alpha$ as an *intermediate spanning tree (IST)*. Although we can easily optimize the spanning tree objective for a single tree $\mathcal{T}_\alpha$, this approach has serious drawbacks. In particular, we know neither the total number of ISTs nor the values of $\alpha$ for which they are valid. While it is easy to see that each IST is valid in exactly one interval, a simple rescaling of either $\mathcal{T}_{\mathrm{KL}}$ or $\mathcal{T}_E$ shifts the transition points nonlinearly either to the left or to the right. In order to make a good choice, *e.g.*, the spanning tree that is halfway between $\mathcal{T}_{\mathrm{KL}}$ and $\mathcal{T}_E$, it is necessary to enumerate *all* intermediate trees and determine their valid intervals. We are thus interested in the quotient set $[0,1]/{\sim_{\mathcal{T}}}$ with respect to the equivalence relation $\alpha \sim_{\mathcal{T}} \alpha' \iff \overline{\mathcal{T}_\alpha} = \overline{\mathcal{T}_{\alpha'}}$.[1] The smallest elements in each equivalence class form a sorted representative system $(\alpha^{(i)})_i$ and, excluding border cases, $\overline{\mathcal{T}_{\alpha^{(i)}}}$ is valid in the half open interval $[\alpha^{(i)}, \alpha^{(i+1)}[$.

---

[1] At a transition point $\tilde{\alpha}$ the IST is not unique and we set $\overline{\mathcal{T}_{\tilde{\alpha}}} := \overline{\mathcal{T}_{\tilde{\alpha}+\epsilon}}$, where $\epsilon$ is infinitely small.

The transition points $\alpha^{(i)}$ can be computed in an efficient way. Our approach relies on a property of Kruskal's algorithm, which, in a nutshell, computes the maximum spanning tree of a graph by first sorting the edges according to their weights and then, from largest to smallest, iteratively adds one edge at a time to the spanning tree if the addition does not lead to a cycle. Changing the edge weights does therefore not change the spanning tree, as long as the *order* of the edge weights is preserved.

For any fixed $\alpha \in [0, 1]$, let $\pi_\alpha$ be the unique permutation with $\pi_\alpha(i) = k$ if and only if $e_i^\alpha$ is the $k$-largest edge in $\mathcal{T}_\alpha$. With the explanation above, we have

$$\pi_\alpha = \pi_{\alpha'} \implies \overline{\mathcal{T}_\alpha} = \overline{\mathcal{T}_{\alpha'}}. \tag{7.6}$$

The equivalence relation $\alpha \sim_\pi \alpha' \iff \pi_\alpha = \pi_{\alpha'}$ is finer than $\sim_\mathcal{T}$, *i.e.*, $\alpha \sim_\pi \alpha'$ is sufficient but not necessary for $\overline{\mathcal{T}_\alpha} = \overline{\mathcal{T}_{\alpha'}}$. However, merging of adjacent equivalence classes that lead to the same IST can be done in a single linear sweep, as explained below. To compute the equivalence classes of $\sim_\pi$, first note that each edge weight in $\mathcal{T}_\alpha$ is a convex combination $e_i^\alpha = \alpha \cdot e_i^1 + (1 - \alpha) \cdot e_i^0$ of the corresponding weights in $\mathcal{T}_{\text{KL}} = \mathcal{T}_1$ and $\mathcal{T}_E = \mathcal{T}_0$. Given two edges $e_i$,$e_j$ with $e_i^0 - e_j^0 \neq e_i^1 - e_j^1$ (*i.e.*, they are not parallel), we can solve for the unique value $\chi_{ij}$ at which their weights cross,

$$\chi_{ij} := \frac{\Delta_{ij}^0}{\Delta_{ij}^0 - \Delta_{ij}^1},$$

with $\Delta_{ij}^\bullet := e_i^\bullet - e_j^\bullet$. The two edges will swap within the valid range $[0, 1]$ if and only if $\mathbb{I}\big[e_i^0 < e_j^0\big] \wedge \mathbb{I}\big[e_i^1 > e_j^1\big]$ or vice versa. By definition, those are exactly the points that separate the equivalence classes w.r.t. $\sim_\pi$.

A valid approach to compute all intermediate spanning trees is thus to sort all candidate transitions $\chi_{ij} \in [0, 1]$ in ascending order and check sequentially if the new order $\pi_{\chi_{ij}}$ leads to a new IST by running Kruskal's algorithm. Since most candidate transitions will not result in a new solution, this is still a lot of computational overhead. Indeed, we need to run Kruskal's algorithm only in those cases that actually lead to a new intermediate spanning tree. To see this, let $\chi_{ij}$ be the $k$-smallest edge swap and $\chi_{ij}^{\leftarrow}$ its predecessor (or 0, if a predecessor does not exist). A swap can only occur between two edges with neighbouring weights and we assume w.l.o.g. that the weight of $e_i$ is smaller than that of $e_j$ before the swap. There are now four scenarios, three of which do not result in a new intermediate spanning tree:

1. $e_i, e_j \in \overline{\mathcal{T}_{\chi_{ij}^{\leftarrow}}} \Rightarrow \overline{\mathcal{T}_{\chi_{ij}}} = \overline{\mathcal{T}_{\chi_{ij}^{\leftarrow}}}$.
2. $e_i, e_j \notin \overline{\mathcal{T}_{\chi_{ij}^{\leftarrow}}} \Rightarrow \overline{\mathcal{T}_{\chi_{ij}}} = \overline{\mathcal{T}_{\chi_{ij}^{\leftarrow}}}$.

3. $e_i \notin \overline{\mathcal{T}_{\chi^{\leftarrow}_{ij}}}, e_j \in \overline{\mathcal{T}_{\chi^{\leftarrow}_{ij}}} \Rightarrow \overline{\mathcal{T}_{\chi_{ij}}} = \overline{\mathcal{T}_{\chi^{\leftarrow}_{ij}}}$.

The last case is different. If we have $e_i \in \overline{\mathcal{T}_{\chi^{\leftarrow}_{ij}}}$ and $e_j \notin \overline{\mathcal{T}_{\chi^{\leftarrow}_{ij}}}$, $e_j$ was rejected in $\overline{\mathcal{T}_{\chi^{\leftarrow}_{ij}}}$, because it took part in $c \geqslant 1$ cycles. We run Kruskal's algorithm to compute the new IST $\overline{\mathcal{T}_{\chi_{ij}}}$, if all of those cycles include $e_i$.

**Kernelization.** The non-parametric Bayesian network priors in our static models (chapters 3 and 5) use conditional kernel density estimates with a Gaussian kernel to model the objects in question, such as poses or scenes. A different type of kernel, called Mercer kernel, is used as a similarity measure in machine learning to allow the implicit computation of inner products in feature spaces [104, 139]. A particular instance of such a kernel is a Fisher kernel [74], which has been used to combine generative and discriminative models for information retrieval purposes. We can derive this kernel for non-parametric Bayesian networks as well to obtain a model-driven similarity measure for poses and scenes.

Given two inputs $\mathbf{x}$ and $\mathbf{x}'$, the Fisher kernel $\kappa_{\mathcal{F}}(\mathbf{x}, \mathbf{x}') := S_{\mathbf{x}}^{\top} \mathcal{I}(\boldsymbol{\theta})^{-1} S_{\mathbf{x}'}$ of a generative model with parameter vector $\boldsymbol{\theta} = (\theta_k)_{k=1}^{M}$ is an inner product between the Fisher scores $S_{\mathbf{x}}$ and $S_{\mathbf{x}'}$, where

$$S_{\bullet} := \nabla_{\boldsymbol{\theta}} \log p(\bullet; \boldsymbol{\theta}) = \left( \frac{\partial \log p(\bullet; \boldsymbol{\theta})}{\partial \theta_1}, \ldots, \frac{\partial \log p(\bullet; \boldsymbol{\theta})}{\partial \theta_M} \right)^{\top} \tag{7.7}$$

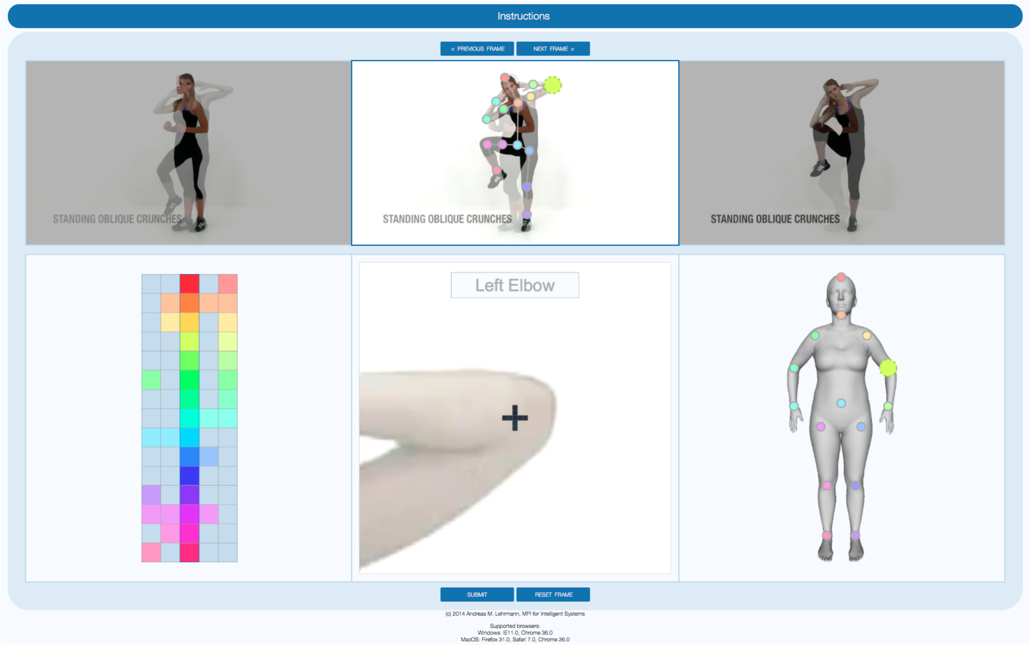
and the inverse of the Fisher information matrix $\mathcal{I}(\boldsymbol{\theta})$ performs a form of directional scaling. Since our model is non-parametric, its score depends directly on the data, *i.e.*, $\boldsymbol{\theta} = \mathcal{D}_{\mathbf{x}}$ and $\theta_k = \mathbf{x}^{(k)}$. As justified in [74], $\mathcal{I}(\boldsymbol{\theta})$ can be taken to be the identity matrix, so that the problem reduces to deriving $S_{\bullet}$. Writing $p(x_{i,j})$ as shorthand for $p(x_i, x_j)$, we have

$$\frac{\partial \log p(\mathbf{x})}{\partial \mathbf{x}^{(k)}} = \sum_{i=1}^{n} \left( \frac{\partial \log p(x_{i,\mathrm{pa}(i)})}{\partial \mathbf{x}^{(k)}} - \frac{\partial \log p(x_{\mathrm{pa}(i)})}{\partial \mathbf{x}^{(k)}} \right). \tag{7.8}$$

The local derivatives are given by

$$\frac{\partial \log p(x_{i,\mathrm{pa}(i)})}{\partial \mathbf{x}^{(k)}} = \frac{\partial}{\partial \mathbf{x}^{(k)}} \log \frac{1}{N} \sum_{j=1}^{N} \mathcal{N}\Big( x_{i,\mathrm{pa}(i)} \;\Big|\; x^{(j)}_{i,\mathrm{pa}(i)}, \big(\mathbf{B}^{(i)}\big)^2 \Big), \tag{7.9}$$

that is, they are partial derivatives of the log-likelihoods of a Gaussian mixture model with respect to its component means. The individual entries $\frac{\partial \log p(x_{i,\mathrm{pa}(i)})}{\partial x^{(k)}_{l}}$ are zero


Instructions
< PREVIOUS FRAME
NEXT FRAME >
STANDING OBLIQUE CRUNCHES
STANDING OBLIQUE CRUNCHES
STANDING OBLIQUE CRUNCHES
Left Elbow
SUBMIT
RESET FRAME

(c) 2014 Andreas M. Lehrmann, MPI for Intelligent Systems

Supported browsers:
Windows: IE11.0, Chrome 36.0
MacOS: Firefox 31.0, Safari 7.0, Chrome 36.0

**Figure 7.1: Online Annotation Tool.** Javascript front-end of our human video pose annotation tool for Amazon Mechanical Turk. Features include a magnifying glass, support for occluded/truncated joints, and previews of surrounding frames. Annotations are propagated from frame to frame to encourage consistency over time.

unless $l = i$ or $l = \mathrm{pa}(i)$, in which case we obtain

$$\frac{\partial \log p(x_{i,\mathrm{pa}(i)})}{\partial x^{(k)}_{i,\mathrm{pa}(i)}} = \frac{\mathcal{N}(x_{i,\mathrm{pa}(i)} \mid x^{(k)}_{i,\mathrm{pa}(i)}, (\mathbf{B}^{(i)})^2)}{\sum_{k'=1}^{N} \mathcal{N}(x_{i,\mathrm{pa}(i)} \mid x^{(k')}_{i,\mathrm{pa}(i)}, (\mathbf{B}^{(i)})^2)} \cdot \left(\mathbf{B}^{(i)}\right)^{-2} \left(x_{i,\mathrm{pa}(i)} - x^{(k)}_{i,\mathrm{pa}(i)}\right). \tag{7.10}$$

We believe that such a data-driven similarity measure could be more precise than the Euclidean distance between two parameterizations and useful in applications.

**Dynamic Autocontext for Human Video Pose Estimation.** Using dynamic autocontext for human video segmentation (chapter 6) turned out to be a challenging task. Possible reasons for the marginal improvement over static autocontext include complex interactions between thousands of class variables, high annotation uncertainty, and small training sets. Taking those challenges into account, human video pose estimation is a related field that might benefit more from dynamic autocontext.

As a starting point, we designed a Django-based online annotation tool with Javascript front-end and PostgreSQL back-end that encourages spatially accurate and temporally consistent annotations of human poses in video sequences (Fig. 7.1). Using this tool on Amazon Mechanical Turk, one can supplement existing image pose datasets (*e.g.*, the popular MPI-Inf dataset [3]) with annotations of surrounding frames, effectively turning them into video pose datasets that can be used to learn temporal dependencies between semantic pose labels.

**Temporal Dynamic Topologies.** Our dynamic forest model (chapter 4) uses information available in the previous $K$ frames to select a conditional motion model and, using this model, to make a prediction for the current frame. It seems promising to combine this temporal context model with the spatial model introduced in chapter 3 and the dynamic topology developed in chapter 5 to obtain a dynamic pose topology that is determined by temporal context.

As a practical example, consider the following situation: Over the course of an activity (*e.g.*, swimming), there might be certain phases in which it is most informative to use a tree-structure connecting the arms and other phases in which it is more appropriate to connect the legs. Such a time-dependent tree-structure could be predicted from previous frames.

# Appendices

# Appendix A

# Acronyms

**(A)MIAE** (Asymptotic) Mean Integrated Absolute Error.

**CDF** Cumulative Distribution Function.
**(C)KDE** (Conditional) Kernel Density Estimation.
**CNN** Convolutional Neural Network.
**CRF** Conditional Random Field.

**DAC** Dynamic Autocontext.
**DAG** Directed Acyclic Graph.
**DFM** Dynamic Forest Model.
**DTW** Dynamic Time Warping.

**ELL** Expected Log-likelihood.
**EM** Expectation Maximization.
**ETL** Expected Test Loss.

**FFT** Fast Fourier Transform.
**FPS** Frames per Second.

**GPDM** Gaussian Process Dynamical Model.
**GPLVM** Gaussian Process Latent Variable Model.

**HMC** Hybrid Monte Carlo.
**HMM** Hidden Markov Model.
**HOG** Histogram of Oriented Gradients.

**IOU** Intersection over Union.
**IST** Intermediate Spanning Tree.

**k-NN** k-Nearest Neighbor.

**LELVM** Laplacian Eigenmap Latent Variable Model.
**LOOCV** Leave-One-Out Cross-Validation.

**MAP** Maximum A-posteriori.
**MCMC** Markov Chain Monte Carlo.
**MI** Mutual Information.
**ML** Maximum Likelihood.
**MRF** Markov Random Field.
**MSE** Mean Squared Error.
**MST** Maximum Spanning Tree.

**PAC** Probabilistically Approximately Correct.
**(P)BP** (Particle) Belief Propagation.
**PCA** Principal Component Analysis.
**PDF** Probability Density Function.
**PMF** Probability Mass Function.

**RBF** Radial Basis Function.
**RCV** Root Coefficient of Variation.
**(R)RMSE** (Relative) Root Mean Squared Error.

**SCG** Scaled Conjugate Gradient.
**SIFT** Scale Invariant Feature Transform.
**(S)LDS** (Switching) Linear Dynamical System.
**SMC** Sequential Monte Carlo.
**SVM** Support Vector Machine.

# Appendix B

# Curriculum Vitae

*(removed in this public version)*

*(removed in this public version)*

*(removed in this public version)*

# Bibliography

[1] M. Aigner and G. M. Ziegler. *Das BUCH der Beweise*. Springer Spektrum, 2014.

[2] K. Alahari, G. Seguin, J. Sivic, and I. Laptev. Pose Estimation and Segmentation of People in 3D Movies. In *ICCV*, 2013.

[3] M. Andriluka, L. Pishchulin, P. Gehler, and B. Schiele. 2D Human Pose Estimation: New Benchmark and State of the Art Analysis. *CVPR*, 2014.

[4] D. Anguelov, P. Srinivasan, D. Koller, S. Thrun, J. Rodgers, and J. Davis. SCAPE: Shape Completion and Animation of People. *SIGGRAPH*, 2005.

[5] S. Arnborg, D. G. Corneil, and A. Proskurowski. Complexity of Finding Embeddings in a k-Tree. *SIAM Journal on Algebraic and Discrete Methods*, 1987.

[6] D. Barber. Dynamic Bayesian Networks with Deterministic Latent Tables. *NIPS*, 2003.

[7] D. Barber. *Bayesian Reasoning and Machine Learning*. Cambridge University Press, 2012.

[8] J. Besag. On the Statistical Analysis of Dirty Pictures. *Journal of the Royal Statistical Society*, 1986.

[9] H. L. Bodlaender. A Partial k-arboretum of Graphs with Bounded Treewidth. *Theoretical Computer Science*, 1998.

[10] H. L. Bodlaender. Discovering Treewidth. Technical report, Institute of Information and Computing Sciences, Utrecht University, 2005.

[11] Y. Boykov, O. Veksler, and R. Zabih. Fast Approximate Energy Minimization via Graph Cuts. *PAMI*, 2001.

[12] M. Brand, N. Oliver, and A. Pentland. Coupled Hidden Markov Models for Complex Action Recognition. *CVPR*, 1997.

[13] C. Bregler. Learning and Recognizing Human Dynamic in Video Sequences. *CVPR*, 1997.

[14] C. Bregler and J. Malik. Tracking People with Twists and Exponential Maps. *CVPR*, 1998.

[15] L. Breiman. Bagging Predictors. *Machine Learning*, 1996.

[16] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees.* Wadsworth, 1984.

[17] T. Brox and J. Malik. Object Segmentation by Long Term Analysis of Point Trajectories. In *ECCV*, 2010.

[18] T. Brox, B. Rosenhahn, U. G. Kersting, and D. Cremers. Nonparametric Density Estimation for Human Pose Tracking. *DAGM*, 2006.

[19] X. P. Burgos-Artizzu, P. Dollar, D. Lin, D. J. Anderson, and P. Perona. Social Behavior Recognition in Continuous Video. In *CVPR*, 2012.

[20] O. Cappé, S. J. Godsill, and E. Moulines. An Overview of Existing Methods and Recent Advances in Sequential Monte Carlo. *Proceedings of the IEEE*, 2007.

[21] O. Cappé, E. Moulines, and T. Rydéen. *Inference in Hidden Markov Models.* Springer, 2005.

[22] C.-C. Chang and C.-J. Lin. LIBSVM: A Library for Support Vector Machines. *ACM Trans. Intell. Syst. Technol.*, 2011.

[23] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman. Return of the Devil in the Details: Delving Deep into Convolutional Networks. In *BMVC*, 2014.

[24] L.-C. Chen, A. Schwing, A. Yuille, and R. Urtasun. Learning Deep Structured Models. *ICML*, 2015.

[25] M. J. Choi, J. J. Lim, A. Torralba, and A. S. Willsky. Exploiting Hierarchical Context on a Large Database of Object Categories. In *CVPR*, 2010.

[26] C. Chow and C. Liu. Approximating Discrete Probability Distributions with Dependence Trees. *IEEE Trans. Inf. Theory*, 1968.

[27] D. Comaniciu and P. Meer. Mean Shift: A Robust Approach Toward Feature Space Analysis. *PAMI*, 2002.

[28] T. Cormen, C. Leiserson, R. Rivest, and C. Stein. *Introduction to Algorithms.* MIT Press, 2009.

[29] A. Criminisi, J. Shotton, and E. Konukoglu. Decision Forests: A Unified Framework for Classification, Regression, Density Estimation, Manifold Learning and Semi-Supervised Learning. *Foundations and Trends in Computer Graphics and Vision*, 2012.

[30] N. Dalal and B. Triggs. Histograms of Oriented Gradients for Human Detection. In *CVPR*, 2005.

[31] A. Darwiche. *Modeling and Reasoning with Bayesian Networks*. Cambridge University Press, 2014.

[32] S. Dasgupta. Learning Polytrees. *UAI*, 1999.

[33] J. Deng, N. Diang, Y. Jia, A. Frome, K. Murphy, S. Bengio, Y. Li, H. Neven, and H. Adam. Large-Scale Object Classification using Label Relation Graphs. In *ECCV*, 2014.

[34] C. Desai, D. Ramanan, and C. Fowlkes. Discriminative Models for Multi-Class Object Layout. In *ICCV*, 2009.

[35] P. Dollár and C. L. Zitnick. Structured Forests for Fast Edge Detection. *ICCV*, 2013.

[36] C. Ek, P. Torr, and N. Lawrence. Gaussian Process Latent Variable Models for Human Pose Estimation. *MLMI*, 2007.

[37] A. Elgammal, R. Duraiswami, and L. Davis. Efficient Kernel Density Estimation Using the Fast Gauss Transform with Applications to Color Modeling and Tracking. *PAMI*, 2003.

[38] G. Elidan and S. Gould. Learning Bounded Treewidth Bayesian Networks. *JMLR*, 2008.

[39] J. Fan and Q. Yao. *Nonlinear Time Series: Nonparametric and Parametric Methods*. Springer, 2005.

[40] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. LIBLINEAR: A Library for Large Linear Classification. *JMLR*, 2008.

[41] C. Fellbaum, editor. *WordNet: An Electronic Lexical Database*. MIT Press, 1998.

[42] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan. Object Detection with Discriminatively Trained Part Based Models. *PAMI*, 2011.

[43] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient Graph-Based Image Segmentation. *IJCV*, 2004.

[44] V. Ferrari, M. Marin, and A. Zisserman. Progressive Search Space Reduction for Human Pose Estimation. In *CVPR*, 2008.

[45] S. Fothergill, H. M. Mentis, P. Kohli, and S. Nowozin. Instructing People for Training Gestural Interactive Systems. *CHI*, 2012.

[46] E. Fox, E. Sudderth, M. Jordan, and A. Willsky. Nonparametric Bayesian Learning of Switching Linear Dynamical Systems. *NIPS*, 2008.

[47] A. M. Fraser. *Hidden Markov Models and Dynamical Systems.* Society for Industrial and Applied Mathematics, 2008.

[48] Y. Freund and R. E. Schapire. A Decision-theoretic Generalization of On-Line Learning and an Application to Boosting. *Journal of Computer and System Sciences*, 1997.

[49] N. Friedman and Z. Yakhini. On the Sample Complexity of Learning Bayesian Networks. In *UAI*, 1996.

[50] K. Fukunaga and L. Hostetler. The Estimation of the Gradient of a Density Function, with Applications in Pattern Recognition. *IEEE Transactions on Information Theory*, 1975.

[51] F. Galasso, N. S. Nagaraja, T. J. Cardenas, T. Brox, and B. Schiele. A Unified Video Segmentation Benchmark: Annotation, Metrics and Analysis. In *ICCV*, 2013.

[52] C. Galleguillos, A. Rabinovich, and S. Belongie. Object Categorization using Co-occurrence, Location and Appearance. In *CVPR*, 2008.

[53] D. Geiger, T. Verma, and J. Pearl. Identifying Independence in Bayesian Networks. *Networks*, 1990.

[54] R. Girshick. Fast R-CNN. *ICCV*, 2015.

[55] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In *CVPR*, 2014.

[56] N. Gordon, D. Salmond, and A. Smith. Novel Approach to Nonlinear/Non-Gaussian Bayesian State Estimation. *IEE Proc. F. Radar and Signal Proc.*, 1993.

[57] M. Goria, N. Leonenko, V. Mergel, and P. Novi-Inverardi. A New Class of Random Vector Entropy Estimators and its Applications in Testing Statistical Hypotheses. *Journal of Nonparametric Statistics*, 2005.

[58] A. Gray and A. Moore. Nonparametric Density Estimation: Toward Computational Tractability. *SIAM Data Mining*, 2003.

[59] M. Grundmann, V. Kwatra, M. Han, and I. Essa. Efficient Hierarchical Graph-Based Video Segmentation. In *CVPR*, 2010.

[60] S. Hauberg, S. Sommer, and K. S. Pedersen. Gaussian-like Spatial Priors for Articulated Tracking. *ECCV*, 2010.

[61] G. Heitz and D. Koller. Learning Spatial Context: Using Stuff to Find Things. In *ECCV*, 2008.

[62] G. E. Hinton. Training Products of Experts by Minimizing Contrastive Divergence. *Neural Computation*, 2002.

[63] A. Hoerl and R. Kennard. Ridge Regression: Biased Estimation for Nonorthogonal Problems. *Technometrics*, 1970.

[64] K.-U. Höffgen. Learning and Robust Learning of Product Distributions. *COLT*, 1993.

[65] D. Hoiem, A. Efros, and M. Hebert. Putting Objects in Perspective. In *CVPR*, 2006.

[66] J. E. Hopcroft and R. M. Karp. An $n^{5/2}$ Algorithm for Maximum Matchings in Bipartite Graphs. *SIAM Journal on Computing*, 1973.

[67] R. A. Howard and J. E. Matheson. *Readings on the Principles and Applications of Decision Analysis*. Menlo Park, California: Strategic Decisions Group, 1984.

[68] R. Huang, V. Pavlovic, and D. N. Metaxas. A New Spatio-Temporal MRF Framework for Video-Based Object Segmentation. In *Machine Learning for Vision-Based Motion Analysis*, 2008.

[69] K. Ickstadt, B. Bornkamp, M. Grzegorczyk, J. Wieczorek, M. Sheriff, H. Grecco, and E. Zamir. Nonparametric Bayesian Networks. *Bayesian Statistics*, 2010.

[70] A. Ihler and D. McAllester. Particle Belief Propagation. *AISTATS*, 2009.

[71] C. Ionescu, D. Papava, V. Olaru, and C. Sminchisescu. Human 3.6M: Large Scale Datasets and Predictive Methods for 3D Human Sensing in Natural Environments. Technical report, University of Bonn, 2012.

[72] E. Ising. Beitrag zur Theorie des Ferromagnetismus. *Zeitschrift für Physik*, 1925.

[73] K. Ito and K. Xiong. Gaussian Filters for Nonlinear Filtering Problems. *IEEE Trans. on Automatic Control*, 2000.

[74] T. Jaakkola and D. Haussler. Exploiting Generative Models in Discriminative Classifiers. *NIPS*, 1998.

[75] H. Jhuang, J. Gall, S. Zuffi, C. Schmid, and M. J. Black. Towards Understanding Action Recognition. In *ICCV*, 2013.

[76] M. C. Jones, J. S. Marron, and S. J. Sheather. A Brief Survey of Bandwidth Selection for Density Estimation. *Journal of the American Statistical Association*, 1996.

[77] S. J. Julier and J. K. Uhlmann. A New Extension of the Kalman Filter to Nonlinear Systems. *Proc. AeroSense: 11th Int. Symp. Aerospace/Defense Sensing, Simulation and Controls*, 1997.

[78] J. H. Kim and J. Pearl. A Computational Model for Causal and Diagnostic Reasoning in Inference Systems. *8th International Joint Conference on Artificial Intelligence*, 1983.

[79] R. Kindermann and J. L. Snell. *Markov Random Fields and Their Applications.* American Mathematical Society, 1980.

[80] D. P. Kingma and M. Welling. Autoencoding Variational Bayes. *arXiv:1312.6114 [stat.ML]*, 2014.

[81] D. Koller and N. Friedman. *Probabilistic Graphical Models.* MIT Press, 2009.

[82] P. Kontschieder, M. Fiterau, A. Criminisi, and S. R. Bulò. Deep neural decision forests. *ICCV*, 2015.

[83] P. Kontschieder, P. Kohli, J. Shotton, and A. Criminisi. GeoF: Geodesic Forests for Learning Coupled Predictors. In *CVPR*, 2013.

[84] J. H. Korhonen and P. Parviainen. Exact Learning of Bounded Tree-width Bayesian Networks. *AISTATS*, 2013.

[85] L. F. Kozachenko and N. N. Leonenko. Sample Estimate of the Entropy of a Random Vector. *Problems of Information Transmission*, 1987.

[86] P. Krähenbühl and V. Koltun. Efficient Inference in Fully Connected CRFs with Gaussian Edge Potentials. In *NIPS*, 2011.

[87] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In *NIPS*, 2012.

[88] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre. HMDB: A Large Video Database for Human Motion Recognition. *ICCV*, 2011.

[89] L. Ladický, P. Sturgess, K. Alahari, C. Russell, and P. H. Torr. What, Where and How Many? Combining Object Detectors and CRFs. In *ECCV*, 2010.

[90] L. Ladický, P. H. Torr, and A. Zisserman. Human Pose Estimation Using a Joint Pixel-wise and Part-wise Formulation. In *CVPR*, 2013.

[91] J.-F. Lalonde, D. Hoiem, A. A. Efros, C. Rother, J. Winn, and A. Criminisi. Photo Clip Art. *ACM Transactions on Graphics*, 2007.

[92] T. Lan, M. Raptis, L. Sigal, and G. Mori. From Subcategories to Visual Composites: A Multi-Level Framework for Object Detection. In *ICCV*, 2013.

[93] X. Lan and D. P. Huttenlocher. Beyond Trees: Common-Factor Models for 2D Human Pose Recovery. *ICCV*, 2005.

[94] S. L. Lauritzen and D. J. Spiegelhalter. Local Computations with Probabilities on Graphical Structures and their Application to Expert Systems. *Journal of the Royal Statistical Society*, 1988.

[95] N. Lawrence. Gaussian Process Latent Variable Models for Visualisation of High Dimensional Data. *NIPS*, 2003.

[96] M. W. Lee and I. Cohen. Human upper body pose estimation in static images. *ECCV*, 2004.

[97] W. Lenz. Beitrag zum Verständnis der magnetischen Erscheinungen in festen Körpern. *Zeitschrift für Physik*, 1920.

[98] C. Li, D. Parikh, and T. Chen. Automatic Discovery of Groups of Objects for Scene Understanding. In *CVPR*, 2012.

[99] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár. Microsoft COCO: Common Objects in Context. In *arXiv:1405.0312 [cs.CV]*, 2014.

[100] D. G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. In *IJCV*, 2005.

[101] Z. Lu, M. Á. Carreira-Perpiñán, and C. Sminchisescu. People Tracking with the Laplacian Eigenmaps Latent Variable Model. *NIPS*, 2007.

[102] F. Luthon, A. Caplier, and M. Liévin. Spatiotemporal MRF Approach to Video Segmentation: Application to Motion Detection and Lip Segmentation. *Signal Processing*, 1999.

[103] C. Meek, D. M. Chickering, and D. Heckerman. Autoregressive Tree Models for Time-Series Analysis. *SIAM*, 2002.

[104] J. Mercer. Functions of Positive and Negative Type and their Connection with the Theory of Integral Equations. *Philosophical Transactions of the Royal Society*, 1909.

[105] A. Montillo, J. Shotton, J. Winn, J. E. Iglesias, D. Metaxas, and A. Criminisi. Entangled Decision Forests and their Application for Semantic Segmentation of CT Images. In *IPMI*, 2011.

[106] D. Munoz, J. A. Bagnell, and M. Hebert. Stacked Hierarchical Labeling. In *ECCV*, 2010.

[107] A. Y. Ng, M. I. Jordan, and Y. Weiss. On Spectral Clustering: Analysis and an Algorithm. In *NIPS*, 2001.

[108] S. Nie, D. D. Mauá, C. P. de Campos, and Q. Ji. Advances in Learning Bayesian Networks of Bounded Treewidth. *NIPS*, 2014.

[109] S. Nowozin, C. Rother, S. Bagon, T. Sharp, B. Yao, and P. Kohli. Decision Tree Fields. In *ICCV*, 2011.

[110] S. Nowozin and J. Shotton. Action Points: A Representation for Low-Latency Online Human Action Recognition. Technical report, 2012. MSR-TR-2012-68.

[111] P. Ochs, J. Malik, and T. Brox. Segmentation of Moving Objects by Long Term Video Analysis. *PAMI*, 2014.

[112] J. Pacheco, S. Zuffi, M. J. Black, and E. B. Sudderth. Preserving Modes and Messages via Diverse Particle Selection. In *ICML*, 2013.

[113] A. Papazoglou and V. Ferrari. Fast Object Segmentation in Unconstrained Video. In *ICCV*, 2013.

[114] E. Parzen. On Estimation of a Probability Density Function and Mode. *Annals of Mathematical Statistics*, 1962.

[115] V. Pavlović, J. M. Rehg, and J. MacCormick. Learning Switching Linear Models of Human Motion. *NIPS*, 2000.

[116] J. Pearl. Reverend Bayes on Inference Engines: A Distributed Hierarchical Approach. *AAAI National Conference on Artificial Intelligence*, 1982.

[117] J. Pearl. Bayesian Networks: A Model of Self-Activated Memory for Evidential Reasoning. Technical report, Computer Science Department, University of California, 1985.

[118] J. Pearl. *Probabilistic Reasoning in Intelligent Systems.* Morgan Kaufmann, 1988.

[119] R. Potts. Some Generalized Order-Disorder Transformations. *Mathematical Proceedings of the Cambridge Philosophical Society*, 1952.

[120] C. Preston. *Gibbs States on Countable Sets.* Cambridge University Press, 1974.

[121] J. Quinlan. Induction of Decision Trees. *Machine Learning*, 1986.

[122] J. Quinlan. *C4.5: Programs for Machine Learning.* Morgan Kaufmann, 1992.

[123] A. Rabinovich, A. Vedaldi, C. Galleguillos, E. Wiewiora, and S. Belongie. Objects in Context. In *CVPR*, 2007.

[124] S. Rena, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *arXiv:1506.01497 [cs.CV]*, 2015.

[125] D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic Backpropagation and Approximate Inference in Deep Generative Models. *arXiv:1401.4082 [stat.ML]*, 2014.

[126] N. Robertson and P. D. Seymour. Graph Minors. III. Planar Tree-width. *Journal of Combinatorial Theory*, 1984.

[127] N. Robertson and P. D. Seymour. Graph Minors. II. Algorithmic Aspects of Tree-width. *Journal of Algorithms*, 1986.

[128] M. Rosenblatt. Remarks on Some Nonparametric Estimates of a Density Function. *Annals of Mathematical Statistics*, 1956.

[129] M. Rudemo. Empirical Choice of Histograms and Kernel Density Estimators. *Scandinavian Journal of Statistics*, 1982.

[130] A. R. Runnalls. A Kullback-Leibler Approach to Gaussian Mixture Reduction. *IEEE Transactions on Aerospace and Electronic Systems*, 2006.

[131] F. Sadeghi, S. K. Divvala, and A. Farhadi. VisKE: Visual Knowledge Extraction and Question Answering by Visual Verification of Relation Phrases. *CVPR*, 2015.

[132] M. A. Sadeghi and A. Farhadi. Recognition Using Visual Phrases. In *CVPR*, 2011.

[133] S. R. Sain, K. A. Baggerly, and D. W. Scott. Cross-Validation of Multivariate Densities. *Journal of the American Statistical Association*, 1994.

[134] D. Salmond. Mixture Reduction Algorithms for Target Tracking in Clutter. *Signal and Data Processing of Small Targets*, 1990.

[135] S. Salvador and P. Chan. Toward Accurate Dynamic Time Warping in Linear Time and Space. *Intell. Data Anal.*, 2007.

[136] J. Schaefer and K. Strimmer. A Shrinkage Approach to Large-Scale Covariance Matrix Estimation and Implications for Functional Genomics. *Stat. Appl. Genet. Molec. Biol.*, 2005.

[137] D. Schlesinger. A Real-Time MRF-Based Approach for Binary Segmentation. In *DAGM/OAGM*, 2012.

[138] U. Schmidt, C. Rother, S. Nowozin, J. Jancsary, and S. Roth. Discriminative Non-blind Deblurring. In *CVPR*, 2013.

[139] B. Schölkopf and A. J. Smola. *Learning with Kernels*. MIT Press, 2001.

[140] A. G. Schwing and R. Urtasun. Fully Connected Deep Structured Networks. *arXiv:1503.02351 [cs.CV]*, 2015.

[141] D. Scott. *Multivariate Density Estimation: Theory, Practice, and Visualization.* Wiley, 1992.

[142] D. Scott and M. Wand. Feasibility of Multivariate Density Estimation. In *Biometrika*, 1991.

[143] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. Real-Time Human Pose Recognition in Parts from Single Depth Images. *CVPR*, 2011.

[144] L. Sigal, A. Balan, and M. J. Black. HumanEva: Synchronized Video and Motion Capture Dataset and Baseline Algorithm for Evaluation of Articulated Human Motion. *IJCV*, 2010.

[145] L. Sigal, M. Isard, H. W. Haussecker, and M. J. Black. Loose-limbed People: Estimating 3D Human Pose and Motion Using Non-parametric Belief Propagation. *IJCV*, 2012.

[146] B. Silverman. Kernel Density Estimation using the Fast Fourier Transform. *Journal of the Royal Statistical Society*, 1982.

[147] B. W. Silverman. *Density Estimation for Statistics and Data Analysis.* Springer, 1986.

[148] E. Snelson and Z. Ghahramani. Sparse Gaussian Processes Using Pseudo-inputs. *NIPS*, 2006.

[149] F. Spitzer. Random Fields and Interacting Particle Systems. *M.A.A. Summer Seminar Notes*, 1971.

[150] C. J. Stone. An Asymptotically Optimal Window Selection Rule for Kernel Density Estimates. *The Annals of Statistics*, 1984.

[151] E. B. Sudderth, A. T. Ihler, W. T. Freeman, and A. S. Willsky. Non-parametric Belief Propagation. *CVPR*, 2003.

[152] E. B. Sudderth, A. Torralba, W. T. Freeman, and A. S. Willsky. Learning Hierarchical Models of Scenes, Objects, and Parts. In *ICCV*, 2005.

[153] G. W. Taylor, G. E. Hinton, and S. Roweis. Modeling Human Motion Using Binary Latent Variables. *NIPS*, 2007.

[154] G. W. Taylor, L. Sigal, D. J. Fleet, and G. E. Hinton. Dynamical Binary Latent Variable Models for 3D Human Pose Tracking. *CVPR*, 2010.

[155] A. Torralba, K. P. Murphy, and W. T. Freeman. Contextual Models for Object Detection Using Boosted Random Fields. In *NIPS*, 2005.

[156] Z. Tu. Auto-context and its Application to High-level Vision Tasks. In *CVPR*, 2008.

[157] Z. Tu and X. Bai. Auto-context and its Application to High-level Vision Tasks and 3D Brain Image Segmentation. *PAMI*, 2010.

[158] J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, and A. W. M. Smeulders. Selective Search for Object Recognition. In *IJCV*, 2013.

[159] R. Urtasun, D. J. Fleet, and P. Fua. 3D People Tracking with Gaussian Process Dynamical Models. *CVPR*, 2006.

[160] R. Urtasun, D. J. Fleet, and N. D. Lawrence. Modeling Human Locomotion with Topologically Constrained Latent Variable Models. *2nd Workshop on Human Motion*, 2007.

[161] V. Vineet, J. Warrell, L. Ladický, and P. H. S. Torr. Human Instance Segmentation from Video Using Detector-based Conditional Random Fields. In *BMVC*, 2011.

[162] P. Viola and M. Jones. Rapid Object Detection Using a Boosted Cascade of Simple Features. *CVPR*, 2001.

[163] M. P. Wand and M. C. Jones. Multivariate Plug-In Bandwidth Selection. *Computational Statistics*, 1994.

[164] J. M. Wang, D. J. Fleet, and A. Hertzmann. Gaussian Process Dynamical Models for Human Motion. *PAMI*, 2008.

[165] J. M. Wang, D. J. Fleet, and A. Hertzmann. Optimizing Walking Controllers for Uncertain Inputs and Environments. *ACM Trans. Graphics*, 2010.

[166] J. M. Wang, S. R. Hamner, S. L. Delp, and V. Koltun. Optimizing Locomotion Controllers Using Biologically-based Actuators and Objectives. *ACM Trans. Graphics*, 2012.

[167] Q. Wang, S. Kulkarni, and S. Verdú. Universal Estimation of Information Measures for Analog Sources. *Found. Trends Commun. Inf. Theory*, 2009.

[168] L. Wasserman. *All of Nonparametric Statistics*. Springer, 2006.

[169] D. H. Wolpert. Stacked Generalization. *Neural Networks*, 1992.

[170] F.-Y. Wu. The Potts Model. *Reviews of Modern Physics*, 1982.

[171] C. Xu, C. Xiong, and J. J. Corso. Streaming Hierarchical Video Segmentation. In *ECCV*, 2012.

[172] C. Yang. Improved Fast Gauss Transform and Efficient Kernel Density Estimation. *ICCV*, 2003.

[173] Y. Yang and D. Ramanan. Articulated Human Detection with Flexible Mixtures of Parts. In *CVPR*, 2011.

[174] B. Yao and F.-F. Li. Modeling Mutual Context of Object and Human Pose in Human-Object Interaction Activities. *CVPR*, 2010.

[175] J. Yao, S. Fidler, and R. Urtasun. Describing the Scene as a Whole: Joint Object Detection, Scene Classification and Semantic Segmentation. In *CVPR*, 2012.

[176] Z. Yin and R. T. Collins. Belief Propagation in a 3D Spatio-temporal MRF for Moving Object Detection. In *CVPR*, 2007.

[177] B. Zadrozny and C. Elkan. Transforming Classifier Scores into Accurate Multiclass Probability Estimates. In *Knowledge Discovery and Data Mining*, 2002.

[178] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. H. S. Torr. Conditional Random Fields as Recurrent Neural Networks. *arXiv:1502.03240 [cs.CV]*, 2015.

# Index

*d*-separation, 11
*k*-nearest neighbor, 52
3D from 2D, 57

action recognition, 52
arborescence, 25
autocontext
    dynamic, 90
    static, 89
autoregressive tree, 46

bagging, 47
bandwidth, 13
base classifier, 89
Bayesian network, 10
    conditional, 64
    location network, 67
    occurrence network, 65
bipartite matching, 70

Chow-Liu tree, 31
classification, 8
cluster
    approximate cluster, 40
    core cluster, 40
compositionality, 38
conditional independence, 10
context
    dynamic, 21
    semantic, 21
    static, 20
contrastive divergence, 45

decision forest, 16
    dynamic, 47
    entangled, 89
decision tree, 15
    classification tree, 16
    regression tree, 16
density, 8
    – estimation, 8
distribution
    conditional, 9
    discrete smoothing, 67
    marginal, 9
    mixture, 17
dynamic time warping, 52

empirical risk, 75
ensemble, 47
    – model, 17
estimation
    entropy, 31
Euler-Mascheroni constant, 32

feature
    CNN FC-7, 75
    HoG, 67
    SIFT, 67
feature space, 8

Gaussian process dynamical model, 29, 44
Gaussian process latent variable model, 29
graph structure, 10
    dynamic, 69
graphical model, 10

- directed, *see* Bayesian network
- non-parametric, 12
- undirected, *see* Markov network

hidden Markov model, 52
human pose, 18

imCRBM, 30
inference, 9
  max-product, 74
  sum-product, 73
information enclosure, 64
input space, 8

Kalman filter, 44
kernel, 13
  Fisher kernel, 107
kernel density estimation, 13
  conditional, 33, 72
  log-likelihood, 34
  sampling, 34
kinematic chain, 29
Kullback-Leibler, 31

Laplacian eigenmap latent variable model, 30
latent variable model, 44
learning, 8
  supervised, 8
local distribution, 11
local model, 11
log-likelihood, 9
  approximate, 39
log-loss, 75

Markov model, 45
  nonlinear, 45
Markov network, 11
maximum a-posteriori (MAP), 9
maximum spanning tree, 31
Monte Carlo
  hybrid, 45
  sequential, 44
motion completion, 54
mutual information, 31

natural scene, 18

pose prior, 28
prior hierarchy, 64
probabilistically approximately correct, 22

regression, 8
ridge regression, 48

sample, 9
SCAPE, 58
scene layout synthesis, 80
scene understanding
  object location, 64
  object occurrence, 64
selective search, 73
split function, 15
  axis-aligned, 16
stacking, 89
state-space equations, 44
support vector machine, 52
switching linear dynamical system, 44

training point, 8
training set, 8
  CMU motion capture, 54
  COCO, 74
  Human 3.6M, 28
  MSRC-12, 51
  supervised, 8
treewidth, 23