University of California
Los Angeles

Stochastic Image Grammars for Human Pose Estimation

A dissertation submitted in partial satisfaction
of the requirements for the degree
Doctor of Philosophy in Computer Science

by

Brandon Rothrock

2013

Abstract of the Dissertation

Stochastic Image Grammars for Human Pose Estimation

by

Brandon Rothrock

Doctor of Philosophy in Computer Science

University of California, Los Angeles, 2013

Professor Song-Chun Zhu, Chair

Robust human pose estimation is of particular interest to the computer vision community, and can be applied to a broad range of applications such as automated surveillance, human-computer interaction, and human activity recognition. In this dissertation, we present a framework for human pose estimation based on stochastic image grammars. Humans in particular are difficult to model, as their articulated geometry, camera viewpoint, and perspective, can produce a very large number of distinct shapes in images. Furthermore, humans often exhibit highly variant and amorphous part appearances, have self-occlusion, and commonly appear in cluttered environments. Our approach capitalizes on the reconfigurable and modular nature of grammatical models to cope with this variability in both geometry and appearance. We present a human body model as a stochastic context-sensitive AND-OR graph grammar, which represents the body as a hierarchical composition of primitive parts while maintaining the articulated kinematics between parts. Each body instance can be composed from a different set of parts and relations in order to explain the unique shape or appearance of that instance. We present grammar models based on coarse-to-fine phrase-

structured grammars as well as dependency grammars, and describe efficient algorithms for learning and inference from both generative and discriminative perspectives. Furthermore, we propose extensions to our model to provide ambiguity reasoning in crowded scenes through the use of composite cluster sampling, and reasoning for self-occlusion and external occlusion of parts. We also present a technique to incorporate image segmentation into the part appearance models to improve localization performance on difficult to detect parts. Finally, we demonstrate the effectiveness of our approach by showing state-of-art performance on several recent public benchmark datasets.

The dissertation of Brandon Rothrock is approved.

Yingnian Wu

Zhuowen Tu

Demetri Terzopoulos

Song-Chun Zhu, Committee Chair

University of California, Los Angeles

2013

To my parents . . .

whose example of discipline is truly unmatched.

Table of Contents

List of Figures

List of Tables

Acknowledgments

I would first like to thank my advisor Song-Chun Zhu for his guidance and mentorship during my study at UCLA. His passion for research, expertise, and perspective has been an inspiration, and has helped me improve as a researcher, professional, and as a person. I would also like to thank my entire committee, Ying-Nian Wu, Demetri Terzopoulous, Zhuowen Tu, as well as Alan Yuille for their insightful advice and discussions along the way. My appreciation also goes to Adrian Barbu for his inspired discussions during his summer collaborations at UCLA.

Many thanks also goes out to my many colleagues in the VCLA lab for their friendship, collaboration, and support throughout my time here. In particular, I extend my thanks to Benjamin Yao, Tianfu Wu, Jake Porway, Zhangzhang Si, Chuck Fleming, Haifeng Gong, Kristy Wang, Mingtian Zhao, Wenze Hu, Zhi Han, Mingtao Pei, Jungseock Joo, Seyoung Park, Shuo Wang, Arash Gholami Rad, Amy Morrow, Maria Pavlovskaia, Bo Zheng, Jifeng Dai, Lu Yang, Joyce Meng, Bruce Nie, Kewei Tu, Ping Wei, Dan Xie, Joey Yu, and Yibiao Zhao.

Lastly, I cannot thank my family enough for their unwavering support.

Portions of this work were supported by DARPA MSEE project FA 8650-11-1-7149, MURI ONR N00014-10-1-0933, NSF CNS 1028381, and NSF IIS 1018751.

Vita

2000	B.S. (Aeronautical and Astronautical Engineering), University of Washington, Seattle, Washington.
2000-2003	Software Design Engineer, Hewlett-Packard, Roseville, California.
2003-2005	Software Design Lead, Software-Labs, Roseville, California.
2005-2007	Staff Researcher, Carnegie Melon University, Pittsburgh, Pennsylvania.
2008-Present	Graduate Research Assistant, Center for Vision, Learning, Cognition, and Art, UCLA.
2009	Teaching Assistant, Statistics Department, UCLA. Introduction to Probability.
2009	Teaching Assistant, Statistics Department, UCLA. Pattern Recognition and Machine Learning.
2010	M.S. (Computer Science), UCLA, Los Angeles, California.
2010	Teaching Assistant, Statistics Department, UCLA. Pattern Recognition and Machine Learning.
2011	Teaching Assistant, Statistics Department, UCLA. Pattern Recognition and Machine Learning.

Publications

B. Rothrock, S. Park, and S.C. Zhu, "*Integrating Grammar and Segmentation for Human Pose Estimation*", Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), 2013.

B. Rothrock and S.C. Zhu, "*Human Parsing using Stochastic And-Or grammar and Rich Appearance*", Int'l Workshop on Stochastic Image Grammar (SIG), Barcelona, Spain, 2011.

E. Fink, U. Bardak, **B. Rothrock**, and J. G. Carbonell, "*Scheduling with uncertain resources: Collaboration with the user*", IEEE International Conference on Systems, Man and Cybernetics, 2006.

J. Nichols, B. A. Myers, and **B. Rothrock**, "*UNIFORM: automatically generating consistent remote control user interfaces*", Proc. of the SIGCHI conference on Human Factors in computing systems, 2006.

J. Nichols, **B. Rothrock**, D. H. Chau, and B. A. Myers, "*Huddle: automatically generating interfaces for systems of multiple connected appliances*", Proc. of the 19th annual ACM symposium on User interface software and technology, 2006.

J. Wobbrock, B. A. Myers, and **B. Rothrock**, "*Few-key text entry revisited: mnemonic gestures on four keys*", Proc. of the SIGCHI conference on Human Factors in computing systems, 2006.

D. H. Chau, J. Wobbrock, B. A. Myers, and **B. Rothrock**, "*Integrating isometric joysticks into mobile phones for text entry*", CHI extended abstracts on Human

factors in computing systems, 2006.

B. Rothrock, B. A. Myers, and S. H. Wang, "*Unified associative information storage and retrieval*", CHI extended abstracts on Human factors in computing systems, 2006.

J. O. Wobbrock, H. H. Aung, **B. Rothrock**, and B. A. Myers, "*Maximizing the guessability of symbolic input*", CHI extended abstracts on Human Factors in Computing Systems, 2005.

CHAPTER 1

Introduction

1.1 Motivation

According to the ImageNet[1] database from Deng and Fei-Fei [DDS09], which has currently labeled over 14 million images from the internet, the most frequent image label is unsurprisingly "people" (Figure 1.1). Although methods for scene classification have been quite successful to identify images that contain people, there has been much less success in understanding their pose in the image. Accurate pose estimation has broad and far-reaching applications, such as scene understanding, automated surveillance, industrial safety, and human-computer interaction. Humans are particularly well tuned to this task, and can often recognize body pose with near-perfect accuracy even when parts are completely obscured or surrounded by confounders. Modern computer vision models, on the other hand, still fall considerably short of matching human performance.

The challenges for pose estimation lies in appropriately modeling and recognizing the large intra-class variabilities between people. These variabilities are often referred to in terms of shape and appearance variability. The shape of the human body is determined by the articulated configuration of the joints as well as body proportion and clothing, which are then projected to the image plane. The appearance consists of the edge structure of the shape boundary in the image,

[1]source: http://www.image-net.org.

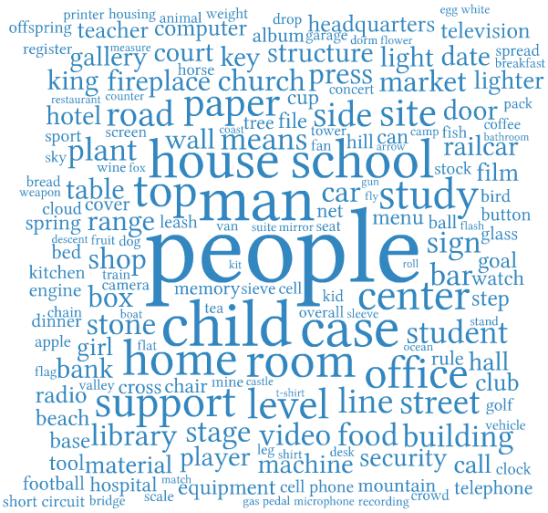

Figure 1.1: In the ImageNet [DDS09] database, which consists of over 14 million annotated images collected from the internet, *people* is the most frequent label.

as well as the regions inside the shape corresponding to colors and texture of the clothing or skin, as well as internal structure such as patterns or creases in clothing. Our work aims to provide a new perspective to modeling the human body in images, motivated from two aspects: articulated models, and image grammars:

1.1.1 Articulated models

Articulated models are part-based models designed to capture the kinematic behavior of bones and joints. Each pair of articulated parts defines a relative location between them for which they can pivot around. This type of model is often referred to in the literature as a pictorial structures (PS) model, originally introduced by Fischler and Elschlager [FE73] and popularized by Felzenszwalb and Huttenlocher [FH05].

As shown in Figure 1.2, the original PS model uses a set of spring-like relations to constrain the relative geometry between parts. Articulation is handled by

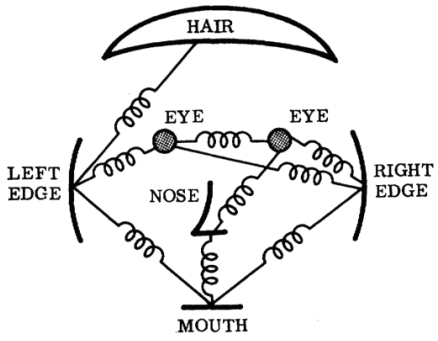

Figure 1.2: Pictorial structures model of Fischler and Elschlager [FE73], which defines spring-like relations to constrain the relative geometries of parts.

placing additional springs on the angular displacement of parts. The model has a canonical rest position for the full arrangement of parts, representing the most typical configuration. To match the template to an image, each of the parts can deform by stretching or compressing their springs to match features on the image. The best location for the object therefore occurs at the position with minimum deformation of the springs.

The PS model is still commonly used in current literature, but exhibits several key limitations that we wish to overcome with our grammar model. We highlight these limitations as follows:

- *All observed examples are treated as a deformation from the canonical position.* This is equivalent to assuming that the distribution of relative positions and orientations for each part is unimodal. This assumption is largely unrealistic for humans in general position, as large angular deformations in the limbs may be just as likely as the canonical pose yet the model must pay a considerable spring penalty to represent these poses.

- *Each part is always visible.* Much of the early work in articulated pose estimation focused only on frontal-view humans where this assumption mostly

holds. In the general case, however, self-occlusion is very prominent. In these cases, the PS model cannot reason that the part is occluded, and must find the best matching location in the image for a part that is not visible.

- *Each part has a common appearance across all examples.* Clothing, in particular, violates this assumption. The same person wearing different types of clothing can exhibit dramatically different appearances for the same part.

- *The size and shape of each part is fixed, relative to the object scale.* Due to perspective and viewpoint, as well as differences in body proportion between people, the size and shape of parts can vary dramatically in the image. The use of fixed-sized shapes for the parts simply cannot accurately represent humans in general position.

1.1.2 Image grammar

In David Marr's seminal work [Mar82], objects are organized in a hierarchy according to the specificity of information they carry, as shown in Figure 1.3. Each composite part therefore has both a coarse and fine representation. The coarse representation summarizes the composition as a whole, and does not explicitly rely on its constituent parts. Conversely, the fine representation explicitly specifies the subparts and their geometric relations. This decomposition is recursive, providing progressively more detailed descriptions until the parts cannot be decomposed further. This is motivated by the multi-scale nature of our own visual system, where we can still identify humans from images in cases where there is insufficient resolution to discern smaller parts.

The image grammar framework of Zhu and Mumford [ZM06], provide a math-

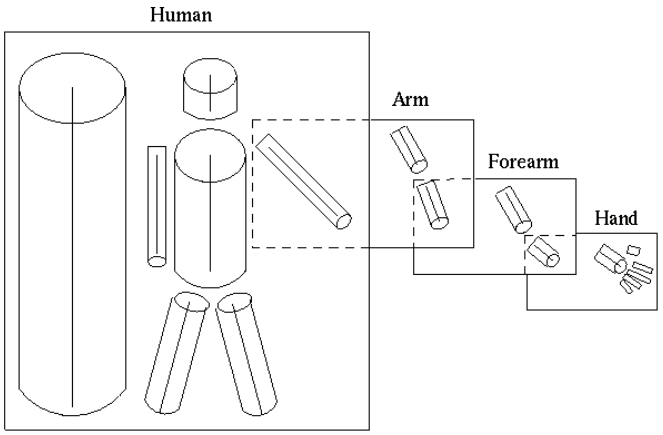

Figure 1.3: David Marr's [Mar82] model for decomposing the human body in a modular and hierarchical manner.

ematical formalism on Marr's construction by formulating a stochastic AND-OR graph grammar. The AND-OR graph is a generative model that represents the image as a hierarchical composition of reconfigurable parts. The AND-nodes in the graph specify a composition of parts, whereas the OR-nodes designates a selection between alternative part variants.

The composite template model of Chen et al. [CXL06] uses an AND-OR graph model applied to the modeling of clothing configurations. In this model, several distinct types of clothing for each body part are identified to have distinct appearance structure. The collar of the polo shirt, for example, has prominent appearance differences between a formal shirt or t-shirt. Figure 1.4 illustrates this decomposition, as well as a fragment of the AND-OR graph for arm. A parse from this grammar is generated by selecting an AND-node for every OR-node recursively, starting from the root node of the grammar. Each AND-node selection generates a parse node, which possess state variables that describe its location in the image, shape parameters, etc. The inference task is therefore to search the space of all valid parses and their corresponding parse node states for

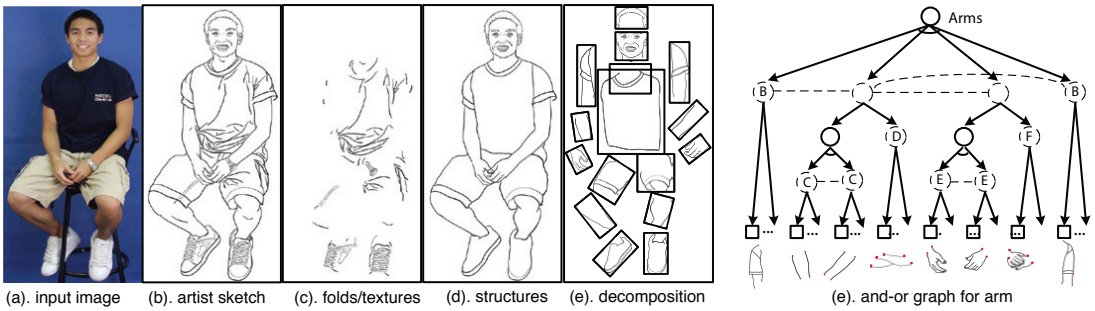

Figure 1.4: Composite Templates for cloth modeling [CXL06]. An AND-OR graph is used to specify the composition rules for selecting a set of part variants to form a valid body configuration.

the configuration with highest probability.

One important limitation to this model is that it assumes the human is in a predefined set of poses. In this sense, the model learns a probability on the appearance and configuration of the body, but not the pose. Without explicit understanding of the articulated kinematics of the body, the number of pose-specific templates to represent the image appearance of all possible human poses would be massive. Such a model would likely be intractable to compute, nor would we have enough data to train it.

Our goal in this work is to integrate, in principle, the articulated nature of the Pictorial Structures model, with the compositional and reconfigurable nature of the AND-OR graph grammar model.

1.2 Grammar models for articulated objects

Our approach utilizes a stochastic grammar model to represent both the articulated geometry of the human body, as well as the highly variant appearances of the parts. This image grammar can be summarized by the following properties:

- *Compositionality*: Complex objects such as the human body often have a natural decomposition into a hierarchy of parts that are semantically meaningful.

- *Modularity*: Each part in the hierarchy is a sub-model in itself. This modular abstraction allows compositions, relations, and properties to be defined recursively. This allows the model to reason about a composite part in the exact same manner as a primitive part.

- *Reconfigurability*: A part may have multiple variants with different shape, appearance, or even substructure. Reconfigurability allows the model to substitute a part for any of its variants. Because these substitutions can occur at any level of the hierarchy, a combinatorial number of compositions can be created with a relatively small number of parts and rules. Noam Chomsky, one of the founders of modern linguistics, describes this property as "infinite generative capacity", referring to our ability to generate an infinite number of meaningful sentences using a compact set of language rules.

- *Shared parts*: A composite part in the grammar can be composed from any other parts. Multiple compositions can therefore use the same part, making it a shared part. For the human body, parts such as the arms and legs have a natural symmetry, and the model can be made more compact by sharing these parts.

- *Contextual relations*: Similar to stochastic text grammars, probabilistic relations determine how frequently each part variant is selected. Unlike text grammars, however, image grammars must also reason about the geometric configuration of the parts – that is, their position and shape in the image.

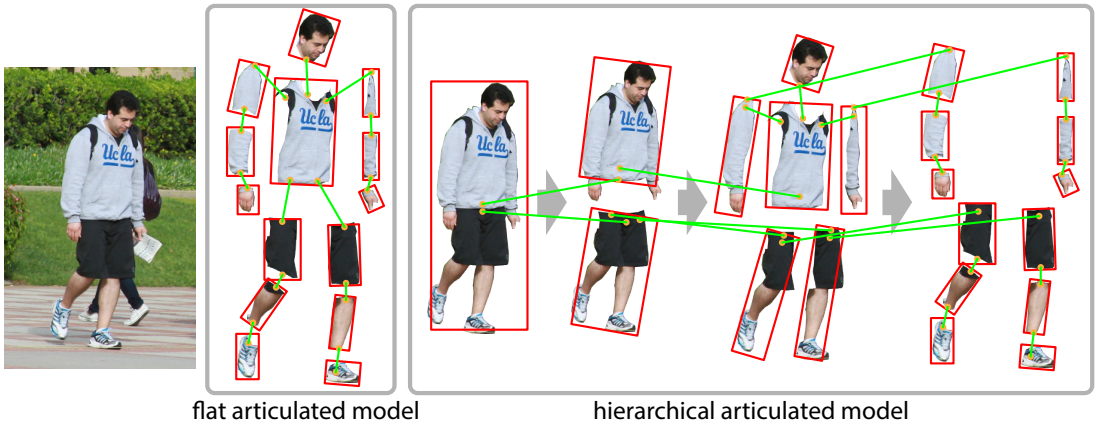

Figure 1.5: Conventional articulated models such as Pictorial Structures [FH05] have a flat structure, connected by relations (green lines) that form the kinematic tree of the body. After incorporating a part hierarchy into this model, additional kinematic constraints must be added to connect coarse-level parts to their fine-level constituents.

For articulated bodies such as human, the most prominent contextual relation between parts are the kinematic constraints that keeps articulated parts connected at their joints. Conventional articulated models such as Pictorial Structures have a flat structure, in that they do not use composition or reconfigurability. Each pair of articulated parts in the model are connected with a kinematic constraint, shown as a green edge in Figure 1.5. These edges naturally form a tree structure to mimic the articulated kinematics of the human body.

Our grammar model uses a compositional representation of parts, meaning that there is a hierarchy of progressive decompositions from whole to part. Each composite part, such as the upper body, is still an articulated part and must be connected with a kinematic relation to either a part on the same level of the hierarchy, or one level coarser. Due to the modularity of the grammar, the composite part is still an articulated part even though it is an abstraction of

smaller parts, for which the joints at which it connects to its neighbors may no longer be an anatomical joint of the body. For this reason, there are several reasonable choices for how to connect the parts of the hierarchical model, one such choice is illustrated on the right side of Figure 1.5.

1.3 Thesis outline

The chapters of this dissertation are organized as follows:

Chapter 2 presents a survey of literature and related work in image grammars and human body modeling, and highlights the shortcomings of current methods.

Chapter 3 introduces our grammar model for articulated structures, and defines all the state variables and probabilistic potential functions used in the following chapters. We also discuss the differences between modeling articulated structures using a phrase-structured grammar and dependency grammar.

Chapter 4 describes generative approaches to learning the human body grammar model. This includes an articulated geometry model learned by the maximum entropy principle, and an appearance model using Active Basis and Hybrid Image Templates. We describe several inference algorithms, and demonstrate experimental parsing and pose estimation results. We also explore ambiguity reasoning of crowded scenes through the use of composite cluster sampling.

Chapter 5 describes a discriminative approach to learning the human body grammar model. This includes a description of the part appearance representation, and training of the model using structured-output learning in an empirical risk minimization framework. We also describe the integration of occlusion and background reasoning into the model, and describe an exact and efficient inference algorithm. Finally, we demonstrate state-of-art performance by evaluating

on several public benchmark datasets for pose estimation.

Chapter 6 concludes the dissertation, and outlines our contributions.

CHAPTER 2

Literature Review

2.1 Methods

2.1.1 Grammar methods

Grammar methods in computer vision have a surprisingly long history, dating at least to the 1970's with the work of Ohta and Kanade [OKS78] and Fu [Fu86]. This initial work was largely conceptual, as the computational tools were not available at the time to adequately explore this topic and the field went dormant for nearly twenty years. In the last decade, however, there has been a resurgence of interest in grammar models in vision.

Beinenstock et al. [BGP97] describes a basic compositional model for object recognition that is recursively composed. Ambiguity is propagated up the model by maintaining multiple interpretations, and ultimately resolved by a minimum-description-length criteria. Amit and Trouv [AT07] describe a "patchwork of parts", which is a edge-based deformable template model where each part is a submodel. Classification is based on likelihood without learning discriminative decision boundaries to achieve competitive performance on digit classification and face detection. Jin and Geman [JG06] also focus on the problem of digit recognition and license plate parsing by developing a "composition machine" to assemble "bricks" that encode local edge structure in a generative probabilistic

model using a Markov "backbone". Zhu and Mumford [ZM06], and Han and Zhu [HZ05] go beyond deformable templates and develop a probabilistic model on the hierarchical decomposition of scenes using AND-OR rules. Aycinena et al. [AKP08] proposed a semi-supervised method for generalizing the inside-outside algorithm to the image grammar case for context-free grammars of semi-rigid parts. AND-OR graph grammars for a wide range of objects including humans were studied by Zhu et al. [ZM06], Chen et al. [CXL06], and Chen, Zhu et al.[CZL07, ZCL08], which is largely the foundation of our own work. Girshick and Felzenszwalb[GFM11] extends the popular discriminative deformable part model of Felzenszwalb et al. [FMR08] into a grammar formalism used for human detection, but not part localization. The hierarchical mixture model of Yang and Ramanan [YR11] differs from our model by replacing articulation geometry with keypoint mixtures, and does not allow reusable or reconfigurable parts. Sun and Savarese [SS11] also uses hierarchical mixtures for modeling articulated bodies, except with restricted coarse-to-fine appearance templates that progressively include an additional articulated part at each level.

2.1.2 Human body models

We can categorize essentially all articulated pose recovery methods by how they model part appearances, how geometries between parts are modeled, and the methods used to compute inference on these models. The ability to compute tractable inference typically dominates the tradeoffs between the complexity of these components.

Appearance models of early techniques were quite simple, and limited to clean images consisting of uncluttered backgrounds and human subjects that were unclad or dressed wearing tight clothing to produce smooth outlines. Ioffe and

Forsythe [IF01] represented body parts as the projection of cylinders onto the image plane, which can be found bottom-up by grouping edges segments into parallel lines, parallel lines into part pairs, and so forth. A similar approach was employed by Ren et al. [RBM05] to find parallel lines through segmentation followed by Delaunay triangulation. Such approaches in finding parallel lines typically fail in complex scenes, as clothing, complex backgrounds, and occlusions cause too many false detections. Pixel-level appearances were also popular in early techniques, using per-pixel color models [SB06], color histograms [EF09], background subtraction [SIS03, FH05], and region-based CRFs [Ram06]. Other region-based approaches such as from Mori and Malik [MRE04] and Srinivasan and Shi [SS07] use super-pixel segmentation to combine small regions granules into parts and body compositions. Most of these pixel and super-pixel models have poor detection rates or do not generalize well, however. Discriminative appearance models using boosting with image features such as SIFT and shape context has had some recent success, such as from Andriluka et al. [ARS09]. The nonparametric kernel technique of Sapp et al. [SJT10] that uses exemplar appearances directly and performs competitively on recent benchmarks. Recently, the Histogram of Oriented Gradient feature of Dalal and Triggs [DT05] have been used with reasonable success in the discriminative compositional models of Girshick et al. [GFM11], and Yang and Ramanan [YR11].

An alternative approach for appearance modeling, motivated primarily from techniques used in full-body pedestrian detection, is to use a set of templates to represent humans in typical poses in a sliding-window framework such as from Mohan et al. [MPP01] and Viola et al. [VJS05]. Early techniques that used this template driven approach could not explicitly recover pose geometry, or generalize well to arbitrary poses. Despite lying dormant for many years, the idea of using multiple templates has seen a resurgence recently with the advent of

modern machine learning techniques. Many current techniques are incorporating large template dictionaries into a pose recovery framework such as the Poselet technique of Bourdev and Malik [BM09].

The use of image-specific background models to improve human pose estimation is an idea that has been revisited many times in recent literature. An iterative learning scheme for CRF appearance models was presented in [Ram06], which incrementally refines a generic part model using image-specific appearance evidence. The work of [FJZ08] utilizes a greedy search space reduction strategy by computing GrabCut segmentations from the bounding boxes produced by a generic upper body detector. Most similar to our approach is the work of [JE09], which efficiently learns a local pixel-based color segmentation model for each proposed part location. Our model, by comparison, uses a global image segmentation as a reference distribution to compute part-based appearance features.

Geometry models controls the relative geometry between parts. The constellation model of Fergus et al. [FPZ03] learns a full joint probability on all parts, but at considerable computational expense. Such fully connected models, as well as methods to improve the computing complexity, were also studied by Bergtholdt et al. [BKS10]. These high-order models models often require large amounts of training data, and are limited to a small number of parts to keep the computation tractable. Factorizing the joint probability into a tree structure imposes conditional independence assumptions between parts, but admits very efficient inference, and was popularized largely by the Pictorial Structures work of Felzenszwalb and Huttenlocher [FH05] and much of its derivative work such as [SB06, EF09, ARS09]. While the tree model is efficient to compute, the independence assumptions can lead to a model that less constrained than the real body. A class of "loopy" models attempts to strengthen the tree-model with non-

tree constraints such that the impact on inference is not as great as the densely connected models. The work in [RBM05] explores pairwise geometric constraints between arbitrary parts using Gaussian relations. Similarly, [JM08] augments the kinematic tree with logical exclusion relations to admit efficient inference. Tian et al. Tran and Forsyth [TF10] explored using a conventional pictorial structures model with fully connected relations at significant computational cost. [TZN12] utilized mixtures of higher order geometries using a latent variable geometry model to express more strongly constrained poses without adding considerable computational cost.

Inference techniques are critical to the viability of any particular model, as designing a model to capture a large number of constraints is relatively easy, but learning and computing inference on that model is often very non-trivial. As such, there is considerable variety in the techniques used within the literature of human pose recovery. The popularity of tree-models is largely due to the efficiency of belief propagation [SIS03, ARS09] and distance transforms [FH05]. Data-driven Markov chain Monte Carlo was used in [LC04, ZLL06]. Many of the techniques using loopy models were able to formulate inference in terms of quadratic programming [RBM05], and dynamic programming [JM08]. The fully connected model of [BKS10] employed A* heuristic search. Of the grammar techniques, [AKP08, FM10] uses dynamic programming to search over the geometry and productions, whereas [CZL07, CXL06] use a compositional bottom-up/top-down search.

2.2 Datasets

UCLA pedestrian. We collect our own dataset of 400 high resolution pedestrians in walking and standing poses, from mostly frontal viewpoints (head not

Figure 2.1: UCLA Pedestrian: our own dataset of 400 images of pedestrians in walking and standing poses. The body geometry is annotated by 15 terminal parts, and 7 nonterminal parts, for a total of 22 part annotations per body.

Figure 2.2: Buffy Stickmen: images from the television show *Buffy the Vampire Slayer* in general pose. Annotations are for the upper body using 6 line segments.

turned backwards). Each image contains exactly one annotated body. Each of the 15 body parts are annotated with a truncated cone with high quality alignments. Our model also uses 7 composite parts that have their geometries automatically labeled by the annotation tool as a function of the constituent part geometries. Each of the 22 total parts are also annotated by a label indicating its appearance class, which is required by the grammar model. The dataset is illustrated in figure 2.1.

Figure 2.3: Pascal Stickmen: this data set contains images from the PASCAL VOC 2008 of humans in general pose. Annotations are for the upper body using 6 line segments.

Buffy Stickmen. This dataset is collected from medium resolution still frames of the television show *Buffy the Vampire Slayer*, and collected for the pose recovery methods of Ferrari at al.[FJZ08] and Eichner et al.[EF09]. Each image has exactly one human annotated for the upper body only, using 6 line segments for the head, torso, upper and lower arms. The segments roughly correspond to the lengths and centerlines of each part. The dataset consists of 748 images containing a total of 4488 part annotations, illustrated in figure 2.2.

Pascal Stickmen. This dataset was collected as a supplement to "Buffy Stickmen" by the same research group, and contains annotations of the same format. The images are medium resolution and contain people in general pose from the PASCAL VOC 2008 trainval release [EGW08]. The data set consists of 549 images containing a total of 3294 part annotations, illustrated in 2.3.

PARSE. This dataset was collected for the iterative image parsing method of Ramanan[Ram06]. The dataset consists of low resolution images of the full body, with poses predominantly involving athletic activity. Each image contains exactly one annotation of the full body using 14 positions of the joints. There are 305 images containing a total of 4270 joint annotations, illustrated in 2.4.

Figure 2.4: PARSE: 305 full body images in general pose, annotated with 14 joint locations per body.

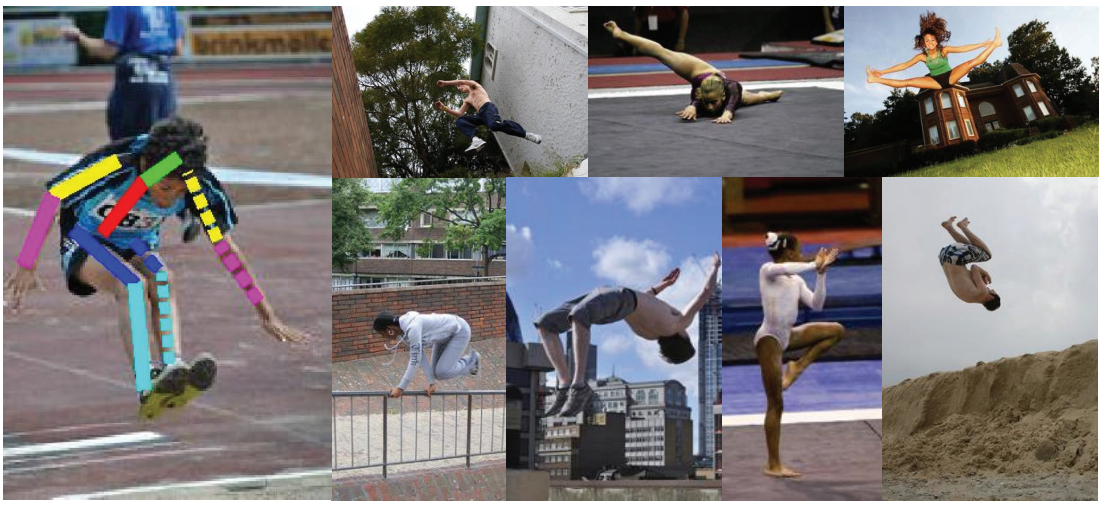

Figure 2.5: Leeds Sports Pose: 11,000 full body images in general pose, annotated with 14 joint locations per body.

Figure 2.6: H3D Poselets: full body images in general pose, annotated using a 3D model by matching the locations of 19 keypoints per body.

Leeds Sports Pose. This dataset was collecting using Amazon Mechanical Turk by Johnson and Everingham [JE11]. The dataset is similar to PARSE, containing low resolution images of the full body. Images in this dataset are closely cropped, and collected from web search terms for "parkour", "gymnastics", and "athletics". Each image contains exactly one annotation of the full body using 14 positions of the joints, illustrated in 2.5.

Humans in 3D. This dataset was collected for the full body pose recovery method of Bourdev and Malik[BM09]. Images are high resolution and contain full body people in general pose. Annotations are performed using a tool to match a 3D articulated model to the image using 19 keypoints for the joints and face. There are 429 image containing a total of 8151 joint annotations, illustrated in 2.6.

2.3 Evaluation

Throughout this work, the focus is on developing improved techniques for human pose estimation. To estimate the quality of these techniques, there has been several protocols adopted by the academic community in order to publish a fair comparison between methods. In this section we review the motivations behind these protocols, and describe the specific variants that are used in this work.

Many of the public human pose datasets are not consistent on which parts to annotate, and how those annotations are represented. Some datasets only annotate the upper-body, as opposed to the full-body. Some represent the body as 14 parts to include the hands and feet, and some only use 10 parts. Given one of these representations, we assume that each image is fully annotated with every part, and there are no missing parts. Occluded or truncated, i.e. parts that appear off the image frame, are still annotated using the best guess of the annotator.

The part annotations are typically defined as a bone, parameterized by the image coordinates of both endpoints. Parts that have multiple occurrences, such as from the arms and legs, are designated left and right and annotated according to some convention of left and right. Typically, the part on the right-most side of the image is designated as the right part, as opposed to from the reference frame of the subject in the image.

Any pose estimation algorithm must therefore output a set of bones that are consistent with the body representation defined by the dataset. Each bone is evaluated independently according to a localization criteria, and the localization rate for each part is reported. This localization rate is coined as the Percentage of Correctly estimate body Parts, or PCP by Eichner and Ferrari [EF09]. The

overall performance of the algorithm is typically reported as the average PCP across all parts.

This general approach leaves many ambiguities, however, that has spawned a large number of variants that produce dramatically different performance numbers. Direct comparison with other techniques in the literature is therefore quite difficult without a very specific specification of the protocol, or a common evaluation code base.

The main points of variation between evaluation protocols are summarized below:

- *Part detection criteria*: typically the part is considered detected if the alignment of the bone endpoints are below a threshold. There are two main variants for this, the first being that the average distance of the predicted endpoints is less than some percentage of the ground-truth bone length, typically set to 0.5. The other common variant requires that both endpoints are below the threshold by replacing the average with the max endpoint distance.

- *Human detection filtering*: some test images may contain multiple people, only one of which is annotated as the ground-truth. If the pose estimator outputs the correct pose for the wrong person in this case, it will be counted as a total failure. To address this, the pose estimator is asked to output a large number of ranked poses. A bounding box of the full human is computed as the envelope of all part boundaries, and the highest ranking pose within some overlap criteria of the human boundary is selected to evaluate against. A common overlap criteria is 50% intersection-over-union, commonly used by VOC [EGW08] for object detection. To further generalize this, bounding boxes for all humans in the test images can be annotated, for

which a detection rate can be computed from the ranked list of predicted poses. The average PCP performance can then be scaled by the detection rate to provide a measurement of both detection and localization.

- *Short and zero-length parts*: it is a common occurrence in human images for a part to point directly into or away from the camera, effectively shortening the bone length in the image to a very small length or even zero. In such a case, in order to detect the part the predicted endpoints must be within half of this length, forcing the prediction to nearly exactly match the ground-truth. Conversely, for parts that are very long and narrow, such as an arm parallel to the image plane, the prediction can be nowhere near the ground-truth and will still be counted as correct. Part of this problem lies in the omission of the part width to compute the localization quality. Because only the centerlines of the bones are annotated, this remains an outstanding problem for which an appreciable number of zero-length parts are essentially undetectable according to this criteria.

- *Object symmetry*: each dataset chooses its own convention for how to annotate left and right parts, typically corresponding to the left and right sides of the image. As such, if a prediction localizes the body correctly but reverses left with right, then all the limb parts will be scored as failures. This is particularly troubling given that roughly 80% of the total score is composed from the limb scores. Furthermore, every public benchmark dataset we have encountered exhibits an appreciable number of annotation inconsistencies, where the left-right labeling of parts does not match their own convention. Such inconsistencies arbitrarily corrupts whatever performance measure being used, and remains an outstanding problem with these datasets.

- *Occluded parts:* parts that are occluded, or not visible because they appear off the edge of the image are still annotated and evaluated against. This is particularly problematic because the location of the non-visible part annotations have been hallucinated by the annotator. The true part may be in a number of viable locations, but it is obviously uncertain. Nevertheless, the algorithm must still predict the precise locations of these human annotations and is penalized if wrong. Many of the current benchmark datasets include a binary flag to indicate if a part or a joint is occluded, but none of the standard performance evaluations currently use them.

We define two variants of the part detection criteria. The first requires that the average distance between the predicted and ground-truth endpoints are below a threshold t, typically defined to be half the length of the ground-truth part. Let d_p and d_d be the endpoint distance between the prediction and ground-truth for the proximal and distal sides respectively. The part is therefore considered correctly localized if $\frac{d_p+d_d}{2} < t$. The second variant is stricter by requiring both endpoint distances to be below the threshold, and is written $\max(d_p, d_d) < t$.

Following the convention described in [YR11], all evaluations will be one of the following four protocols:

- *single-avg*: selects the single best parse for the image, and compares each part against the ground truth by thresholding the average endpoint distance.

- *single-both*: selects the single best parse for the image, and compares each part against the ground truth by thresholding the maximum endpoint distance.

- *match-avg*: this is the most common case in the literature, and we assume

this variant is used unless mentioned otherwise. This protocol uses the maximal bounding rectangle of all the ground truth parts to select the highest scoring parse from a list of n-best parses output by the inference algorithm being evaluated. After selecting the parse, each part is evaluated against the ground truth by thresholding the average endpoint distance.

- *match-both*: this protocol uses the same selection criteria as match-avg, but compares each part using the maximum endpoint distance.

CHAPTER 3

Grammar Models for Articulated Structures

3.1 Overview

Our articulated grammar model provides a unified probabilistic framework to decompose the body into modular parts while maintaining the articulated kinematics between parts. The grammar takes the form of an AND-OR graph, denoted as

$$\mathcal{G} = (S, P, s_0) \tag{3.1}$$

where S is a set of symbols (OR-nodes), P is a set of productions (AND-nodes), and s_0 is the root symbol. Each production $p \in P$ is a rule that decomposes a symbol into a set of constituent symbols, and takes the form $(\alpha \rightarrow \beta, t, R)$. The symbol $\alpha \in S$ is referred to as the proximal symbol, whereas the set of constituent symbols $\beta \subset S$ is referred to as the distal symbols. t is an appearance template for α. R is a set of probabilistic relations that control the spatial geometry and compatibility between α and β, and thus expresses contextual information about neighboring parts.

In cognitive neuroscience, the human visual cortex is largely treated as a bottom-up process motivated by experimental observations of local spatial filtering in V1, and grouping processes in V2 and onward to identify contours and structures. Computational theories to emulate this process have been largely pop-

ularized by Marr [Mar82]. In contrast, there is also evidence for a large number of backwards connections in the visual context, implying some form of top-down process occurring in the brain before all the bottom-up signals have been fully processed. Furthermore, experiments have shown that humans can still make reasonable predictions for objects of such low resolution that any smaller parts are indiscernible. Models for this top-down process, such as from Bar [Bar03] provide a complementary explanation of both processes working simultaneously. Conventional grammars, such as those used for text, only connect to the data through the terminal symbols. The productions, in this case, only control the arrangement of terminals in the sentence and have no direct representation in the data. The nonterminal symbols in this case, such as a noun phrase or prepositional phrase, are abstractions in the sense that they can only be observed by looking for their components in a bottom-up fashion. Images and the objects within them, on the other hand, are subject to the phenomenon of scale where certain parts are only observable above some corresponding level of resolution.

In this light, the grammar models we develop are intended to represent both low-level/high-resolution features that can be combined using the compositional machinery of the grammar, as well as high-level/low-resolution features that complement these compositions. For this reason, our grammar makes no distinction between terminal and nonterminal symbols, except for the fact that terminal symbols have no productions to further decompose them. Instead, each production $p \in P$ represents a part at some resolution, and contains an appearance template t that represents the features of that part. Terminal parts are effectively represented as productions with no children, which take the form $(\alpha \to \emptyset, t, R)$. Therefore, every part in the grammar, both terminal and nonterminal, defines a template to connect to the image data, as well as a set of contextual relations R specific to that part and its children, if any.

Part reconfigurability is naturally represented in the grammar by the presence of multiple production rules that expand the same symbol. A part can also behave both like a nonterminal and terminal by having productions with and without children respectively. Similarly, productions can be defined with the same set of child symbols, but different relations R that specify distinct geometric configurations of the same child parts.

Part sharing occurs whenever two or more productions use the same distal symbol. The advantages of part sharing are threefold: (i) sharing parts reduces the total number of parts in the grammar, directly reducing the number of parameters that must be learned in the model, (ii) sharing inherently provides more examples to train on for that part, and (iii) computation during learning and inference are reduced. Furthermore, both terminal and nonterminal symbols can be shared, resulting in a potentially large reduction in both model complexity and computational time.

We represent the grammar as a graphical structure called an AND-OR graph, an example of which is shown in Figure 3.1. At the root is the OR-node symbol s_0. Each OR-node is represented as a set of all productions (AND-nodes) that expand it, in other words, each of these AND-node productions have the OR-node symbol as the left-hand-side (α) symbol. The OR-nodes therefore enumerate all the different variants that can be interchanged with that part. In the case of the human body, it is common for the variants within an OR-node to have the same child structure, but different appearance and contextual relations.

The language of the grammar is the set of all parses the grammar can generate. A parse is derived by traversing the grammar $\mathcal{G}$ top-down from the root symbol s_0 and selecting a single AND-node production for every OR-node. This process then continues to the children of the selected production. For each production

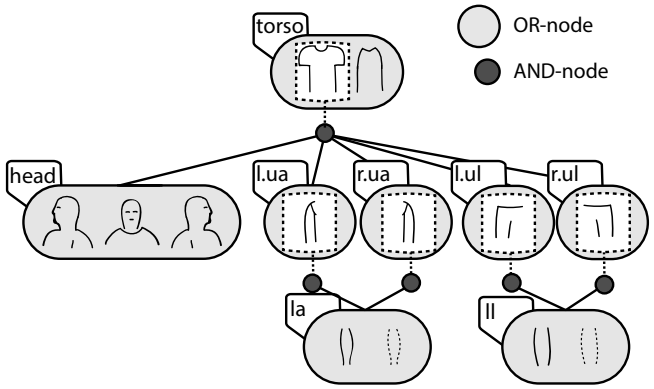

Figure 3.1: AND-OR graph example for the upper-body. OR-nodes are the symbols of the grammar, and represented as a collection of AND-nodes. The AND-nodes are the productions of the grammar, and contain edges that connect to their constituent child (OR-node) symbols. It is common for all the AND-nodes within an OR-node to have the same child structure, in which case only a single AND-node is expanded for clarity.

selected in this process, a parse node v is instantiated. Edges are then placed between instantiated parse nodes according to the contextual relations R from their corresponding production. The state of each parse node is parameterized as

$$v = (\omega, x, y, \theta, \ell, s). \tag{3.2}$$

The variable ω indexes the production used to instantiate the parse node. The remaining variables are for position x, y, orientation θ, aspect ratio ℓ, and scale s.

Parses from the grammar are referred to as *parse graphs* due to the relation edges between the nodes, and denoted as pg. Each production that gets selected during a derivation instantiates a unique parse node into the parse graph. If the grammar uses shared parts, and the productions for these parts are used multiple

times in a derivation, a parse node with its own unique state variables is created for each selection.

In this work, we use two different types of grammatical constructions to model the human body, called phrase-structured grammars and dependency grammars. These constructions relate to how parts are composed, and consequently how their geometries are related. These grammars are described in Section 3.3.1 and 3.3.2 respectively.

3.2 Probability model

Let $V(pg)$ be the set of parse nodes, and $E(pg)$ be the set of edges for parse graph pg. For each production $(\alpha \rightarrow \beta, t, R)$ selected in the derivation, the relations in R are constrained to be only between the parse nodes corresponding to the α and β symbols. This locality of relations defines a neighborhood system of the grammar, and each production can be viewed as a self-contained sub-model between a part and its constituents.

The probability on parses is formulated in a Bayesian framework, which computes the joint posterior as the product of a likelihood and prior probability, and equivalently represented as the following Gibbs distribution

$$p(pg|I;\lambda) \propto p(pg;\lambda)p(I|pg;\lambda) = \frac{1}{Z}\exp\{-\mathcal{E}(pg;\lambda) - \mathcal{E}(I|pg;\lambda)\}. \tag{3.3}$$

The model parameters are denoted as λ. The energy functions $\mathcal{E}$ are further decomposed into a set of potential function. These potentials constrain all aspects of the grammar, from the selection of productions during derivation, to the geometric compatibility between parts, to the image appearance likelihood of parts. The choice and particular forms for these potential functions vary on the design and intention of the grammar, which we explore in the following chapters. In the

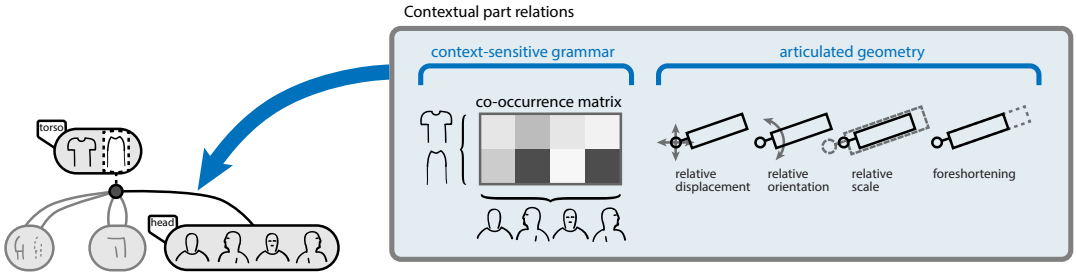

Figure 3.2: Contextual relations are specific to each AND-node production, and define constraints both on the selection frequency of the productions, as well as the geometry of the parts.

following sections, we describe the role for each type of potential. An illustration for many of these potential functions is shown in Figure 3.2.

3.2.1 Parse node geometric state representation

To simply computations over part states and avoid the complications of learning probability distributions over continuous orientations, the full geometric state of each parse node x, y, θ, ℓ, s is discretized. Position x, y is typically subsampled to 25% of the image dimensions. θ are discrete orientations, typically using 24 increments spanning all angles over 2π. The number of scales s and aspect ratios ℓ are typically limited to 5.

The image is computed on a pyramid of logarithmic scales such that there are a predefined number of levels per octave, typically 6. Part scales all reference these image level scales. To assess the scale of a parsed object, a reference symbol (OR-node) is used as a representative part for which all other parts in the parse are relative to. This is typically the root symbol of the grammar, and its width is used as a normalizing factor for all other parts so as not to be affected by changes in the aspect ratio of the root part.

Each symbol (OR-node) in the grammar has a canonical width associated with it, obtained from the training annotations. For a given ground truth parse annotation, the width for all parts are measured as a percentage of the reference parse node width. The canonical width of each symbol is then computed as the mean relative width among all observations for that symbol. Similarly, a set of discrete aspect ratios for each symbol are selected to span the range of observed aspect ratios for that symbol. These aspect ratios are then stored in a table, and indexed by the variable ℓ.

3.2.2 Context-free potentials

A derivation from stochastic context-free grammars (SCFG), such as those used for text language modeling, is a random branching process parameterized by selection frequencies for each production. For example, in the trivial grammar defined by the two productions:

$$0.7 \;\; VP \rightarrow V \;\; NP$$

$$0.3 \;\; VP \rightarrow V \;\; NP \;\; NP$$

The symbol VP is expanded using the first production with frequency 0.7, and the second production with frequency 0.3. The production frequencies are normalized to sum to one for all productions that expand the same symbol. They represent the conditional probabilities that a production is used given the occurrence of the symbol in a parse.

This context-free production frequency is incorporated into the image grammar by adding a potential function on the variables ω for each parse node. In the case of the text grammar above, the production frequency is now written $p(\omega = 1) = 0.7$, where $\omega = 1$ indicates that the first production in the list was

used to expand the symbol. The potential function is written with the convention f^{c1}, indicating that it is a unary contextual potential, and defined as

$$f^{c1}(pg;\lambda) = \sum_{v \in V(pg)} \psi(\omega;\lambda). \tag{3.4}$$

Where $\psi(\omega;\lambda)$ is a function that scores the production selection ω using model parameters λ. The specific form of these potential functions will be described in detail in the following chapters.

3.2.3 Context-sensitive potentials

The context-free assumption that all productions are selected independently is overly naive, and doesn't allow for correlations and compatibilities between the productions of neighboring parts to be captured by the grammar. For example, in the grammar illustrated in Figure 3.2, there are two torso productions for side-view and front-view, and four productions for head corresponding to left/right/front/back viewpoints. Any combination of these two parts is possible, as the body can often be facing one way and looking another. There is likely strong correlation, however, that the side-view torso favors the left/right head, and the front-view torso favors the front/back head. A SCFG grammar will select the productions for these two symbols independently, however, without taking into account their correlation.

The context-sensitive potential captures these dependencies by using pairwise co-occurrence probabilities between parse node pairs:

$$f^{c2}(pg;\lambda) = \sum_{(v_i,v_j) \in E(pg)} \psi(\omega_i, \omega_j;\lambda). \tag{3.5}$$

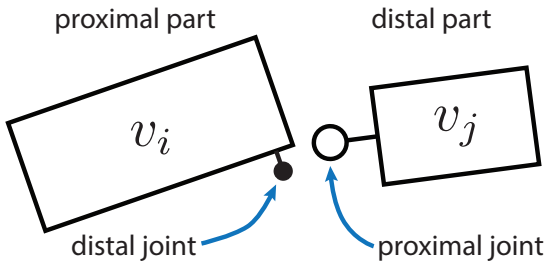

Figure 3.3: Each part in an articulated pair are referred to as a proximal and distal part, each defining a distal and proximal joint location between them where they connect.

3.2.4 Kinematic potentials

The kinematic relations control the geometric arrangement of parts relative to each other. Most prominently in the case of articulated bodies in the requirement that connected parts revolve around a common hinge point. The human body naturally forms a kinematic tree, for which we arbitrarily define the root part to be the torso. We refer to a proximal joint as a connection point to a part closer to the kinematic root of the body, and a distal joint as one that connects a part further away from the kinematic root and toward the extremities.

The kinematic relation connecting a pair of articulated parts therefore defines two joint locations, one for each part. Each joint location is defined in the relative coordinate system of its corresponding part. When referencing an articulated pair, the parts are identified as a proximal and distal part, each connected through their distal and proximal joints respectively, as shown in Figure 3.3.

The locations of these joint positions are not specified in the annotations, and must be inferred. Annotations are far from perfect, however, and will rarely be consistent with a rigid articulated body such that the distal and proximal joint locations between each articulated pair always coincide at the same location.

Instead, parts must be "dislocated" slightly away from their joints to match the annotations. The best joint locations are therefore computed such as to minimize the amount of dislocation.

The articulated joints are defined in the relative coordinate system of their part, and normalized by the length and width of that part. Let x, y, θ be the center location and orientation of a part, $J^{(l)}, J^{(w)}$ be the relative joint location in the length and width dimension of the part respectively, and l, w be the annotated length and width of an oriented part. Under rigid articulation, the displacement of the part locations can be computed purely from their orientations and relative joint locations:

$$\begin{aligned} x_j - x_i &= J_{l_i} l_i \cos(\theta_i) - J_{w_i} w_i \sin(\theta_i) - J_{l_j} l_j \cos(\theta_j) + J_{w_j} w_j \sin(\theta_j) \\ y_j - y_i &= J_{l_i} l_i \sin(\theta_i) + J_{w_i} w_i \cos(\theta_i) - J_{l_j} l_j \sin(\theta_j) - J_{w_j} w_j \cos(\theta_j). \end{aligned} \tag{3.6}$$

The part displacements $x_j - x_i$ and $y_j - y_i$ are linear functions of the four unknown relative joint parameters $(J_{l_i}, J_{w_i}, J_{l_j}, J_{w_j})$. This is expressed as the linear system $d = XJ$ for displacements d, joint locations J, and factors X from equation 3.6. With N training examples, the residuals d is a $2N \times 1$ column vector, J is a 4×1 column vector, and X is a $2N \times 4$ matrix. The joint locations that minimize the square residuals can then be computed by least squares as $J = (X^\top X)^{-1}(X^\top d)$.

The kinematic relations between a pair of articulated parts has five components: orientation, aspect ratio, relative orientation, relative scale, and joint displacement. The orientation and aspect ratios are the prior probabilities for each part independently, and written as

$$f^{g1}(pg; \lambda) = \sum_{v \in V(pg)} \left(\psi(\theta; \lambda) + \psi(\ell; \lambda) \right). \tag{3.7}$$

Relative orientation over k discrete orientations between θ_i and θ_j is computed as $d\theta := (\theta_j - \theta_i) \mod k$. Relative scale is similarly denoted $ds := s_j - s_i$. To

compute the joint displacement, we first define a coordinate transformation to the joint locations of both proximal and distal parts:

$$T_{dist}(v) = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} J^{(l)}_{dist} \cdot l \\ J^{(w)}_{dist} \cdot w \end{bmatrix}$$
$$T_{prox}(v) = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} J^{(l)}_{prox} \cdot l \\ J^{(w)}_{prox} \cdot w \end{bmatrix} \tag{3.8}$$

The length and width l, w of parse node v can be computed from the appropriate table lookups using the aspect ratio and scale states ℓ, s. The joint displacements between a pair of articulated parts (v_i, v_j) can now be computed as the difference of transformed locations from the perspective of both proximal and distal parts $dL := T_{prox}(v_j) - T_{dist}(v_i)$. The potential function for the relative states is then

$$f^{g2}(pg; \lambda) = \sum_{(v_i, v_j) \in E(pg)} (\psi(d\theta; \lambda) + \psi(ds; \lambda) + \psi(dL; \lambda)) \tag{3.9}$$

3.2.5 Appearance potentials

The appearance potentials represent the image likelihood of the parse. As in most part-based models, this likelihood is decomposed into a product of conditionally independent part likelihoods given their geometries.

$$f^{a1}(I|pg; \lambda) = \sum_{v \in V(pg)} \psi(I|v; \lambda) \tag{3.10}$$

Multiple types of appearance models are used throughout this work, and the specifics behind how each of them represent and compute the image likelihood will be described in detail in the following chapters for generative and discriminative models.

3.3 Grammar relational structure

For a given production $(\alpha \to \beta, t, R)$, the symbol α is referred to as the root part, and the set of symbols β as the child parts. For many types of text grammars, the "geometry" of the root part of a production is the start and end position of the part in the sentence, which is computed as a deterministic function of its child parts. In the image grammar case, however, the geometry of each part is represented as an oriented rectangle, for which there is typically not a meaningful deterministic function to compute the root part geometry from its children. For example, how should the geometry of the upper-body part in Figure 3.4(b) change when the arms are outstretched? Instead, the same probabilistic relations that are used to constrain the child parts are also used to constrain the root part to one or more of its children.

When designing a grammar for articulated bodies, there are multiple strategies for decomposing parts into subparts, and consequently placing relations between those parts. This must all be done while preserving the articulated kinematics of the body. We describe two approaches that have a direct analogy to grammar types used in linguistics: phrase-structured grammars, and dependency grammars.

3.3.1 Phrase-structured grammars

Phrase-structured grammars, also known as constituency grammars, use the grammar productions to define phrase structure rules. In text grammars, these phrases contain their constituents in that they explicitly represent a contiguous fragment of the sentence consisting of the recursive union of all its children. This type of decomposition naturally encodes a coarse-to-fine or whole-to-part nature

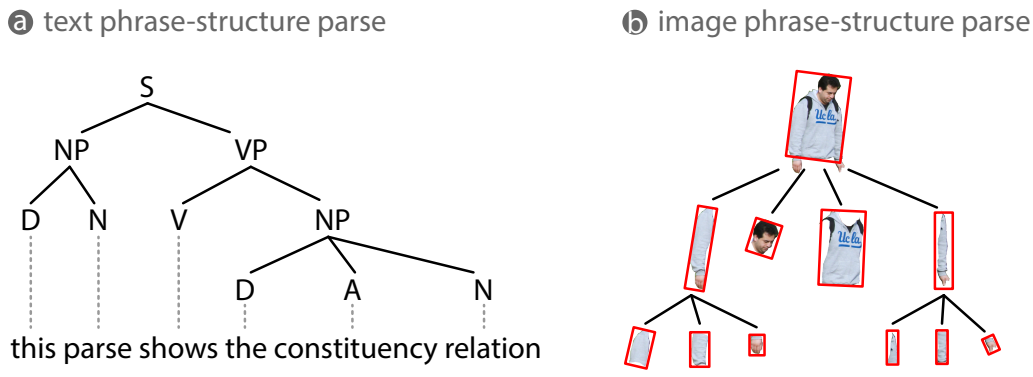

Figure 3.4: Parses from a phrase-structured grammar for both text and images. Phrase-structured grammars represent a coarse-to-fine decomposition of parts, unlike dependency grammars that organizes the data into a flat dependency structure.

of the data.

As illustrated in Figure 3.4(a), the structure of the text sentence in a phrase-structured grammar is composed from noun phrases and verb phrases. Similarly, in Figure 3.4(b), the image regions of the upper body is composed from regions containing the full arms, which are in turn composed from the regions of each arm segment. Note that each parse node is represented with an appearance and rectangular geometry, even for the non-terminal nodes in this case.

While the phrase-structure of the productions defines how a part decomposes into child parts, we must also specify which parts to place relations between. We first consider the case where all the relations are strictly between the root part and a child part, commonly referred to as a star-structured part model, and illustrated in Figure 3.5(a). This type of relation structure is the most popular among hierarchical part-based models in computer vision due to its simplicity, but problematic for the case of articulated objects. By forcing the child parts to be articulated with their coarser-level parent leads to a very unnatural motion of the body. For

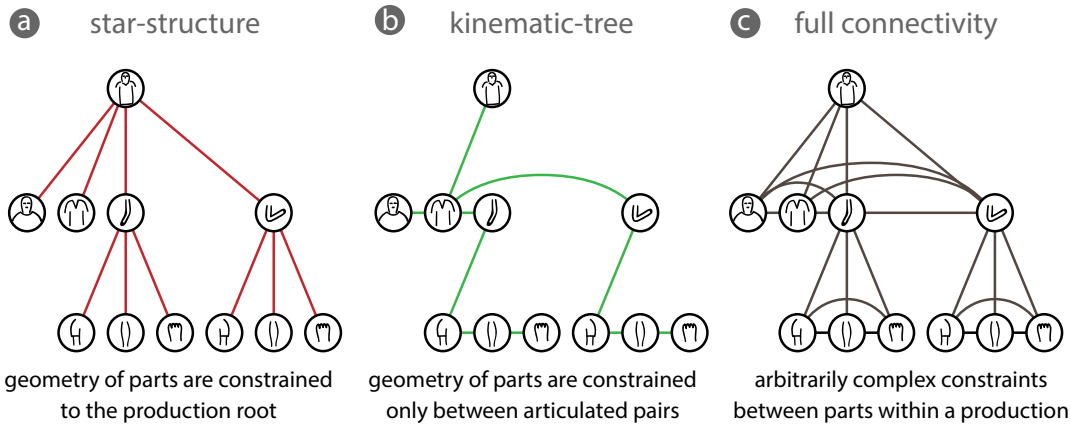

Figure 3.5: Comparison of relation structures for a production. In the phrase-structure grammar, each child part is related to the production root resulting in a star-model in the simplest case (a) and a fully connected model in the most complex case (c).

example, consider the arm production (arm $\rightarrow$ {upper-arm, lower-arm, hand}). In this case, a child part such as the hand would hinge off of the coarser-level arm part, independent of where the upper or lower arm parts are, leading to a very unnatural model that does not preserve the articulated motion of the arm.

Alternatively, as shown in Figure 3.5(b) relations can be placed among the children according to their natural kinematics. In this case, the root part must still be related to at least one child part. We usually choose to place this relation between the root part and the most proximal child part, as these two parts typically have the strongest geometric coupling. For example, the root part in the arm production is likely well aligned with the shoulder joint location, as is the upper-arm child part. By placing a relation between these two parts enforces that the articulated chain of child parts is essentially connected at the shoulder with the coarse-level part that summarizes the whole arm as a single rectangle. Another way to view this construction is that the production summarizes the

ensemble of multiple articulated child parts into a single coarse-level articulated part, for which they have a single joint location in common determined by which child part this root-to-child relation is placed on. This relational structure is guaranteed to form a tree, which has particular importance in keeping the computational complexity of learning and inference tractable.

Using the kinematic-tree structure for the relations is the simplest viable choice, in that it uses the fewest number of relations while preserving the body kinematics. Lastly, we consider the most general case where relations are placed between arbitrary pairs of part, leading to the fully-connected model in Figure 3.5(c). This model is the most expressive, but at a considerable cost to inference and learning due to the cycles that are added in the relation graph.

3.3.2 Dependency grammars

Unlike phrase-structured grammars which use productions to define containment of their children, dependency grammars use production rules to declare dependency relations between parts. For text dependency grammars, the verb is typically the central element of the sentence and all other words are linked either directly or indirectly to the verb, illustrated by the example parse in Figure 3.6(a).

One of the key differences between dependency grammars and phrase-structured grammars is that there is no abstraction in the way that phrase-structured grammars summarize a collection of parts into a single abstract non-terminal part. In a dependency grammar, there is a one-to-one correspondence between the parse nodes and the elements of the data. To illustrate this, every parse node of the dependency parse in Figure 3.6(a) has a dotted line linking it directly to a word, whereas only terminal parse nodes in the phrase-structure parse in Figure 3.4(a) have such a link. In the image case, only one parse node can represent an im-

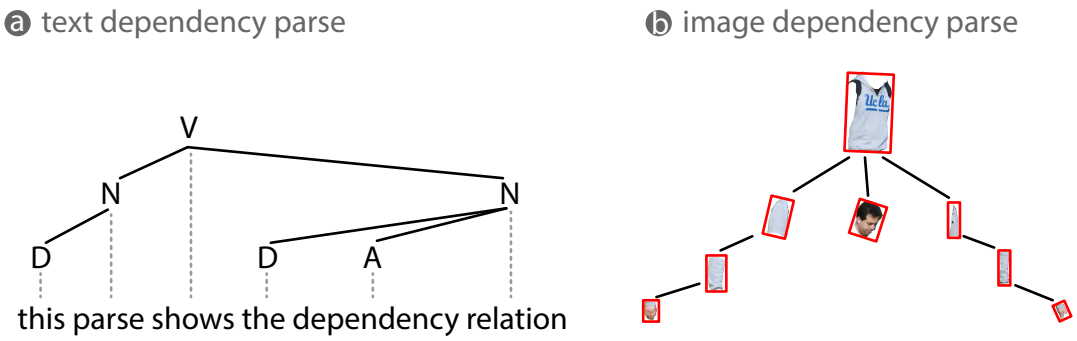

Figure 3.6: Parses from a dependency grammar for both text and images. The dependency grammar does not summarize compositions into coarse-level parts like phrase-structure grammars. Instead, each parse node directly links to an element of the data.

age region corresponding to a part, whereas in the phrase-structure case multiple parse nodes from different productions can represent the same image region at multiple levels of resolutions.

A parse for an image dependency grammar is illustrated in Figure 3.6(b), where the dependencies are consistent with the kinematic relations between parts. One of the obvious differences between the phrase-structure grammar in the image case is the lack of coarse-to-fine representation. Furthermore, the relational structure of the dependency grammar always relates the child directly to the production root (e.g. star-structured) simplifying the design over using a phrase-structure grammar.

CHAPTER 4

Generative Methods

In this chapter we explore learning the posterior distribution of the grammar model as an explicit factorization of likelihood and prior models, and trained via maximum likelihood. First we explore training the prior model inspired by the minimax entropy principle of Zhu et al. [ZM06]. For the image appearance models, we use a generative deformable template model based on the Active Basis model of Wu et al. [WSF07] and Hybrid Image Template model of Si and Zhu [SZ12]. We demonstrate the model through random sampling, as well with parsing performance evaluation. We also explore discriminative re-estimation of the model, ambiguity reasoning through cluster sampling, and an approximate inference algorithm based on n-best reranking.

4.1 Learning the prior through maximum entropy

In a Bayesian framework where the posterior probability has been factorized into a prior $p(pg; \lambda)$ and likelihood $p(I|pg; \lambda)$, the prior is often called the shape model and represents the distribution over parse labels. The probability on parse labels includes the probability of deriving the parse from the grammar, as well as the probability on the geometric arrangement of the resulting parse nodes.

We assume we have a set of observed images and their corresponding ground-

truth parses, drawn from an underlying target distribution f:

$$D^{obs} = \{(I_i^{obs}, pg_i^{obs}) : i = 1, 2, \ldots, N\} \sim f(I, pg) \tag{4.1}$$

The language of the grammar, which is the set of every valid parse, is denoted Ω_{pg}, and the domain of all images is Ω_I.

Motivated by the models described by Zhu et al. [ZWM98, ZM06], the grammar probability model can be derived from the maximum entropy principle under the constraints of matching relation statistics from the training data. Let $\phi(pg)$ be a marginal statistic of a parse pg. These statistics measure properties of the object for which we wish to constrain, which includes the selection frequency of each production, and the geometric arrangements of the parse nodes.

We wish to learn the model such that the expectation of the statistics $\phi(pg)$ match those from the underlying target distribution f. Let Ω be the set of all probability distributions that match these constraints:

$$\Omega = \{p(pg; \lambda) | E_p[\phi_i(pg)] = \mu_i^{obs}, \quad i = 1, \ldots, k\} \tag{4.2}$$

$$\mu_i^{obs} = \frac{1}{N} \sum_{pg \in D^{obs}} \phi_i(pg) \tag{4.3}$$

The maximum entropy (ME) principle embodies Occom's razor, and states that the best distribution to select from this family is the simplest one. In other words, given that all distributions in the family match the constraints we are interested in, the one with maximum entropy is the most unprejudiced among them. The ME distribution is defined as:

$$p^*(pg; \lambda) = \arg\max \left\{ - \int p(pg) \log p(pg) dpg \right\} \tag{4.4}$$

subject to

$$E_p[\phi_i(pg)] = \int \phi_i(pg)p(pg)dpg = \mu_i, \quad i = 1...k \tag{4.5}$$

$$\int p(pg)dx = 1. \tag{4.6}$$

The first constraint matches the relation statistics, the second ensures that the probability distribution is well formed by summing to one. Solving for $p^*(pg)$ by the method of Lagrange multipliers leads to the following distribution

$$p(pg;\lambda) = \frac{1}{Z(\lambda)} \exp\left\{ -\sum_{i=1}^{k} \lambda^{(i)}(\phi_i(pg)) \right\}. \tag{4.7}$$

For the set of general relations, there is no closed form for the functions $\lambda^{(i)}(\cdot)$. As a result, we treat the potential functions non-parametrically and rewrite the model as

$$p(pg;\lambda) = \frac{1}{Z(\lambda)} \exp\left\{ -\sum_{i=1}^{k} \langle \lambda^{(i)}, h_i(pg) \rangle \right\}. \tag{4.8}$$

The parameter $\lambda^{(i)}$ is now a vector with an element for every bin in histogram $h_i(pg)$ populated from the responses from $\phi_i(pg)$.

To estimate the parameters λ, we minimize the KL-divergence between our model p and the underlying distribution f, across the domain of all parse graphs Ω_{pg},

$$p^* = \arg\min KL(f||p) = \arg\min \sum_{pg \in \Omega_{pg}} f(pg) \log \frac{f(pg)}{p(pg;\lambda)}. \tag{4.9}$$

This is equivalent to the maximum likelihood estimation for parameters λ:

$$\lambda^* = \arg\max_{\lambda} \sum_{i=1}^{N} \log p(pg_i;\lambda). \tag{4.10}$$

given a set of N observed parse graphs pg_i. The parameters λ are optimized using the stochastic gradient method of [ZWM98]. In maximizing the likelihood function in equation 4.10 leads to the following implicit constraint:

$$E_{p(pg;\lambda)}[h_i(pg)] = h_i^{obs}. \tag{4.11}$$

This constraint states that the expectations of the histograms for each marginal statistic must match the histograms of the observed parses. For modeling articulated structures, we introduce a set of these constraints into the model, corresponding to the potentials defined in Chapter 3. The log-likelihood of Equation 4.10 is then written to include these constraints:

$$\mathcal{L}(\lambda) = \sum_{i=1}^{N} \log p(pg_i; \lambda) = -\sum_{i=1}^{k} \langle \lambda^{(i)}, h_i(pg) \rangle - \log Z(\lambda) \tag{4.12}$$

$$= f^{c1}(pg; \lambda) + f^{c2}(pg; \lambda) + f^{g1}(pg; \lambda) + f^{g2}(pg; \lambda) - \log Z(\lambda) \tag{4.13}$$

$$f^{c1}(pg; \lambda) = -\sum_{v \in V(pg)} \langle \lambda^{(c1)}, h^{(c1)}(\omega) \rangle \tag{4.14}$$

$$f^{c2}(pg; \lambda) = -\sum_{(v_i, v_j) \in V(pg)} \langle \lambda^{(c2)}, h^{(c2)}(\omega_i, \omega_j) \rangle \tag{4.15}$$

$$f^{g1}(pg; \lambda) = -\sum_{v \in V(pg)} \langle \lambda^{(\theta)}, h^{(\theta)}(\theta) \rangle \tag{4.16}$$

$$f^{g2}(pg; \lambda) = -\sum_{(v_i, v_j) \in E(pg)} \langle \lambda^{(d\theta)}, h^{(d\theta)}(d\theta) \rangle + \langle \lambda^{(ds)}, h^{(ds)}(s_j - s_i) \rangle + \langle \lambda^{(dL)}, h^{(dL)}(dL) \rangle \tag{4.17}$$

$$\lambda^* = \arg\max_{\lambda} \mathcal{L}(\lambda) \tag{4.18}$$

To control the selection of productions when generating a parse from the grammar, we wish to match the selection frequency, as well as the co-occurrence frequency of the productions with the observed data. Each parse node in a parse uses the state variable ω to indicate which production was used to generate the node. In the AND-OR notation, the variable ω can be viewed as the index of the AND-node contained within the OR-node being expanded. Let $s(v)$ indicate the OR-node symbol of parse node v, and $n(v)$ indicate the total number of AND-node choices to expand $s(v)$. For each production $z \in P$, we compute the frequency the production is selected within the parse, relative to the total number of choices available from the enclosing OR-node symbol. This relative frequency

is computed as

$$h^{(c1,obs)}(z) = \frac{\sum_{pg \in D^{obs}} \#(\omega = z|pg)}{\sum_{p \in S(z)} \sum_{pg \in D^{obs}} \#(\omega = p|pg)}, \quad \forall z \in P. \tag{4.19}$$

The function $\#(\omega = i|pg)$ counts the number of occurrences where symbol v is rewritten with production $(\omega = i)$ in the parse pg. The expression $p \in S(z)$ refers to the set of production choices from the OR-node symbol expanded by production z. Similarly, we can measure the relative co-occurrence frequency of production pairs as follows:

$$h^{(c2,obs)}(z_i, z_j) = \frac{\sum_{pg \in D^{obs}} \#(\omega_i = z_i, \omega_j = z_j|pg)}{\sum_{(p_i,p_j) \in (S(z_i),S(z_j))} \sum_{pg \in D^{obs}} \#(\omega_i = p_i, \omega_j = p_j|pg)}, \quad \forall (z_i, z_j) \in P \times P. \tag{4.20}$$

The productions z_i and z_j can be viewed as parent and child parse nodes in the parse graph, respectively.

Next are the geometry statistics. Given that a parse has been derived from the grammar, we wish to measure important properties about the geometric states (x, y, θ, ℓ, s) of all the parse nodes. For all relation edges in the parse graph $E(pg)$, we assume that the relative geometries $d\theta, ds, dL$ are computed as defined in Section 3.2.4. The statistics we measure correspond to part orientation θ, relative orientation between articulated pairs $d\theta$, relative scale ds, and relative joint displacement dL.

The structure of the geometric relations are not limited to a Markov-tree structure, such as those illustrated in Figure 3.5(a,b). A Markov random field, as illustrated in Figure 3.5(c), can be defined by introducing additional pairwise relations into the geometry potential f^{g2} in Equation 4.18.

The parameters λ are obtained by maximum likelihood estimation by setting $\frac{d\mathcal{L}(\lambda)}{d\lambda} = 0$ and solving for λ by Lagrange multipliers. As shown in Chi and Geman [CG98] and Zhu et al. [ZM06], the maximum-likelihood solution for the

Algorithm 1 Parameter estimation of the geometry model

procedure LEARNMAXENT(grammar $\mathcal{G}$, dataset $D^{obs} = \{pg_i^{obs}, i = 1, \ldots, N\}$, stopping condition ϵ)
  Compute histograms h_i^{obs} from D^{obs} for each geometry relation.
  Initialize each λ_i to a zero-vector.
  repeat
    Sample a collection of parse graphs $D^{syn} = \{pg_j^{syn}, j = 1, \ldots, M\}$ from $\mathcal{G}$ using the current model parameters $\lambda^{(t)}$.
    Compute histograms h_i^{syn} from D^{syn} for each relation.
    $\lambda_i^{(t+1)} \leftarrow \lambda_i^{(t)} \cdot \eta(h_i^{syn} - h_i^{obs})$ for each relation.
  until $\max_i |h_i^{syn} - h_i^{obs}| < \epsilon$
  return λ
end procedure

context-free parameters are very simply and intuitively the same as the observed production frequencies:

$$\lambda^{(c1)}(z) = -\log h^{(c1,obs)}(z). \tag{4.21}$$

Similarly, the context-sensitive parameters are as follows:

$$\lambda^{(c2)}(z_i, z_j) = -\log\left(\frac{h^{(c2,obs)}(z_i, z_j)}{h^{(c1,obs)}(z_i)h^{(c1,obs)}(z_j)}\right). \tag{4.22}$$

The context-sensitive terms encode the compatibility of neighboring production variables. Incorporating the context-sensitive potentials in this form to the context-free model effectively makes the production selection a first-order Markov process.

The remaining parameters on the geometric relations have no analytical solution, but can be estimated by stochastic gradients. The stochastic gradient method requires random sampling of the model to generate a set of synthetic

parses. Relation statistics are then measured on these synthesized samples and compared with the observed statistics to incrementally update the λ parameters. This process begins by sampling an ensemble of parse graphs D^{syn} from the grammar $\mathcal{G}$ using the context-free and context-sensitive parameters estimated from the observed parses D^{obs}. The geometric configuration (x, y, θ, ℓ, s) of the parse nodes for each of these synthetic parses are then sampled from the model using the current estimates of parameters λ. The marginal statistics of the synthesized ensemble are then compared against the observed parses to compute a stochastic gradient, which is used to update the model parameters. The algorithm continues to sample and update until the statistics between D^{syn} and D^{obs} match sufficiently closely. This algorithm is outlined in Algorithm 1. An intuitive explanation of this algorithm is analysis-by-synthesis, in which the model repeatedly makes predictions on what the data should look like, identifies any mistakes from looking at actual observed data, and uses this information to improve the model and make better predictions in the next iteration.

Sampling is performed using sequential Monte Carlo and Gibbs sampling. First, a parse graph pg is derived from the grammar G by recursively selecting a production for every OR-node symbol, starting from the root symbol s_0. Productions are selected with probability $h^{c1,obs}(z)$ for some production z among all available choices from the OR-node. For each production chosen, the geometric state (x, y, θ, ℓ, s) for each of the production children are conditionally sampled along the dimensions of $d\theta$, ds and dL using Gibbs sampling. If the graph of geometric relations has cycles, the Gibbs sampling step is repeated multiple times for each production. For the start symbol, the geometry is Gibbs sampled using only the unary terms f^{g1}. The sampling and maximum entropy learning is illustrated for the first 5 iterations in Figure 4.1. Lastly, the normalizing constant $Z(\lambda)$ is dependent only on the grammar and constant for all parse graphs, and is

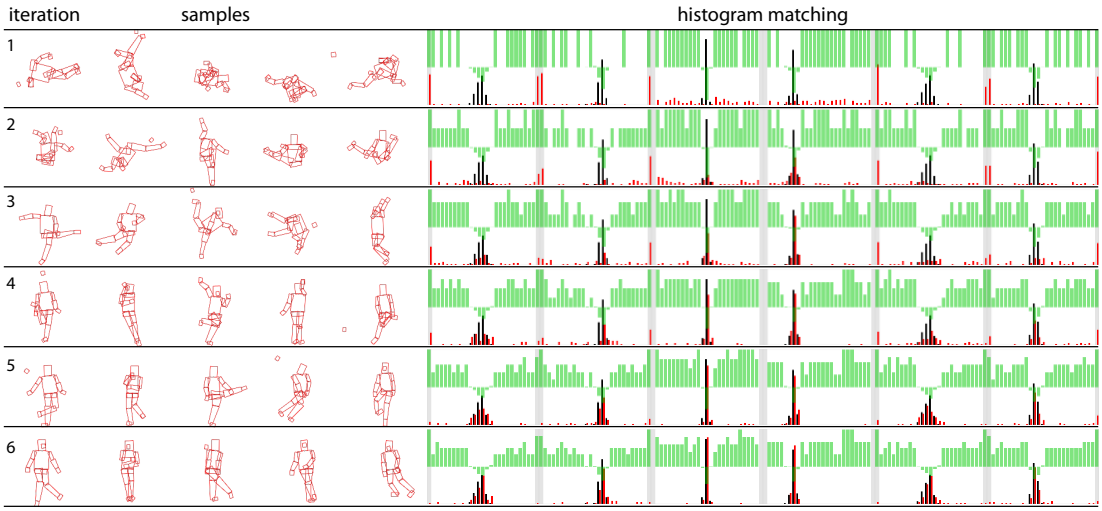

Figure 4.1: The first 6 iterations of the parameter estimation algorithm used to train the prior model. On the left are the first five poses from the synthesized ensemble. On the right are marginal histograms from the first six relations. Black bars in the histograms correspond to the observed frequencies in the training data. Red bars correspond to the synthesized statistics from the sampled ensemble. The green bars represent the parameter vector λ for each relation, locally normalized to fit within the plot. After each iteration, the synthesized and observed statistics match more closely, and the corresponding pose samples begin to look more representative of the observed data.

therefore omitted from the total energy because it does not affect the sampling, learning, or minimization during inference.

4.2 Image likelihood models

The image likelihood term $p(I|pg;\lambda)$, often called the appearance model, represents the conditional probability of the image pixels given a parse label. A common assumption among part-based models is to assume that the appearance of each part occurs independently of each other, given their geometry. This assumption allows us to factorize the parse likelihood into individual part likelihoods that are conditionally independent:

$$p(I|pg) = q(I_{\overline{\Lambda_{pg}}}) \prod_{v \in pg} p(\mathbf{I}_{\Lambda_v}|v). \tag{4.23}$$

The image region occupied by parse node v is denoted I_{Λ_v}, and the image region not occupied by any of the parse nodes is $I_{\overline{\Lambda_{pg}}}$ and also called the background region. Recall the productions of our grammar are defined as $(\alpha \to \beta, t, R)$. The appearance template t defines the part appearance model for the α symbol of the production.

In this section we explore part likelihood models based on the Active Basis model of Wu et. al [WSF07] and Hybrid Image Template model of Si and Zhu [SZ12].

4.2.1 Active Basis and Hybrid Image Templates

The Active Basis model [WSF07] is a generative image appearance model motivated by sparse coding. The model takes the form of a deformable template of basis elements that combine to reconstruct the image region in a linear-additive

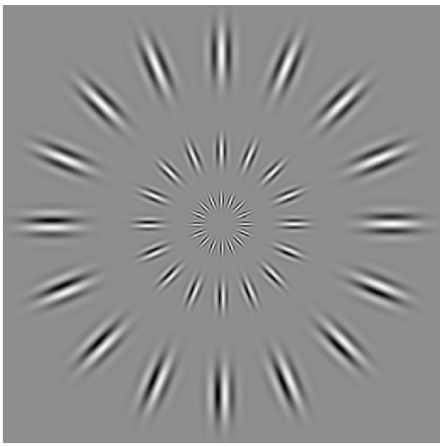

Figure 4.2: Gabor basis filters at various locations, orientations, and scales.

manner:

$$I_\Lambda = \sum_{i=1}^{k} c_i B_i + U. \tag{4.24}$$

The template consists of k basis elements B_i with weight coefficients c_i. U is the unexplained residual image. The basis elements consist of Gabor filters at various locations, orientations, and scales, illustrated in Figure 4.2. One of the distinguishing properties of the Active Basis model is that each basis elements can perturb locally by shifting in position or orientation, allowing the template to deform slightly to compensate for structural variabilities in the appearance of parts.

Let $\{I_{\Lambda_m}, m = 1, \ldots, M\}$ be the collection of aligned image regions corresponding to all M observations of some production ω, and $q(I_m)$ be a reference background distribution. Active Basis learns the template by incrementally adding basis elements using a shared sketch algorithm that combines matching pursuit (Mallat and Zhang [MZ93]) with projection pursuit (Friedman [Fri87]). The shared sketch algorithm incrementally adds a basis element to the model and updates the density function. At each iteration, matching pursuit is used to

select a Gabor basis element B_i that best explains the unexplained residual U. The model is then updated using projection pursuit by applying the pooled density of the basis coefficient $p_i(c)$ from $\{c_{m,i}, m = 1, \ldots, M\}$ and using a density substitution scheme. The resulting model after n iterations is:

$$p(I_{\Lambda_m}|v) = q(I_m) \prod_{i=1}^{n} \frac{p_i(c_{m,i}|v)}{q(c_{m,i})}. \tag{4.25}$$

The Hybrid Image Template (HIT) model of Si and Zhu [SZ12] generalizes Active Basis by expanding the dictionary of basis elements to include arbitrary image features, called prototypes. One of the limitations of Active Basis is that it only focuses on the edge structure, or sketch, of objects and is unable to represent or explain regions that are flat, textured, or colored. The HIT model explicitly incorporates features to represent these heterogeneous image phenomena, and is learned in a similar feature pursuit framework.

The dictionary of feature prototypes consists of each feature type (such as edge, color, texture, etc.) at each location and orientation (if relevant) in the part image patch I_Λ. Let B_k be a given prototype from the dictionary, and $r_k(I_\Lambda)$ be a scalar response from the prototype applied to the image. The specific prototypes used and their response calculations will be discussed further on.

By the maximum entropy principle, we wish to learn a distribution $p(I_{\Lambda_m}|v)$ that approximates the underlying true part appearance distribution $f(I_\Lambda|v)$ by minimizing the Kullback-Leibler divergence between the two distributions $KL(f||p)$ under the constraints that the model matches the observed statistics $E_p[r_k] = E_f[r_k]$. A reference distribution q is used as the initial model, and each iteration of adding a basis element brings the model closer to the underlying distribution $q = p_0 \rightarrow p_1 \rightarrow ... \rightarrow p_k \sim f$. This is a constrained optimization problem to minimize the KL between each increment, while satisfying the constraints that

the expectations of the prototype responses match:

$$p_k^* = \arg\min KL(p_k||p_{k-1}), \quad s.t.\ E_{p_k}[r_k] = E_f[r_k]. \tag{4.26}$$

The solution of this optimization is the following log-linear model

$$p_k(I_\Lambda|v) = q(I_\Lambda)\exp\left\{\sum_{i=1}^{k}\lambda_i r_i(I_{\Lambda_i}) - \log z_i\right\} \tag{4.27}$$

with Lagrange multipliers λ_i for each appearance prototype r_i. z_i is the normalizing constant for each prototype, and can be computed empirically from background image patches $I_{bg,\Lambda}$

$$z_i \approx \frac{1}{N}\sum_{j=1}^{N}\exp\{\lambda_i r_i(I_{bg,\Lambda_j})\} \tag{4.28}$$

At each iteration, an appearance prototype is added to the model such that the $KL(p_k||p_{k-1})$ is maximized between the model with and without the feature. This is defined as the information gain of the appearance prototype, and can be approximated from the positive examples:

$$IG(r_k) = KL(p_k||p_{k-1}) = \lambda_k E_f[r_k] - \log z_k \approx \lambda_k \frac{1}{M}\sum_{i=1}^{M} r_k(I_{\Lambda_i}) - \log z_k \tag{4.29}$$

To learn the template, a dictionary is populated to contain a large number of candidate appearance prototypes at all positions in the template. At each iteration, the prototype with maximum information gain is selected and added to the template. The learning process terminates when the information gain falls below a prescribed threshold. This framework assumes that each selected prototype is uncorrelated with the prototypes already in the template. To enforce this, once a prototype is selected from the dictionary, all correlated prototypes in the dictionary are removed. The HIT learning algorithm is summarized in Algorithm 2.

Next, we describe how responses are computed for each of the prototypes used in our model.

Sketch: sketch prototypes are represented by Gabor basis functions $B_{x,y,\theta}$ at specific locations and orientations. The Gabor response is computed by convolving the basis with the image. The location and orientation of each basis are also allowed to perturb over a local neighborhood to give some invariance to small deformation. A sigmoid function S is used to saturate extreme responses, and I_{Λ_i} is a small image patch enclosing the basis and its deformation range.The response is therefore calculated as a maximum over these deformations:

$$r^{skt}(\mathbf{I}_{\Lambda_i}) = \max_{\delta x,\delta y,\delta\theta} S(||\langle \mathbf{I}_{\Lambda_i}, B_{x+\delta x,y+\delta y,\theta+\delta\theta}\rangle||^2). \tag{4.30}$$

Texture: regions of medium contrast are considered texture. Again using Gabor filters at multiple orientations, we construct a texture prototype t by transforming each bin of the response histogram with a normal distribution favor a medium contrast response. The dictionary of texture prototypes is populated by pooling the training data and computing a candidate prototype at every location. Finally, the response is computed by taking the intersection of the transformed histograms:

$$r^{tex}(\mathbf{I}_{\Lambda_i}) = \sum_{i=1}^{|t|} \min(t_i, t'_i(\mathbf{I}_{\Lambda_i})). \tag{4.31}$$

Flat: regions of essentially no contrast are considered flat. Similar to the texture case, a bank of Gabor filters is used at multiple orientations. The response is computed by taking the maximum response over all orientations and applying a reverse sigmoid transformation to favor a near-zero contrast. This result is averaged over a local neighborhood, and the local max is taken to produce the final result. Using S' as the reverse sigmoid and avg() as the average over the

local neighborhood:

$$r^{flt}(\mathbf{I}_{\Lambda_i}) = \max_{\delta x, \delta y} \text{avg} \left(\max_{\theta} S'(||\langle \mathbf{I}_{\Lambda_i}, B_{x+\delta x, y+\delta y, \theta} \rangle||^2) \right). \tag{4.32}$$

Color primitives: color prototypes consist of 2d histograms pooled from a fixed patch location in the training images. The color space used for the histograms is (h, s) from a HSV cylinder. We compute a pooled prototype histogram h'_i over all the training data, and compute the response of the prototype against an observed image patch by using the histogram intersection:

$$r^{clr}(\mathbf{I}_{\Lambda_i}) = \sum_{i=1}^{|h|} \min(h_i, h'_i(\mathbf{I}_{\Lambda_i})). \tag{4.33}$$

The cumulative information gains are illustrated for all templates in our model in figure 4.12. The colors represent the relative proportion each prototype contributes for that part. All part templates in our model are trained with two sketch prototypes, one at a fine scale (typically a 9x9 Gabor) and one at a course scale (17x17).

4.2.2 Background features

Both Active Basis and HIT are designed around the pursuit and modeling of features that occur more commonly in the foreground (e.g. the appearance of a part or object), than the background of natural images. This can be problematic when attempting to detect parts or objects in cluttered background, as is commonly the case for human detection and pose estimation. For example, the template for an arm segment predominantly contains sketch elements that form a pair of parallel lines. Such parallel lines can be found commonly in natural images, such as in images of brick walls or hatched fences. In these cases, the background appearances contain additional features, notably in the form of edges orthogonal

Algorithm 2 Learning the HIT appearance model

procedure LEARNHIT($v, \mathbf{I}_{\Lambda_i}, i = 1..M, \mathbf{I}_{bg,\Lambda_j}, j = 1..N$)

  $\mathcal{A}_v \leftarrow \emptyset$ ▷ initialize template to empty set

  Populate dictionary Δ with appearance prototypes at all orientations and locations

  for each prototype $r \in \Delta$ **do**

    Compute information gain $IG(r)$ using positive images

  end for

  repeat

    Select r_k with maximum information gain

    Compute λ_k and z_k using negative images

    $\mathcal{A}_v = \mathcal{A}_v \cup r_k$ ▷ add feature r_k to template

    Inhibit all prototypes in Δ correlated with r_k

  until $IG(r_k) < threshold$

end procedure

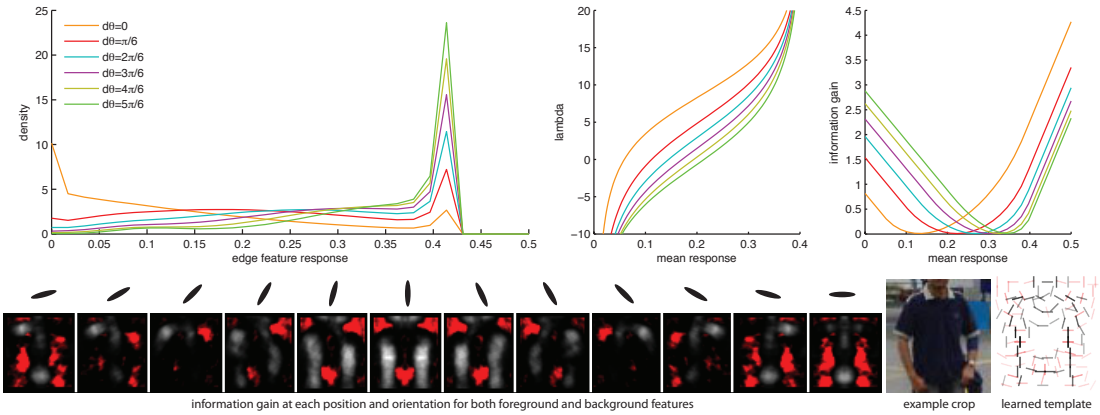

Figure 4.3: Some feature prototypes are more representative of the background than the foreground. The distribution of edge responses over different levels of orientation maxing is shown for a set of natural background images. $d\theta$ indicates the range of orientations each of the edge features are maximized over. Features at high maxing are informative at low means which correspond to a negative lambda, making them features of the background. The use of these features allows the model to better discriminate against the background by identifying locations and orientations that should be absent of features. The information gain map on the bottom is computed from the mean feature responses of all the torso parts, the feature with the least maxing is in white, the most maxing is in red. The template on the right is learned by sequentially selecting the location and orientation of either the foreground or background feature corresponding to the highest information gain.

to the foreground edges that would occur rarely in a true foreground example. These features are what we refer to as background features, in that their presence should reduce the likelihood that an image patch belongs to a positive example.

An HIT template containing background features should therefore describe what the part should look like, as well as what it should not look like. In that light, we make the following two modifications to the HIT model. First, each feature prototype can define its own local activity (e.g. deformation range). For features such as the Gabor, multiple instances are placed in the dictionary, each with a different local activity range in both orientation and location. Second, the value of the λ parameter of the exponential model for a given prototype is allowed to be negative.

To illustrate this, we consider the Gabor feature of Active Basis, and vary the local activity in orientation only. Orientations are discretized to 24 increments, so the first prototype is simply the Gabor response, the second prototype computes the max response over ± 1 orientation increment, the third prototype maxes over ± 2 orientation increments and so forth. In Figure 4.3, the response histograms for the resulting prototype responses over a large set of natural images. As we max over a larger number of orientations, the mean response shifts higher. Consequently, information gain occurs at a lower mean response for prototypes with larger activity. At the bottom of Figure 4.3, for a collection of torso regions, the information gain maps are for each orientation of the Gabor with no activity in white, and the Gabor with activity over ± 5 orientations in red. From these maps we can see that the prototype with large activity has considerable information in the orthogonal orientation from the prototype with no activity. After running the pursuit algorithm, the resulting template is shown on the bottom-right of Figure 4.3, where elements with negative λ's are drawn in red.

4.2.3 Discriminative parameter re-estimation

The parameters of the HIT template are learned under a generative framework to minimize reconstruction error on image intensities. While this approach accurately explains the image data, classification and detection performance tend to fall behind discriminative models trained to minimize classification error. Motivated from the work in [Zha10], we use logistic regression to discriminatively re-estimate the parameters of the HIT model. The logistic regression model takes the form

$$p(y|\mathbf{x}) = \frac{1}{1 + \exp\{-y(\lambda^T \mathbf{x} + b)\}} \tag{4.34}$$

and is trained by minimizing the following regularized negative log-likelihood

$$C \sum_{i=1}^{l} \log(1 + \exp\{-y(\lambda^T \mathbf{x}_i + b)\}) + \frac{1}{2}\lambda^T \lambda. \tag{4.35}$$

y represents the class ± 1, l is the number of training examples, b is the bias, and C is a penalty parameter. The feature vector $\mathbf{x}$ consist of the HIT responses for all k prototypes in the template $(r_{i,0}, r_{i,1}, ..., r_{i,k})$. Once λ and b are estimated using this criteria, the λ's can be substituted directly back into the HIT model, and the bias can be distributed evenly across all k features by setting each $z = b/k$.

The improvement of this parameter reestimation on classification of test data is shown in figure 4.4 for several parts that are particularly difficult to detect. This approach allows us to learn a sparse representation through a generative model, and then refine the parameters of the model to improve classification rate. The optimizer from [FCH08] is used to estimate the logistic regression parameters using the penalty parameter $C = .001$ for all parts.

The likelihood of the full parse is computed by substituting the HIT part

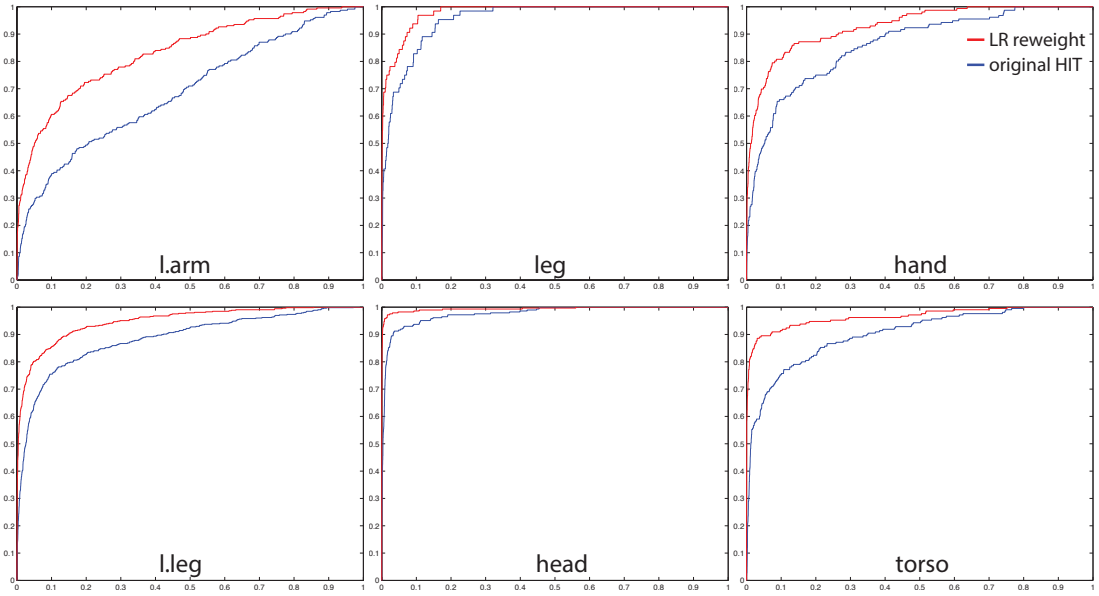

Figure 4.4: HIT performance gain from parameter re-estimation by logistic regression. ROC curves are shown for several selected part productions with MLE estimation in blue and LR estimation in red.

likelihood in Equation 4.27 into the parse likelihood in Equation 4.23 to get

$$p(\mathbf{I}|pg) = q(\mathbf{I}_{\overline{\Lambda_{pg}}}) \prod_{v \in V(pg)} q(I_{\Lambda_i}) \exp\left\{\sum_{j=1}^{k} \lambda_j r_j(I_{\Lambda_j}) - \log z_j\right\}$$
$$= q(I) \prod_{v \in V(pg)} \exp\left\{\sum_{j=1}^{k} \lambda_j r_j(I_{\Lambda_j}) - \log z_j\right\}. \quad (4.36)$$

Because $q(I)$ is constant for any given image I, it does not affect the energy minimization during inference and is dropped.

4.3 Parsing

The parsing task is to find the parse with the maximal posterior probability, also called the Viterbi parse in linguistics:

$$pg^* = \arg\max_{pg} p(pg|I). \quad (4.37)$$

In this section, we present several inference strategies for parsing images. The problem can be equivalently stated as energy minimization, or score maximization where a parse score is simply the negative energy.

We assume the grammar model is phrase-structured, with general relations as illustrated in Figure 3.5. This is the most general case, which encompasses phrase-structured articulated-tree grammars as well as dependency grammars. Due to the hierarchical nature of the grammar, the score can be formulated recursively. We then present a best-first search strategy, an approximation using dynamic programming (DP), and finally an adjustment using a n-best reranking approach.

4.3.1 Recursive scoring of parses

We refer to the parse score as the negative energy, or equivalently the log-probability. The model has a log-linear form, for which the score can be computed as a sum of potential function responses. The grammar naturally encodes a modular structure of sub-models, for which we can compute the score recursively as a function of smaller sub-parse scores. Let $s(v, I)$ denote the cumulative score of parse node v, which includes all parse nodes below v. Furthermore, let $C(v)$ denote the set of children of v. In other words, given the production $(\alpha \to \beta)$ used to rewrite some OR-node symbol α into parse node v, $C(v)$ is the set of parse nodes rewritten by some other productions corresponding to the set of OR-node symbols β. We omit the normalization constant in the score, as it does not affect the maximization in Equation 4.37. Let the the parse node v_0 denote the root part, corresponding to the start symbol s_0 for some parse graph pg. The score of the root part is therefore proportional to the log-probability

$$\log p(pg|I) \propto s(v_0, I). \tag{4.38}$$

and can be decomposed as

$$s(pg|I) = f^a(pg|I;\lambda) + f^{c1}(pg;\lambda) + f^{c2}(pg;\lambda) + f^{g1}(pg;\lambda) + f^{g2}(pg;\lambda) \quad (4.39)$$

$$s(v,I) = g(v,I) + \overbrace{\sum_{v_j \in C(v)} s(v_j,I)}^{\text{child sub-score}} \quad (4.40)$$

$$g(v,I) = \overbrace{s^a(v,I)}^{\text{appearance}} + \overbrace{s^{c1}(v) + \sum_{v_j \in C(v)} s^{c2}(v,v_j)}^{\text{grammar}} + \overbrace{s^{g1}(v) + \sum_{v_j \in C(v)} + s^{g2}(v,v_j)}^{\text{geometry}} \quad (4.41)$$

$$s^{c1}(v) = \langle \lambda^{(c1)}, h^{(c1)}(\omega) \rangle \quad (4.42)$$

$$s^{c2}(v,v_j) = \langle \lambda^{(c2)}, h^{(c2)}(\omega,\omega_j) \rangle \quad (4.43)$$

$$s^{g1}(v) = \langle \lambda^{(\theta)}, h^{(\theta)}(\theta) \rangle \quad (4.44)$$

$$s^{g2}(v,v_j) = \langle \lambda^{(d\theta)}, h^{(d\theta)}(d\theta) \rangle + \langle \lambda^{(ds)}, h^{(ds)}(s_j - s_i) \rangle + \langle \lambda^{(dL)}, h^{(dL)}(dL) \rangle \quad (4.45)$$

$$(4.46)$$

The inference problem is to find the parse node v_0 rooted at the start symbol s_0 for grammar $\mathcal{G}$ and parameters λ with maximal score given an image I,

$$v_0^* = \arg\max_{v_0} s(v_0, I). \quad (4.47)$$

To recover the full parse graph of the optimal parse v_0, we can recursively backtrack through each local arg max to recover the child parse states for every parent.

4.3.2 Best-first parsing

This approach is motivated by heuristic search introduced by Pearl [Pea84], the best-first chart parsing of natural language from Charniak et al.[CGJ98], as well as the top-down/bottom-up image parsing framework of Wu and Zhu [WZ11]. The basic idea is to maintain a table, or chart, of the best scoring sub-solutions

Algorithm 3 Greedy Best-First Inference

procedure PARSEBFS($\mathbf{I}, \mathcal{G}, N$) ▷ Beam size N
    Run detection for each production
    Apply non-maximal suppression among all proposals, and populate beams for each symbol with top N detections
    repeat
        Select the highest scoring proposal ρ among all beams
        if ρ is missing parent **then**
            Run Bottom-up(ρ)
        end if
        if ρ is missing children **then**
            for each unexpanded child v of ρ **do**
                Run Top-down(v)
            end for
        end if
        if ρ is a complete parse graph **then**
            Remove ρ from its beam and place into candidate solution list
            Remove all parts overlapping with ρ from all beams
        end if
    until no further progress can be made
    return candidate solution list for root part of $\mathcal{G}$
end procedure

and systematically update these records in a best-first manner. Instead of tables, we use lists of the top scoring sub-solutions, one for each symbol. These proposal lists are called the beams, and have length N. The best-first order is determined by finding the locally optimal extension of a sub-solution by localizing its children in a top-down manner, or by localizing its parent and siblings in a bottom-up manner.

Each sub-solution in the list is a proposal, and represents a partial parse graph. Once a proposal is fully expanded, meaning it contains a full parse tree, it is placed in a list of candidate solutions, and all proposals in the beams that overlap with the solution are removed. Proposals in each list are ordered by descending score. Because each of the sub-solutions in a beam may be at multiple levels of expansion, their scores are not directly comparable due to the exclusion of terms from the missing parts. In combinatorial search such as A^*, this missing parts would be estimated by an admissible heuristic. For the image parsing case, no such heuristic is is readily computable, and instead a simple approximation is made by replacing each missing term with the best observed response from the training data. This generally causes the sub-solution scores to decrease as they are expanded further, promoting the algorithm to expand the most promising candidates until they are surpassed by alternate proposals in the beam.

The algorithm begins by running part detection for all parts in the model. For each production appearance template, a sliding-window detector is run over all relevant locations, orientations, and scales. The threshold used for detection is determined such that there were no false negatives in the training data, or in other words none of the true positives were missed. Non-maximal suppression is run on the result, and the proposals are sorted by descending score. The top N proposals are then placed into the beam for that part type. These are the best

parse proposals without considering any context.

For each iteration of the algorithm, the top scoring proposal is selected among all the beams. A top-down search is then performed on all unexpanded child nodes, and a bottom-up search is performed on the root node. These two operations are described as follows, for a selected proposal ρ:

Top-down(ρ): during training, we record a maximum bounding area that each child part can appear in, using the coordinate frame of the parent part. For each non-terminal leaf node in ρ, the form of the node is already known, and all combinations of child proposals to that node within these bounding areas are searched from their corresponding beams. The composition with the highest increase in score when added to ρ is kept and the score of ρ is updated. If no composition could be found because either beams were either empty or no parts were present in the bounding areas, then stochastic sampling is used to predict the locations of the children. Using the same Gibbs sampler used to train the geometry model, conditioned samples are drawn for the derivation and geometry of the children with the parent geometry fixed. After a fixed number of iterations, the best scoring sample is selected and added to ρ. Examples of conditioned sampling is shown in figure 4.14, except in this case only the immediate children of some leaf node of ρ is sampled.

Bottom-up(ρ): this operation involves identifying a parent and corresponding siblings for ρ of compatible productions and geometries. The search along the beams proceeds in exactly the same manner as the top-down search by identifying the maximal bounding area for each sibling and parent part and selecting the composition with maximal score increase to ρ. The one difference in this case is that ρ may appear in multiple locations within the parent production. For example, an arm may appear at the left or right side of the body. When this is

the case, compositions for ρ at all valid child positions of the parent production must also be searched. Again, if no composition can be found, the best composition found through conditioned sampling is used. Once a parent and siblings are found, the score for the new parent is computed and added to its respective beam, and ρ is removed from its beam.

The major drawback to this best-first beam search approach to parsing is that it is greedy by nature, and subject to getting stuck in local minima. This is true particularly when the parts models are very ambiguous, causing a large number of part candidates to be in the search beam. Secondly, the use of such large beams will significantly impact the computational time needed by the algorithm. To address some of these drawbacks, we explore an alternative approach using dynamic programming, which is described in the following section.

4.3.3 Dynamic programming

The basis of any dynamic programming algorithm is to solve a complex problem by breaking it down into a series of simple problems. The natural sub-problem in the grammar case is finding the optimal configuration of parts for each production. More specifically, given a production $(\alpha \rightarrow \beta, t, R)$, and a parse node for the α symbol, we must search of all possible productions to expand the β symbols, as well as search over the full joint space of their geometries:

$$s^*(v, I) = \max_{v_1, v_2, \ldots, v_{|\beta|}} g(v, I) + s^*(v_1, I) + s^*(v_2, I) + \cdots + s^*(v_{|\beta|}, I) \qquad (4.48)$$

where $g(v, I)$ are the local potentials within the production from Equation 4.41.

In the most general case of a phrase-structured grammar with general relations, finding the globally optimal parse involves searching over the product space of $|\beta|$ parse nodes for each production, which is immense. As an example, using

a 100×100 image, with 10 part orientations, 5 aspect ratios, 5 scales, $|\beta| = 3$ children in the production, and 5 productions for each child symbol, the total number of configurations to search over is $(100^2 \cdot 10 \cdot 5 \cdot 5 \cdot 5)^3 \approx 10^{21}$, for each production.

To reduce this search space to a tractable number, we first restrict the relations in the model to form a tree topology. This enforces conditional independence between the children, and reduces the search space from being exponential in the number of children to linear. These topologies can be either the star or tree structures in Figure 3.5(a,b) for the phrase-structure grammar case, or the dependency grammar case in Figure 3.6, which is always tree structured.

The most general of these cases is the phrase-structured articulated tree case, which has two types of hierarchies. The first is the natural phrase decomposition of the grammar, for which the production is the basic unit of computation of finding the optimal children for every parent state. Within each production, however, is a tree-structured sub-model, for which the children are connected in their own articulated hierarchy. The dynamic programming algorithm, therefore, has two nested operations – one for optimizing the parts within a production, and one for assembling the optimal productions into an optimal parse.

To distinguish between these two hierarchies, for some production ω and parse node v we define $R_\omega(v)$ to be the *relational* children of the part, which are the parts on the distal side of v that are connected to v by some relation. Figure 3.5(b), illustrates these relations. The set of relational children are not to be confused by the set of production children $C(v)$, which is the set of all parse nodes formed by the production, except for the production root.

Each production in the grammar is essentially a self-contained sub-model, in that it defines its own appearance model for the root part of the production, as

well as all the contextual relations between the root and children. All of these relations are therefore conditioned on the production. For example, assume we have multiple full-body productions that decompose into upper-body and lower-body. Furthermore, assume there are multiple productions that also expand the upper-body and lower-body parts into smaller constituents. The co-occurrence compatibility between the upper-body and lower-body production selections will therefore be dependent on which full-body production is used, because that relation is defined only for the full-body production. We therefore write the score function with a subscript $s_\omega(v, I)$ to indicate the production that is providing the relations from which the score is being computed on. This is important to disambiguate because parts can participate in multiple productions, i.e. a child part can be the root part of some other production.

The maximization over all parses can then be generally written

$$s^*_\omega(v, I) = \sum_{v_i \in R_\omega(v)} \max_{\omega_i, x_i, y_i, \theta_i, s_i} \left[h(v_i, I) + h(v, v_i, I) + s^*_{\omega_i}(v_i) + s^*_\omega(v_i) \right]. \quad (4.49)$$

where $h(v, I)$ are the unary score terms, and $h(v, v_i, I)$ are the pairwise contextual score terms. The term $s^*_{\omega_i}(v_i)$ is the optimal score for the root part of production ω_i at state $(x_i, y_i, \theta_i, s_i, \ell_i)$, whereas $s^*_\omega(v_i)$ is the optimal sub-tree score of parts within the same production ω. We can think of $s^*_{\omega_i}(v_i)$ as being the optimal compositional appearance score, incorporating the optimal appearance and geometries of its constituents. $s^*_\omega(v_i)$ on the other hand is the articulated sub-tree score, incorporating the optimal scores of all its distal parts into the score.

Like all dynamic programming algorithms, we maintain a score map for every part state containing the optimal sub-scores over each relation being optimized over. If we ensure that the maximization over one variable does not depend on the remaining variables, the maximization can be simplified by optimizing over each variable separately and updating the sub-score map. For example, let (x, y)

be our state space, $f(x, y)$ is the optimal sub-score map, and $g(\cdot)$ are the relations we wish to optimize over:

$$f(x_i, y_i) = \max_{x_j, y_j}(g(x_i, x_j) + g(y_i, y_j) + f(x_j, y_j)). \tag{4.50}$$

We can max over each variable separately, and sum the incremental results into the sub-score map:

$$f(x_i, y_i) = \max_{x_j}\left(g(x_i, x_j) + \max_{y_j}\left(g(y_i, y_j) + f(x_j, y_j, z_j)\right)\right) \tag{4.51}$$

$$= \max_{x_j}\left(g(x_i, x_j) + f_y(x_i, y_j)\right) \tag{4.52}$$

where f_y is the sub-score map with the optimal $g(y_i, y_j)$ added in.

After substituting in all the potential functions, the maximization equation is

$$s^*_\omega(v, I) = \sum_{v_i \in R_\omega(v)} \max_{\omega_i, x_i, y_i, \theta_i, s_i} [s^a(v, I) + \langle \lambda^{(c1)}, h^{(c1)}(\omega_i)\rangle + \langle \lambda^{(\theta)}, h^{(\theta)}(\theta_i)\rangle \tag{4.53}$$

$$+ \langle \lambda^{(c2)}, h^{(c2)}(\omega, \omega_i)\rangle + \langle \lambda^{(d\theta)}, h^{(d\theta)}(d\theta)\rangle \tag{4.54}$$

$$+ \langle \lambda^{(ds)}, h^{(ds)}(s_i - s)\rangle + \langle \lambda^{(dL)}, h^{(dL)}(dL)\rangle \tag{4.55}$$

$$+ s^*_{\omega_i}(v_i) + s^*_\omega(v_i)]. \tag{4.56}$$

Note that these optimal sub-scores do not include any of the unary terms for their root part. Therefore, to compute the globally optimal parse score, we maximize over the unary terms plus the optimal sub-scores for each production of the root grammar symbol. Let $\omega_1^{s_0}, \ldots, \omega_k^{s_0}$ be the set of all k productions that can rewrite the start symbol s_0. The globally optimal parse score can then be computed as

$$s^*(v, I) = \max_{\omega = \omega_1^{s_0} \ldots \omega_k^{s_0}} \left[s^*_{\omega_i} + \langle \lambda^{(c1)}, h^{(c1)}(\omega_i)\rangle + \langle \lambda^{(\theta)}, h^{(\theta)}(\theta_i)\rangle\right]. \tag{4.57}$$

The primary concern in computing the maximization over relations is the potential term for dL, which is the joint displacement between articulated parts.

Maximizing this term requires searching over every image location of the distal part, for every image location of the proximal part. The number of image locations can be quite large, and because this operation has $O(n^2)$ complexity, the computation time for large images can easily become intractable. If we can maximize over this relation more efficiently, then the remaining variables have small enough cardinalities to be maximized in quadratic time without considerable impact on actual computing time.

To more efficiently compute the maximization over joint displacement, we first make an approximation to the potential function. In its current form, this potential function can be viewed as a piecewise-constant function mapping. In order to exploit a more efficient computation, we first approximate this function mapping with a Gaussian. The displacement should be zero-mean, as the most common part configuration should articulate exactly around the joint, and has the following form:

$$\langle \lambda^{(dL)}, h^{(dL)}(dL) \rangle \approx (dL)^\top \Sigma^{-1} (dL) + b. \tag{4.58}$$

The displacement $dL := T_{prox}(v_j) - T_{dist}(v_i)$ is a 2-dimensional vector representing the difference of transformed joint coordinates computed from the proximal and distal part states using Equation 3.8. Because the prior distribution is unnormalized, there is an unknown normalization constant b contributed by the factor. This quadratic function $(dL)^\top \Sigma^{-1} (dL) + b$ is simply fit to best match the piecewise constant function mapping $\lambda^{(dL)}$ using least-squares. We further simplify this approximation by assuming that the covariance Σ is diagonal, in which case we can fit the distribution in the x and y dimensions independently. The parameters $\lambda^{(dL)}$ are now just two numbers, representing the inverse variance in

x and y, and denoted $\lambda^{(dx^2)}$ and $\lambda^{(dy^2)}$ respectively

$$\Sigma^{-1} = \begin{bmatrix} \lambda^{(dx^2)} & 0 \\ 0 & \lambda^{(dy^2)} \end{bmatrix}. \tag{4.59}$$

The maximization can now be written

$$s^*_\omega(v, I) = \sum_{v_i \in R_\omega(v)} \max_{\omega_i, x_i, y_i, \theta_i, s_i} [s^a(v, I) + \langle \lambda^{(c1)}, h^{(c1)}(\omega_i) \rangle + \langle \lambda^{(\theta)}, h^{(\theta)}(\theta_i) \rangle \tag{4.60}$$

$$+ \langle \lambda^{(c2)}, h^{(c2)}(\omega, \omega_i) \rangle + \langle \lambda^{(d\theta)}, h^{(d\theta)}(d\theta) \rangle \tag{4.61}$$

$$+ \langle \lambda^{(ds)}, h^{(ds)}(s_i - s) \rangle \tag{4.62}$$

$$+ (\lambda^{(dx^2)} \cdot dx^2) + (\lambda^{(dy^2)} \cdot dy^2) + s^*_{\omega_i}(v_i) + s^*_\omega(v_i)]. \tag{4.63}$$

The terms dx^2 and dy^2 are the squared x and y components of dL.

The generalized distance transform, introduced by Felzenszwalb and Huttenlocher [FH04], is an algorithm for maximizing an expression of the form

$$f(x_i, y_i) = \max_{x_j, y_j} \left[(x_j - x_i)^2 + (y_j - y_i)^2 + h(x_j, y_j) \right] \tag{4.64}$$

where $h(x_j, y_j)$ is an arbitrary function. The simplified Gaussian joint displacement terms in the score (Equation 4.63) has the same form, where the optimal sub-score maps $s^*_{\omega_i}(v_i) + s^*_\omega(v_i)$ are rolled into $h(x_j, y_j)$. These maps are over the full set of variables, and not just x and y, but this doesn't affect the optimization due to the decomposition of max operations in Equation 4.52. The principal advantage of using the generalized distance transform to compute this maximization is that it produces the same solution as dynamic programming, but runs in linear ($O(n)$) time as opposed to the quadratic complexity ($O(n^2)$) of dynamic programming. Computing exact solutions by maximizing over the relations can now be computed with enough efficiency to be practical.

Algorithm 4 Dynamic programming algorithm for finding maximal scoring parse. Let $\mathbf{x} = (x, y, \theta, s, \ell)$.

procedure PARSEDP(grammar $\mathcal{G}$, image I)
  $score \leftarrow -\infty$
  MAXOR(s_0)
  for each production ω that can rewrite the root symbol s_0 **do**
    $score \rightarrow \max(\max_{\mathbf{x}} M_\omega, score)$
  end for
  return *score* ▷ replace max with arg max to compute pg^*
end procedure

The final dynamic programming algorithm can be viewed as a chart parser similar to those used in natural language, as well as a message-passing architecture used in graphical models. The chart parsing aspect handles the optimization over production selection, which is the grammatical aspect of the model. The message-passing component handles the optimization over the relations, which is the geometric aspect of the model. Like any dynamic programming algorithm, score maps are recursively computed to contain optimal scores of all sub-configurations, using the optimal score maps of even smaller sub-configurations.

The algorithm maintains a score map for each production ω, denoted M_ω. These maps form the chart, and are used to find the optimal production for a given geometric state. Similarly, M_r is the score map for some relation r defined within a production. These maps are equivalent to a factor state map in a message passing architecture. Both maps are 5-dimensional to represent the entire state space (x, y, θ, s, ℓ) of their respective parse nodes. For the production maps, these maps store the optimal score for a sub-parse starting from that production for each possible geometry. For the relations maps, the scores represent the optimal

Algorithm 5 Dynamic programming algorithm for finding maximal scoring parse. Let $\mathbf{x} = (x, y, \theta, s, \ell)$.

procedure MAXAND(production ω)
  if M_ω already exists **then**
    return
  end if
  $M_\omega(\mathbf{x}) \leftarrow s^a(\mathbf{x}, I) \quad \forall \mathbf{x}$ ▷ Appearance score
  for each child symbol s **do**
    MAXOR(s)
  end for
  for each child relation r of the root part of ω **do**
    MAXRELATION(r)
    $M_\omega(\mathbf{x}) \leftarrow T_{dist}^{-1}(T_{prox}(M_r(\mathbf{x}))) \quad \forall \mathbf{x}$
  end for
  $M_\omega(\mathbf{x}) \leftarrow M_\omega(\mathbf{x}) + \langle \lambda^{(c1)}, h^{(c1)}(\omega_i) \rangle \quad \forall \mathbf{x}$
  $M_\omega(\mathbf{x}) \leftarrow M_\omega(\mathbf{x}) + \langle \lambda^{(\theta)}, h^{(\theta)}(\theta_i) \rangle \quad \forall \mathbf{x}$
end procedure
procedure MAXOR(symbol s)
  for each production ω that can rewrite s **do**
    MAXAND(ω)
  end for
end procedure

Algorithm 6 Dynamic programming algorithm for finding maximal scoring parses from AND node relations. Let $\mathbf{x} = (x, y, \theta, s, \ell)$.

procedure MAXRELATION(relation r)
  for each child relation r_i **do**
    MAXRELATION(r_i)
  end for
  $M_r(\mathbf{x}) \leftarrow -\infty \quad \forall \mathbf{x}$
  for each proximal production ω_p compatible with r **do**
    for each distal production ω_d compatible with r **do**
      $M_{temp}(\mathbf{x}) \leftarrow M_{\omega_d}(T_{prox}(\mathbf{x})) \quad \forall \mathbf{x}$
      $M_{temp}(\mathbf{x}) \leftarrow \max_{x',y'}[(\lambda^{(dx^2)} \cdot dx^2) + (\lambda^{(dy^2)} \cdot dy^2) + M_{temp}(x', y', \theta, s, \ell)] \quad \forall \mathbf{x}$ ▷ Distance transform
      $M_{temp}(\mathbf{x}) \leftarrow \max_{\theta'}[\langle \lambda^{(d\theta)}, h^{(d\theta)}(d\theta) \rangle + M_{temp}(x, y, \theta', s, \ell)] \quad \forall \mathbf{x}$
      $M_{temp}(\mathbf{x}) \leftarrow \max_{s'}[\langle \lambda^{(ds)}, h^{(ds)}(s' - s) \rangle + M_{temp}(x, y, \theta, s', \ell)] \quad \forall (x, y, \theta, s, \ell)$
      $M_{temp}(\mathbf{x}) \leftarrow M_{temp}(\mathbf{x}) + \langle \lambda^{(c2)}, h^{(c2)}(\omega_p, \omega_d) \rangle \quad \forall \mathbf{x}$
      $M_r(\mathbf{x}) \leftarrow \max(M_r(\mathbf{x}), M_{temp}(\mathbf{x})) \quad \forall \mathbf{x}$
    end for
  end for
end procedure

scores of the proximal part in the coordinate frame of the joint defined by the relation. The transforms T_{prox} converts from the distal part reference frame to the joint reference frame, whereas the inverse transform T_{dist}^{-1} converts from the joint reference frame to the proximal part reference frame. These transforms are defined in Equation 3.8. Pseudocode for the dynamic programming algorithm is shown in Algorithm 4, 5, and 6.

The results of the dynamic programming algorithm are visualized in Figure 4.5. The grammar used for this figure is visualized in Figure 4.7, which has 4 levels of depth. We run the dynamic programming algorithm four times, each time including an additional level of depth in the grammar. At the top level, only the root part is present and the score map consists purely of the full-body appearance score. In this case, black represents high score, and white is low score. This result is essentially equivalent to running a sliding window detector of a single object template. We visualize the scores by filling the entire boundary of the part, instead of just the center point in the score map. At the second level, two parts for the upper and lower body are included. All pose configurations and productions are searched, and the root score map is updated to reflect the best production and corresponding geometries found for each location. As more parts are introduced into the search process, the high level parts tend to become more strongly localized. Similarly, for parts with very weak appearance such as hands and lower arms, the algorithm is able to largely localize the true arm locations.

To retrieve the optimal parse graph pg^*, all max operators are replaced with an arg max. In practice, we maintain a second table for every optimal score map, which contains the backtrack to the state that produced that score. After all dynamic programming algorithm has finished, any state in optimal production score maps (e.g. charts) can be backtracked to recover the full parse or sub-parse.

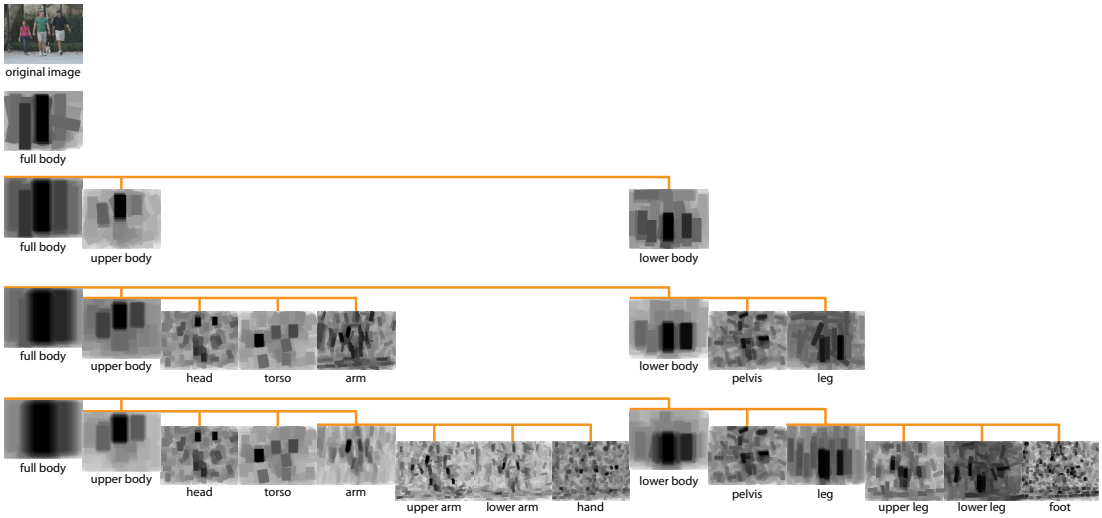

Figure 4.5: Optimal score maps from dynamic programming. The DP algorithm is run four times on the top image, each run including one more level of depth from the AND-OR graph. Dark regions represent high scoring part locations, and light regions represent low scores. As more context is added, score maps for the higher level parts begin to rule out much of the image where no valid compositions of parts could be found.

4.3.4 Tree reranking algorithm

This approach is motivated by the rerank parser of Collins [CK00] as well as Charniak and Johnson [CJ05], used for parsing natural language. The basic observation is that there are considerable contextual relationships that are crucial in detecting the correct parse of a sentence, but cannot be captured with simple generative grammars such as a context-free grammar. Because the grammar can parse very efficiently, however, it can easily generate a top-N list of parses ranked by their probability. If the true parse can be in this top-N list with high probability without N being too large, then a discriminative classifier can be trained to re-score each parse such that reranking the list moves the true parse to the top rank. Because all the proposals are already syntactically valid by virtue of the grammar, the discriminative classifier only needs to focus on long range contextual features.

For the image grammar case, we assume the case with general relations within each production, illustrated in Figure 3.5(c). For this model, exact inference is completely intractable and approximate inference can be arbitrarily bad on difficult images. Instead, the model is simplified by enforcing a tree topology in the relations to enable exact inference using the dynamic programming algorithm described in the previous section. The tree topology is enforced by simply cutting all the relation edges that are not consistent with the articulated tree, as in Figure 3.5(b).

After running the dynamic programming algorithm on the image, the optimal parse can be generated by recursively backtracking from the state with maximal score. To compute the top-N list of parses, a local neighborhood around the maximal state is suppressed by setting the scores in the map to $-\infty$. The next-best maximal parse is then recovered using the same process until N parses are

Algorithm 7 Dynamic programming algorithm with reranking. Let $\mathbf{x} = (x, y, \theta, s, \ell)$.

procedure MAXANDRERANK(production ω)
  if M_ω already exists **then**
    return
  end if
  MAXAND(ω)
  RERANKMAP(ω)
end procedure
procedure RERANKMAP(production ω)
  for each state $\mathbf{x}$ in score map M_ω **do**
    $v_{\omega,\mathbf{x}} \leftarrow$ BACKTRACK($M_\omega, \mathbf{x}$) ▷ Root node of the backtracked sub-parse graph
    $M_\omega(\mathbf{x}) \leftarrow s^{(full)}(v_{\omega,\mathbf{x}}, I)$ ▷ Rescore using full model
  end for
end procedure

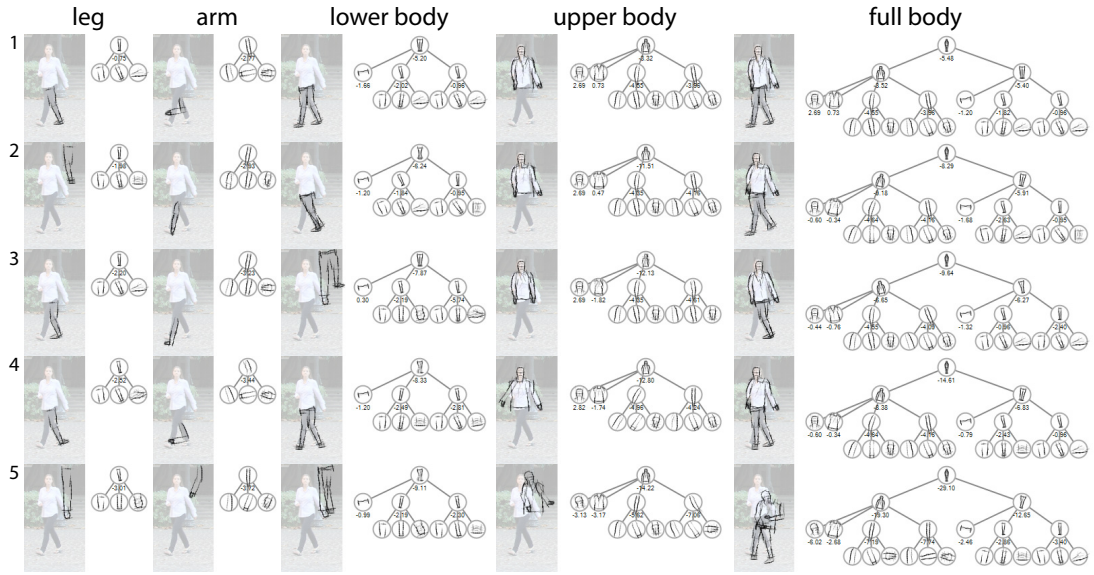

Figure 4.6: The top-5 parses are extracted from the dynamic programming algorithm for arms, legs, upper and lower body, and full body. For low-level parts such as arms and legs, the top proposals are often incorrect. Many of these ambiguities become resolved by the contextual relations at higher levels of the grammar.

generated, or all root states have been suppressed. Examples of the top-5 parses and sub-parses extracted in this manner are shown in Figure 4.6.

Instead of utilizing a secondary discriminative classifier to rerank the top-N parses, we simply replace the relation edges that were cut and recompute the score using the full model. Furthermore, sub-solutions can also be reranked in this manner, and this reranking strategy can be applied recursively to each sub-computation in the dynamic programming algorithm as opposed to only at the final result.

The reranked variant of the DP algorithm is summarized in Algorithm 7. Each time an optimal score map for a production (AND-node) is computed, the

entire map is rescored using the full model before returning. Every state in the score map must be rescored in this manner, because the full map is consumed by the previous recursion level. For this reason, we simply call the algorithm tree reranking instead of N-best reranking.

This algorithm is another form of approximate inference, and relies on the true solution somewhere in the DP maps where it can hopefully be picked up by the reranking. If the true solution does not appear anywhere in the DP maps, then it is impossible for this algorithm to recover it. For this reason, the tree reranking strategy is more suited to rejecting false positives than identifying true positives.

4.4 Parsing experiments

4.4.1 Grammar design

In the following experiments, we represent the human body as a coarse-to-fine composition of 22 parts, which include parts for the hands, feet, and face. The grammar we design is a phrase-structure grammar that composes the parts using 14 OR-node symbols due to part sharing, and a total of 104 AND-node productions. Figure 4.7 illustrates the AND-OR grammar structure, and visualizes a subset of AND-node productions with icons portraying the part appearances they represent.

4.4.2 Justification of OR-nodes for appearance

One of the main motivations for using grammatical models is to treat distinct appearance variabilities between part instances as a selection from a set of sub-models, in the same spirit as modeling a complex multimodal distribution as

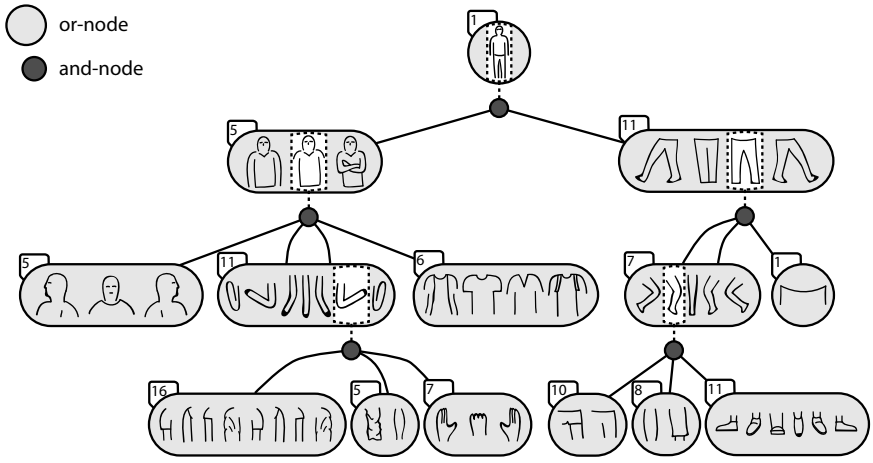

Figure 4.7: AND-OR grammar design used for experiments. The model represents a body with 22 parts, composed from 14 OR-node symbols and 104 AND-node productions. The set of icons inside each OR-node symbol represents some of the varieties in appearance the model represents. The numbers on the top-left of each OR-node represents the total number of AND-node productions that can rewrite that symbol.

a mixture model. For example, an arm segment has a significantly different appearance depending on if it is bare, or clothed in a t-shirt or sweatshirt. Each of these distinct part appearances become AND-node productions in the grammar, and each get their own appearance representation as well as geometric relations to their children.

To justify using the OR-node to break parts into a mixture of prototypical appearances, we conduct an experiment to evaluate the classification performance of the mixture model compared to the performance of each individual appearance model. Only primitive parts that have no further decomposition are used. For this experiment, we designate the probability of each appearance model as p_{and}, and the probability of the mixture combining all appearances for the same part as p_{or}. The trained appearance template for each AND-node production is designated as t. Using only the appearance terms and context-free grammar terms described in the previous sections, these models can be written:

$$p^{(and)}(I_\Lambda|v) = q(I_\Lambda) \exp\left\{ \sum_{(\Lambda_i, r_i, z_i)\in t} \lambda_i r_i(I_{\Lambda_i}) - \log z_i \right\} \tag{4.65}$$

$$p^{(or)}(I_\Lambda|v) = \sum_{i=1}^{k} \exp\left\{ \langle \lambda^{(c1)}, h^{(c1)}(\omega_i) \rangle \right\} p_i^{(and)}(I_\Lambda|v). \tag{4.66}$$

For this experiment we use the UCLA pedestrian dataset shown in Figure 2.1, which is split into 250 training examples and 150 testing examples. This experiment is evaluating detection and not parsing, so the positive examples are aligned and rescaled crops from the ground truth parts, and negative examples are randomly cropped from a large set of natural background images that contain no people. The appearance templates in this experiment are trained from the positive examples using the Active Basis model with re-weighted parameters from logistic regression.

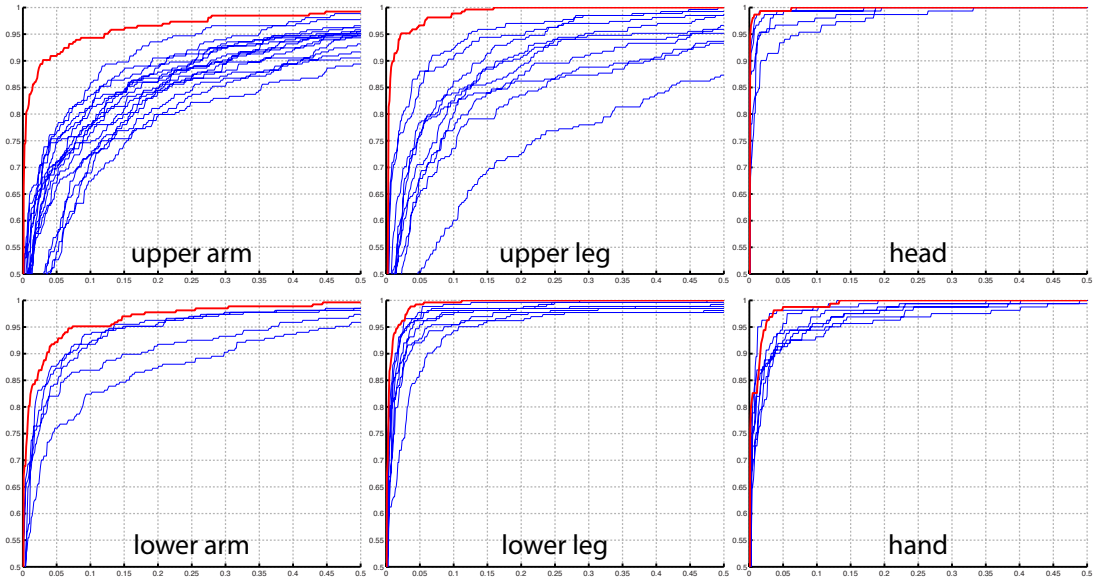

Figure 4.8: OR-node classification performance: the top-left quadrant of the ROC curves for several OR-node models. Blue lines indicate the performance of each AND-node production evaluated against all parts of the same OR-node symbol. The red line is the performance of the OR-node mixture model that combines all the AND-node models together.

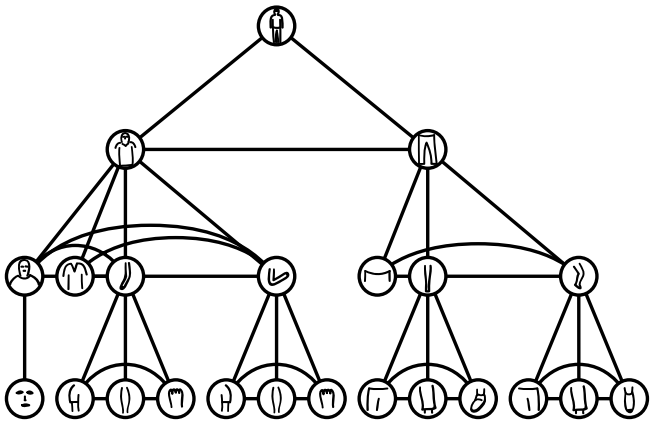

Figure 4.9: Parse graph derived from the AND-OR graph grammar in Figure 4.7.

The positive and negative test examples are classified by each AND-node model $p^{(and)}$ in isolation, as well as the mixture model $p^{(or)}$ ROC curves for a selection of parts are shown in Figure 4.8. The blue curves are the AND-node models, and the red curve is the combined OR-node mixture model. On many classes such as the legs and arms, there is a considerable amount of separation between the AND-node models, suggesting that each of these specific appearance models do not generalize particularly well to the remainder of the dataset. The OR-node mixture model, however, typically dominates all the AND-node models. Therefore, by structuring the model to have interchangeable appearance models, we observe a performance improvement in essentially all cases.

4.4.3 Parsing performance using Active Basis appearances

In this experiment we assume a fully-connected relational structure for each production, as illustrated in Figure 3.5(c). The grammar is also context-free in this case, which omits the f^{c2} potential terms from the score function. Parse graphs derived from this grammar are illustrated in Figure 4.9. Each parse graph will have the same topology of parse nodes and relations by design, even though they

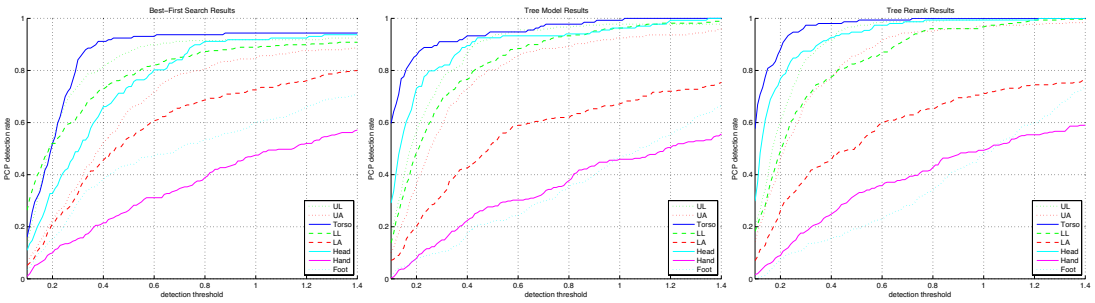

Figure 4.10: Comparison of part localization performance on the UCLA pedestrian dataset using best-first parsing, DP-Tree, and DP-Rerank parsers.

can be instantiated through different compositions of productions. This is because we ultimately evaluate on the localization performance of each primitive part, which is assumed to be present in every parse.

The UCLA pedestrian dataset is divided into 250 training examples, and 150 testing examples. The appearance templates for the AND-node productions are again trained using Active Basis with re-weighted parameters from logistic regression. These templates therefore represent the edge structure of the parts, but do not use the color or background features of the HIT model. The prior model is trained using the stochastic gradients described in Section 4.1, and the articulation potentials are re-fit to Gaussian distributions from Equation 4.58.

All three inference algorithms describe above are evaluated. The first is the best-first parser, using the full model with cyclical relations. Next, we evaluate the dynamic programming algorithm (DP-Tree) by simplifying the model to a tree topology by cutting all the non-articulated edges. Last is the dynamic programming algorithm with reranking (DP-Rerank) to correct for some of the errors caused by the tree approximation. Part localization performance from each of these algorithms is measured using the PCP criterion described in Section 2.3, and plotted in Figure 4.10.

Figure 4.11: Selected parsing results on the UCLA pedestrian dataset using the DP-Rerank parser. The bottom row illustrates some typical failures, which include matching parts to background, double-counting of arm or leg regions, or matching parts on the wrong edge.

Dataset	Method	torso	head	u.leg	l.leg	foot	u.arm	l.arm	hand	avg
UCLA	Best-first	92.1	73.4	85.6	78.1	43.5	64.4	53.6	25.2	61.9
	DP-Tree	94.6	80.2	93.2	82.7	21.6	79.9	50.4	27.3	62.3
	DP-Rerank	98.9	85.6	94.6	82.0	18.7	85.3	51.8	31.7	65.2

Table 4.1: Comparison of PCP results at threshold=0.5 on the UCLA pedestrian dataset using context-free grammar and Active Basis model.

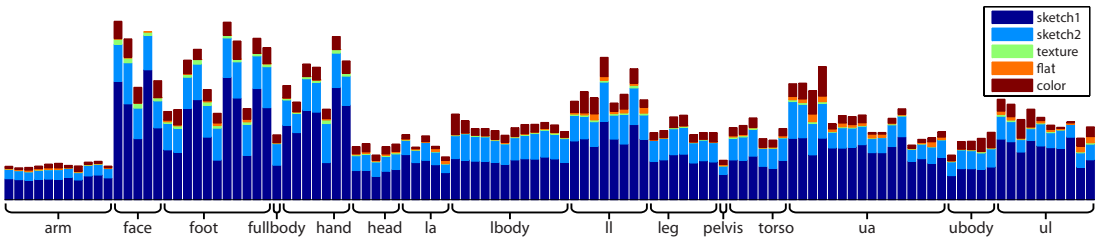

Figure 4.12: Breakdown of information gain for each prototype and part. Each bar represents the cumulative information gain for each template normalized by template size. The colors represent the relative contribution of each prototype. Two sketch prototypes are used for all templates, sketch1 is fine scale while sketch2 is coarse scale.

Selected examples of the top scoring parse from the DP-Rerank algorithm are shown in Figure 4.11. Typical failure modes often include matching limbs to strong edges in the background, or the aligning the wrong edge of a part. Another common failure is the so-called double-counting problem, where the parser matches both legs or arms to the same image region.

4.4.4 Parsing performance using HIT appearances

This experiment utilizes the same grammar structure in Figure 4.7, but uses the HIT model to represent the appearance of all the AND-node productions.

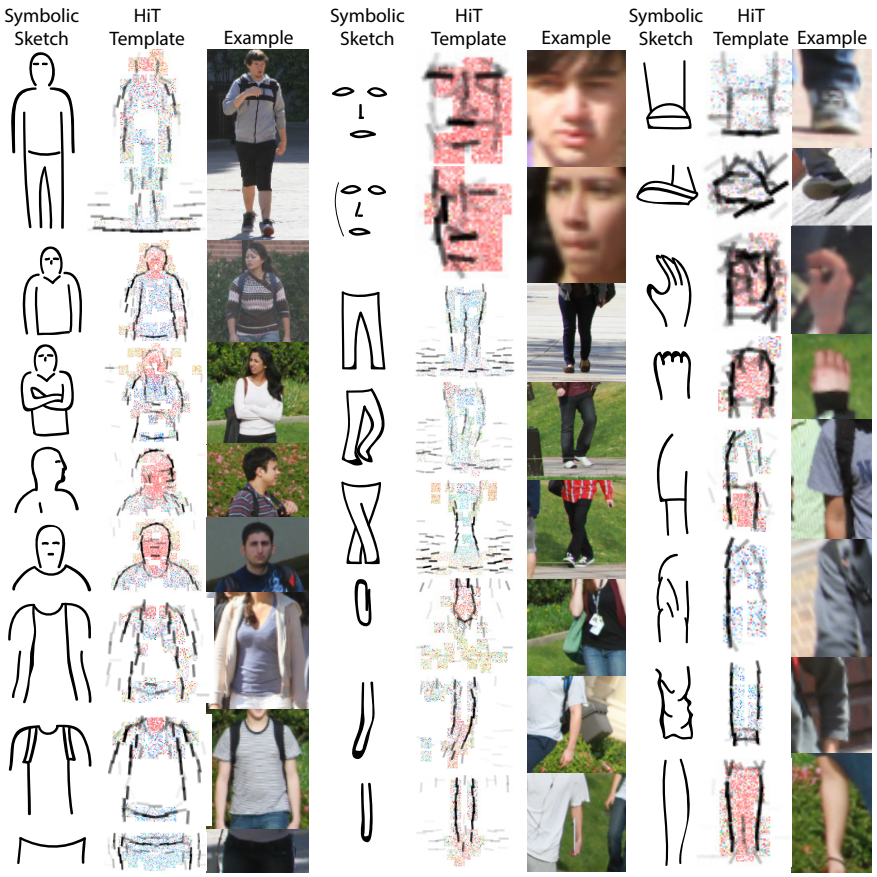

Figure 4.13: Learned HIT appearance templates for 25 of the 110 productions in the model. Each production corresponds to a prototypical part configuration which captures a particular perceptual aspect of the part's appearance such as clothing, color, geometry, pose, and lighting. The color features are rendered by sampling pixels from their corresponding prototype histograms.

Each HIT template consists of prototypes for sketch, texture, flat, and color. The prototype dictionary includes two types of sketch features at low and high resolution, as well as two variants of the sketch features with a low and high amount of deformation activity. The sketch prototype with high deformation activity serves as a background feature, described in Section 4.2.2. Unlike the grammar used in the previous experiment, this model uses the full set of potential terms, including the context-sensitive potentials.

The relational structure within the AND-node productions follows the kinematic tree, as visualized in Figure 3.5(b). Because the dynamic programming algorithm can compute exact inference on any model with tree-structured relations, the best-first and DP-Rerank algorithms provide no benefit in this case. Therefore, only the DP-Tree algorithm is used for this experiment.

We again use the UCLA pedestrian dataset, with the same 250 training and 150 testing examples used in the previous experiment. Examples of the learned HIT templates are shown in Figure 4.13. Furthermore, the information gain breakdown for each of the prototypes are shown for each AND-node production in Figure 4.12.

One of the advantages of generative models is that we can synthesize random samples to validate whether they indeed match our expectation of what the model represents. Such samples are shown in Figure 4.14. The same sampler used to train the model in Section 4.1 is used to synthesize the geometry. To render the appearance, a bar at the appropriate position and orientation is drawn for each sketch element, with the darkness proportional to the corresponding λ weight. The colors are sampled from their corresponding HIT color prototype histograms. We can observe the influence of the context-sensitive potentials in these samples, as the clothing types between adjacent parts are largely consistent, e.g. an upper

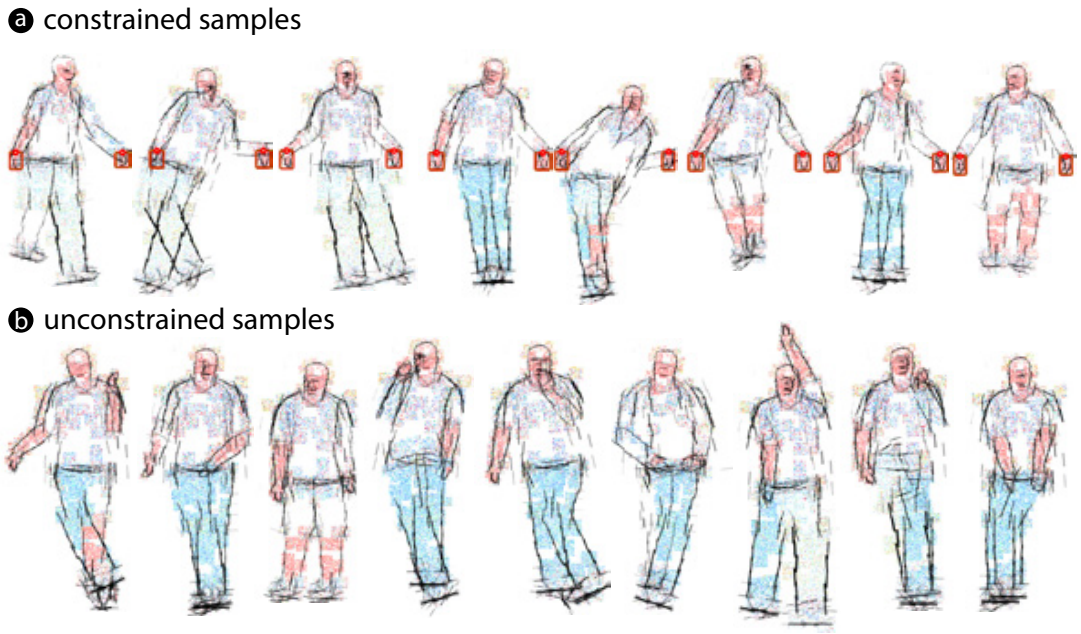

Figure 4.14: Random samples from the prior model are synthesized by sampling a random parse graph from prior model, then sampling the appearance template for each part. To illustrate the flexibility of the model, samples can be conditioned on any set of parts. Constrained samples conditioned on fixed hand positions are shown in (a), and unconditioned samples are shown in (b).

leg with shorts is never adjacent to a lower leg with pants. We can also condition on any variables we like during the sampling, which is shown in the top row where the hands have been locked in place.

Part localization performance results are shown in Table 4.2 and Figure 4.15. There is a significant improvement on essentially every part over the context-free case with Active Basis appearance models. Furthermore, we train the current published state-of-art technique of Yang and Ramanan [YR11] on the same data and demonstrate a 1.5% gain in average part localization performance.

We also evaluate the production prediction accuracy using a smaller gram-

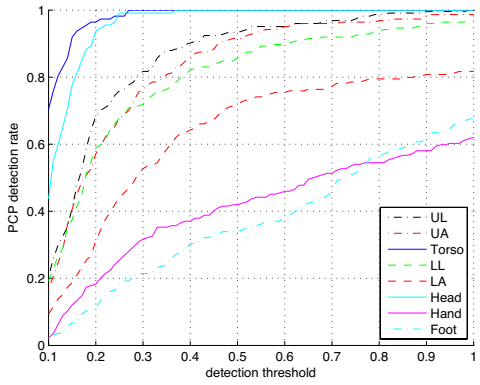

Figure 4.15: Part localization performance using context-sensitive phrase-structured grammar with HIT appearance models on the UCLA pedestrian dataset.

Dataset	Method	torso	head	u.leg	l.leg	foot	u.arm	l.arm	hand	avg
UCLA	[YR11] (2011)	100.0	100.0	97.5	83.9	-	95.1	57.7	-	86.9
	DP-Tree	100.0	100.0	93.3	85.7	33.9	91.5	71.9	42.0	88.4

Table 4.2: Comparison of PCP results at threshold=0.5 on the UCLA pedestrian dataset using context-sensitive grammar with HIT appearance models.

	torso	head	u.body
PARSE	79.5	63.0	69.3

Table 4.3: Production prediction accuracy on the PARSE dataset.

	torso	head	u.arm	l.arm	u.leg	l.leg	full-body
UCLA	36.5	67.7	55.6	68.5	51.0	75.0	70.0

Table 4.4: Production prediction accuracy on the UCLA pedestrian dataset.

Figure 4.16: Selected parse results from the DP-Tree algorithm using a context-sensitive grammar and HIT appearance models. Localization of the arms and legs are much improved from the context-free Active Basis experiment. Failure modes are still similar, with parts matched to background structures and double-counting being the most common.

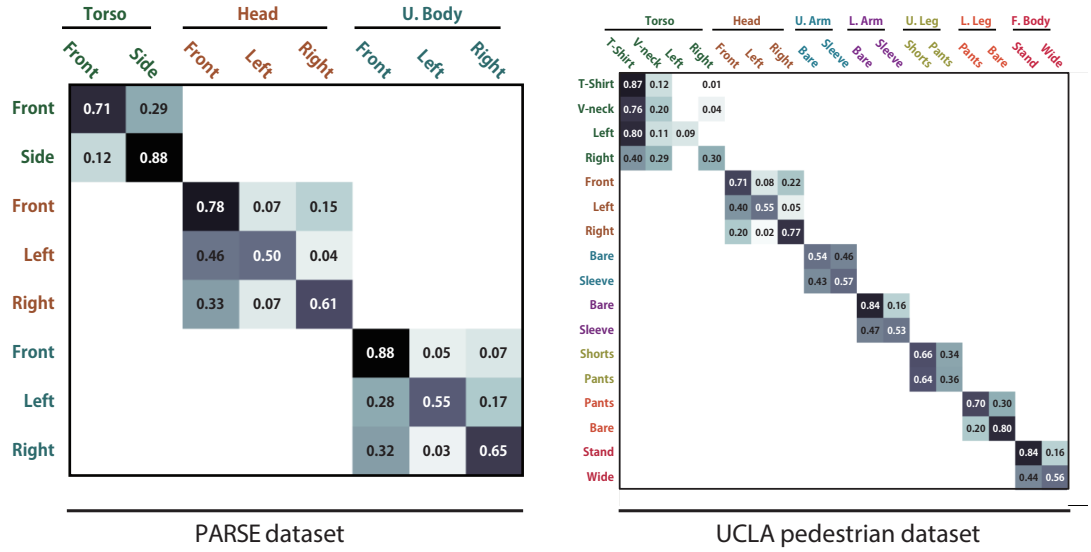

Figure 4.17: Confusion matrix of production prediction accuracy on the PARSE and UCLA pedestrian datasets.

mar containing a subset of productions from Figure 4.7. This evaluation is run on both the PARSE [Ram06] and UCLA pedestrian datasets, using grammar productions designed to be appropriate for each dataset. The confusion matrix for the production prediction accuracy is shown in Figure 4.17, along with the list of productions used. The evaluation protocol utilizes the same selection criteria from PCP for determining which parse to select for evaluation, for which we compare the predicted production labels (columns) against the ground truth annotated labels (rows). Accuracy is computed by summing the diagonal of the confusion matrix for each part, and dividing by the number of production choices for that part, and is shown in Table 4.4.4 and 4.4 for the PARSE and UCLA pedestrian datasets respectively. To our knowledge, no other work has been published to predict these types of attributes on PARSE for which to compare against.

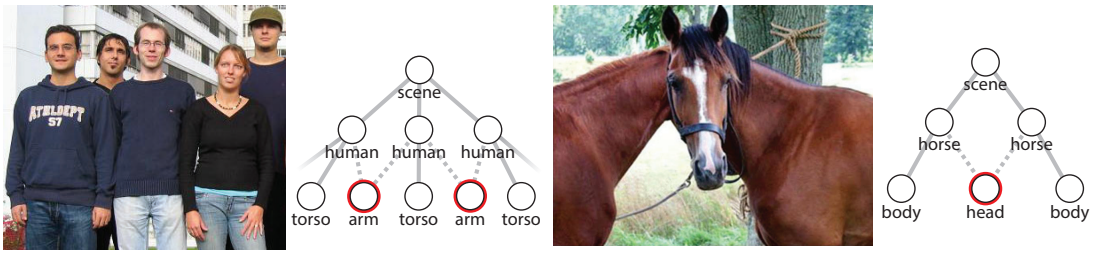

Figure 4.18: Ambiguous images: in these scenes, occlusion causes an ambiguity on parts, shown in red on the parse trees. In these cases, there is often no single correct solution, but several competing alternative solutions.

4.5 Ambiguity reasoning

Crowd scenes contain dense arrangements of people. In addition to normal environmental confounders, the inter-occlusions between people cause full-body or part based human detectors to frequently fail. This is due to the prevalence of genuinely correct parts from different people that coincidentally form a favorable geometry. To model this phenomenon, we introduce a latent structure and define a network of constraints to reason about occlusion and ambiguity between parts. We therefore seek to find parse solutions that also obey global consistency in terms of these constraints.

These constraints are hierarchical in terms of part ambiguity at each level of the AND-OR graph. A part is ambiguous if there is support in the image for multiple adjoining parts to form a set of competitive hypotheses. These ambiguities cannot be resolved locally, as a globally optimal solution may have many sub-optimal local configurations. Nor do we wish to find a single globally optimal solution, as there are potentially many different satisfying solutions to these constraints that correspond to multiple valid interpretations of the image.

Instead, we describe a method for efficiently exploring the solution space of this problem to recover multiple and distinct solutions using statistical sampling.

4.5.1 Representation of ambiguity

When using a part-based model in a scene with multiple subjects, there is an ambiguity for each part pertaining to which subject it is associated with. In the single-body case we assume a single correct interpretation of the image. In the multi-body case, however, there may be multiple reasonable interpretations of the image due to some unresolvable ambiguities. This shifts the task away from object detection and towards scene interpretation, which may have multiple valid solutions depending on what aspect of the image we are interested in.

The posterior distribution for complex scenes such as this are often multi-modal, where each mode corresponds to a reasonable interpretation of the image. Instead of searching for maximal posterior parse, we instead wish to find as many high probability modes as possible. Enumerating modes exhaustively is generally not tractable, nor is it straightforward to find secondary modes from the majority of inference algorithms used on these types of images. A notable exception is the k-adventurers algorithm proposed by [TCY05].

Our approach addresses this problem from the prospective of cluster sampling and simulated annealing. Part ambiguities are represented in a graphical model, designed such that we can sample from the model efficiently. We first assume that there is a set of candidate part proposals for each part in the AND-OR graph. For each proposal, we construct a node for each possible interpretation of the part. These interpretations correspond to the OR-nodes defined in the AND-OR graph. For each node, an integer-valued latent variable is defined. A value of zero indicates that this particular interpretation of the part is *off*, meaning that

it does not belong to any solution. Non-zero values indicate that the proposal interpretation participates in *some* solution.

Ambiguity constraints are defined as pairwise relations on these latent variables. There are two types of pairwise relations used: compatible and competitive. Compatible parts are in a favorable geometric configuration, and should participate in the same solution. An upper and lower arm proposal with good joint alignment will likely belong to the same person, and should have a compatible constraint between them. Competitive parts can never belong in the same solution. A part proposal can't have multiple interpretations simultaneously, and each pair of interpretations for the same proposal will have competitive constraints. Similarly, occluding parts may each participate in a distinct solution, but never the same solution, and a competitive edge is placed between them.

4.5.2 Candidacy graph model

The set of all proposal interpretations and their ambiguity constraints are arranged into a graph structure we call a candidacy graph, denoted as $G = (V, E)$. Each node $v \in V$ represents a proposal interpretation, and can take a value of $x \in \{0, 1, ..., k\}$. Each constraint between interpretations is represented by an edge $e \in E$. The state of the graph is denoted $X = \{x_1, x_2, ..., x_{|V|}\}$, and the probability of a given state is defined as

$$p(X) \propto \exp\left\{ -\sum_{e_{ij} \in E} \beta_{ij} \cdot c_{ij}(x_i, x_j) \right\}. \tag{4.67}$$

The constraint function $c_{ij}(x_i, x_j)$ evaluates to 1 when the constraint between x_i and x_j is satisfied, and 0 otherwise. The model parameters are β_{ij} for each edge e_{ij}. This is a generalization of the Potts model from statistical physics,

which uses only the equality constraint $c(x_i, x_j) = \delta(x_i, x_j) = \mathbf{1}(x_i = x_j)$. Our constraints are denoted as c_+, c_-, for compatible and competitive constraints respectively. These constraints are defined as follows:

$$c_+(x_i, x_j) = \begin{cases} 1 : \text{if } x_i = x_j \\ 0 : \text{otherwise} \end{cases}$$

$$c_-(x_i, x_j) = \begin{cases} 1 : \text{if } (x_i \neq x_j) \vee ((x_i = 0) \wedge (x_j = 0)) \\ 0 : \text{otherwise} \end{cases}$$

Any part assigned a value of 0 is considered *off* and is not included in any solution. Therefore, both compatible and competitive constraints are considered satisfied when both parts are *off*.

4.5.3 Composite cluster sampling

Simulated annealing is a common strategy for finding the minimum energy configuration of a probability distribution. The central idea is to draw samples from the distribution under a schedule of decreasing temperature. Eventually the distribution will form a delta function around its global maxima, and all samples will be drawn at this point. The probability distribution with a temperature term T takes the following form:

$$p(X) \propto \exp\left\{ -\frac{1}{T} \sum_{e_{ij} \in E} \beta_{ij} \cdot c_{ij}(x_i, x_j) \right\}. \tag{4.68}$$

One of the key problems with this approach on Markov random fields, however, is that the available sampling techniques are often single-site samplers such

as the Gibbs sampler. These samplers are very slow to converge, particularly when the connectivity of constraints is high. Furthermore, because each step of the sampler changes the state only at a single node in the graph, the likelihood of the sampler escaping a local energy minima is often diminishingly small, particularly when the temperature is low.

Our approach is motivated from the cluster sampling technique of Swendsen-Wang cuts [BZ05] and an extension called the C^4 algorithm [PZ10]. The motivation behind this method is to allow the sampler to make large jumps in the configuration space by updating multiple sites in the graph simultaneously, while still drawing fair samples. This has two particularly desirable consequences of faster convergence, and rapid exploration of modes in the distribution. We briefly discuss some background on these techniques to elucidate the development of our model.

The original Swendsen-Wang algorithm draws samples from the Ising model, which has the form

$$P_{ising}(X) \propto \exp\left\{-\beta \sum_{e_{ij} \in E} \mathbf{1}(x_i = x_j)\right\}. \tag{4.69}$$

where the values $x \in X$ are limited to binary states $\{0, 1\}$. Auxiliary variable $u_{ij} \in U$ are introduced on the edges $e_{ij} \in E$ that can take values $u_{ij} \in \{on, off\}$. By designing a joint distribution $p(X, U)$ such that the the original distribution is obtained when marginalizing out U, samples can be drawn from $p(X)$ by alternatingly drawing samples from the conditionals $p(U|X)$ and $p(X|U)$. The advantage of this scheme is that sampling from these conditionals is extremely efficient. $p(U|X)$ is sampled by first turning *off* edges u_{ij} for all edges where $x_i \neq x_j$ and with probability $1-q_0$ otherwise, where $q_0 = 1-e^{-\beta}$. $p(X|U)$ is sampled by grouping the remaining nodes into connected components $CP = \{V_i : i = 1, ..., n\}$. A

connected component V_0 is selected uniformly (with probability $1/|CP|$), and the nodes within V_0 are assigned their complement value $x = x^c, \forall x \in V_0$. Because of the initial step of turning edges *off* between nodes of different value, all nodes within a connected component must always have the same value. The Swendsen-Wang algorithm can also draw samples from the Potts model, where $x \in X$ can take states $\{1, 2, ..., k\}$. The only modification from the Ising case is to assign a state chosen uniformly from $\{1, 2, ..., k\}$ to the nodes in V_0 instead of flipping their states.

The Swendsen-Wang cuts algorithm (SWC) [BZ05] generalizes the Swendsen-Wang algorithm to arbitrary posterior distributions. By treating the Potts model as a proposal probability to a more general posterior model, an MCMC chain can be designed to draw samples from the posterior. First, a slightly more flexible version of the Potts model is considered, which includes a parameter β_{ij} for each edge instead of a single parameter for the whole model:

$$P_{swc}(X) \propto \exp\left\{ -\sum_{e_{ij} \in E} \beta_{ij} \cdot \mathbf{1}(x_i = x_j) \right\}. \tag{4.70}$$

Many problems in vision can be expressed in this form, and the degree of compatibility between specific nodes can be leaned from data. Sampling $p(U|X)$ in the SWC algorithm proceeds in the same manner as the original Swendsen-Wang algorithm, except the edge probabilities q_{ij} are used instead of the generic q_0. For the $p(X|U)$ step, connected components are formed as before and V_0 is selected uniformly among them. The nodes in V_0 are then relabeled (or recolored) with a new value $x = \ell, \forall x \in V_0$. The key difference with the SWC algorithm is that this relabeling must be accepting according to a Metropolis-Hastings rejection step $\alpha(A \to B)$, where A and B are the previous and proposed states of the full graph respectively. This acceptance probability is

$$\alpha(A \to B) = \min\left(1, \frac{\Pi_{e \in \mathcal{C}(V_0, V_{\ell'} \setminus V_0)}(1-q_e)}{\Pi_{e \in \mathcal{C}(V_0, V_{\ell} \setminus V_0)}(1-q_e)} \cdot \frac{q(\ell|V_0, B)}{q(\ell'|V_0, A)} \cdot \frac{p(B|\mathbf{I})}{p(A|\mathbf{I})}\right). \quad (4.71)$$

$\mathcal{C}(V_1, V_2)$ refers to the *cut* between components V_1 and V_2 which consists of the edges that separate V_1 and V_2, or more specifically $\{e_{ij} : x_i \in V_1, x_j \in V_2\}$. $q(\ell|V_0, B)$ is the proposal probability of relabeling V_0 to ℓ, which is often uniform. Finally, $p(B|\mathbf{I})$ follows from equation 4.70.

Finally we return to our own case of the candidacy graph, defined by equation 4.73. The C^4 algorithm [PZ10] is used in this case, which follows the same procedure as SWC by drawing repeated samples from $p(U|X)$ and $p(X|U)$. For drawing samples from $p(U|X)$, we turn *off* all edges e_{ij} for which $c_{ij}(x_i, x_j) = 0$. These are the the edges that violate their constraints. The remaining edges are turned *off* with probability $1 - q_{ij}$. To sample $p(X|U)$, a set of composite connected components $CCP = \{V_i : i = 1, ..., n\}$ are formed. These are called composite because they can be connected by a variety of different constraint types now, as opposed to only compatible constraints in the previous cases. A component V_0 is selected from CCP, however, some care must be taken in relabeling the nodes. If there are non-compatible constraints present in V_0, assigning the same label to all the nodes will cause a constraint violation. To account for this, we further partition V_0 into additional connected components $CP = \{\mathcal{V}_i : i = 1, ..., m\}$ such that each component is connected only by compatible constraints.

Each component in CP must be assigned the same label, however, the labels of different components must be selected such that all constraints within V_0 are satisfied. Denote $\mathcal{L}$ as the set of all possible satisfying assignments of V_0. We select an assignment $l \in \mathcal{L}$ to relabel V_0 and accept the proposal with probability

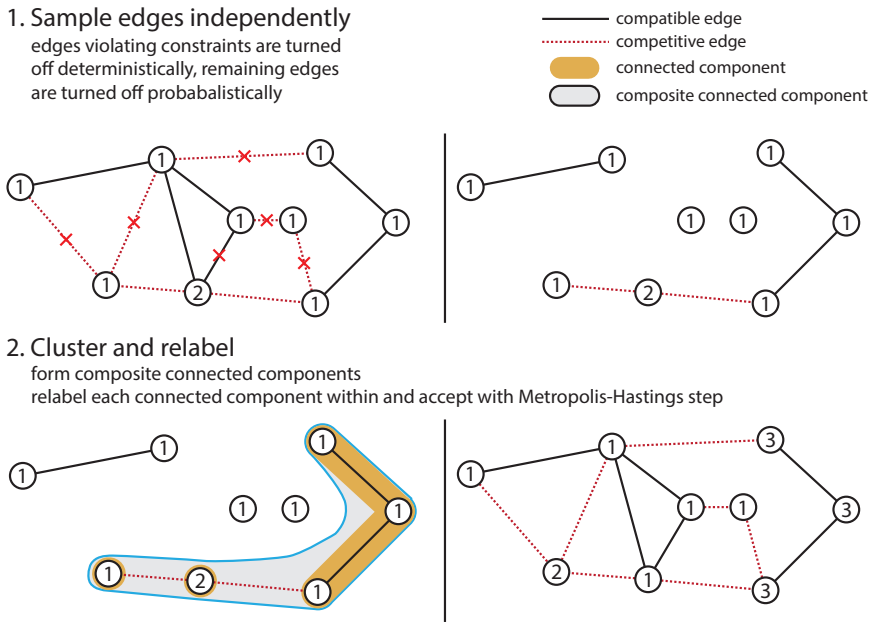

Figure 4.19: $\mathbf{C^4}$ algorithm on a Potts model: similar to other cluster sampling algorithms, C^4 is able to make large configuration changes by changing the state of entire clusters simultaneously. C^4 extends the conventional Swendsen-Wang and Swendsen-Wang Cuts algorithms to allow additional pairwise constraint types, shown here as competitive constraints that are only satisfied when nodes have different labels.

$$\alpha(A \to B) = \min\left(1, \frac{\Pi_{e \in \mathcal{C}(V_0, V_{l'} \setminus V_0)}(1 - q_e)}{\Pi_{e \in \mathcal{C}(V_0, V_l \setminus V_0)}(1 - q_e)} \cdot \frac{q(l|V_0, B)}{q(l'|V_0, A)} \cdot \frac{p(B|\mathbf{I})}{p(A|\mathbf{I})}\right). \tag{4.72}$$

This is exactly the same form as in the SWC case, except the cut consists of multiple edge constraint types between two different labeling schemes l and l'. The sampling process using the C^4 algorithm is illustrated in figure 4.19.

Algorithm 8 Sampling the candidacy graph

procedure SAMPLECANDIDACYGRAPH(proposals $\Pi = \{\pi_i\}$)
  for each new part proposal π_i **do**
    Create nodes $\{\pi_i^1, ..., \pi_i^n\}$ for all n interpretations of proposal π_i
    Add competitive constraints between each of the interpretations of π_i with $q_e = 1$
    Add competitive constraints with any overlapping node already in the candidacy graph with q_e proportional to the amount of overlap
    Add compatible constraints with all compatible nodes already in the candidacy graph with q_e proportional to their relation score
    Insert nodes $\{\pi_i^1, ..., \pi_i^n\}$ into the candidacy graph
  end for
  repeat
    Turn all edges *on*
    Turn *off* all edges where $s(c_i) = 0$
    Turn *off* all remaining edges with probability $1 - q_e$
    Form composite connected components (CCP) from edges still *on*, uniformly select one CCP V_0
    Further divide V_0 into smaller connected components (CPs) using only compatible edges
    Relabel the CPs in V_0 uniformly
    Accept the new labeling of V_0 with probability $\alpha(A \to B)$
  until the energy of the system stabilizes
end procedure

4.5.4 Hierarchical candidacy graph

The above description of the candidacy graph only models local ambiguity between part interpretations, and does not take into consideration the hierarchical part assemblies and their probabilities from the grammar. We begin by considering a subset of the AND-OR graph which consists of 3-layers: an OR-node at the root, the children AND-nodes, and their immediate OR-node children. We enumerate all the unique leaf-node parts of this sub-graph, and record the AND-nodes from which they were derived. Any part with multiple derivations is ambiguous, and we construct candidacy nodes for each of these interpretations. Competitive constraints are placed between all interpretation node pairs of the same part to enforce that only one interpretation can be active at a time. We make this competitive constraint a hard constraint by assigning an edge probability of 1, or $q_{ij} = 1, \forall q_{ij} \in E^{-}$.

Occlusion constraints are handled by placing an exclusion edge between any nodes that overlap in the image by more than some threshold. This overlap is computed using intersection-over-union of the part regions $IoU(x_i, x_j) > \tau$, and the probability placed on the edges are simply $q_{ij} = \text{IoU}(x_i, x_j), \forall q_{ij} \in E^{-}$. Compatibility between parts is dictated by the pairwise geometric relations in the prior model, which we will write as $q_{ij} = \frac{1}{Z} exp\{-\mathcal{E}_{geom}(x_i, x_j)\}, \forall q_{ij} \in E^{+}$. The normalization Z is computed locally such that $Z = \sum_{(i,j)\in E^{+}} q_{ij}$.

Next, we incorporate the part appearance likelihood into the model:

$$p(X|\mathbf{I}) \propto \exp\left\{\frac{1}{T}\left[\sum_{e_{ij}\in E} \beta_{ij} \cdot c_{ij}(x_i, x_j) + \sum_{i=1}^{N} \log \frac{p(\mathbf{I}_{\Lambda_i}|x_i)}{q(\mathbf{I}_{\Lambda_i})} \cdot \mathbf{1}(x_i \neq 0)\right]\right\}. \tag{4.73}$$

Nodes assigned a value of zero indicate that the part is not included in the solution, and therefore its appearance does not contribute to the probability.

Proposals for the candidacy graph are populated from running part detection for each of the relevant appearance models. The value of k, relates to how many competing solutions we wish to reason about simultaneously. Even when $k = 1$, however, the sampler can often jump between valid solutions in a single iteration to recover multiple solution by drawing repeated samples.

The result of the sampling is a set of proposal labelings with low energy. These labels favor the grouping of compatible parts and exclusion of incompatible parts, but does not enforce the composition rules of the grammar. To accomplish this, the sampled solution of the candidacy graph is divided into layers, each layer containing all nodes with the same label. Ignoring the layer with label 0 which indicate nodes that are turned *off*, all valid parses are enumerated for each layer to produce proposals of the root part.

This bottom-up process is applied recursively to the full AND-OR graph. First, part detectors are run for all parts, and stored in a proposal list for each part. For each OR-node encountered in the recursion, we populate a candidacy graph from proposal lists of the children OR-nodes. These lists can contain proposals found directly through detection, as well as proposals found through cluster sampling. Any pair of proposals that overlap are given negative edges with $q_{ij} = 1$ to indicate that only one can be chosen. Once the root node is reached, the cluster sampler is run one last time with only competitive constraints among parts of the root type. The result of this is a set of high probability scene parses.

After running the partitioning, we are not guaranteed that the resulting set of parts forms a valid parse. This is because the C^4 algorithm is only grouping mutually compatible parts, without adhering to any of the grammar production rules. Therefore, after each partitioning step, a search is performed to find the highest probability parse from the partition to add to the candidacy graph. Furthermore,

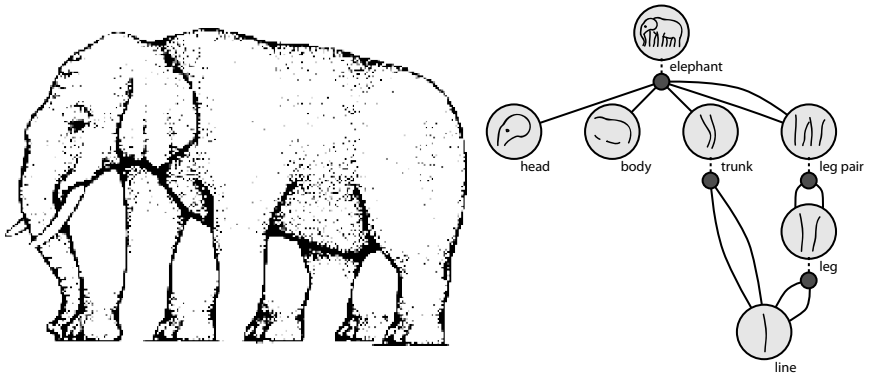

Figure 4.20: Elephant illusion AND-OR graph: this classic optical illusion forms an inconsistent shape, yet has two reasonable interpretations. The elephant is modeled hierarchically using the AND-OR graph shown on the right. For simplicity, all OR-nodes have only a singe selection. This representation will be used in our example for how the ambiguity model can recover these alternate interpretations.

when we have noisy or unreliable detections, we can introduce a top-down phase as described in Section 4.3.2 to help find missing proposals. The bottom-up, top-down, and partitioning process are then repeated for several iterations for each AND-node of the grammar.

4.5.5 Toy example: elephant illusion

To illustrate how cluster sampling provides ambiguity reasoning to the parsing process, we demonstrate a simple toy example using the popular elephant illusion shown in figure 4.20. For simplicity, we construct and an-or graph with only one possible selection on each OR-node. The focus of the example is to provide ambiguity reasoning about the legs and trunk, which can be formed from a front edge and a rear edge. Each edge therefore has an ambiguous interpretation as

either front or rear. Legs are composed into pairs, which again have ambiguous interpretations as the front legs or hind legs.

Our assumptions are that there all detections are true positives, and that $k = 2$. We begin with a set of bottom-up "detections", which consist of only lines, the head, and body. We assume there are no false detections although the algorithm will still work in their presence. Compatible relations are placed between parts if the distance between them lies within a defined range. Competitive relations are placed between any competing interpretations, as well as any overlapping parts. The initial state of the sampler is set to a reasonable solution for brevity. The same results should still be obtained from a random initialization.

The algorithm begins bottom-up by composing line segments into leg parts, which is illustrated in figure 4.21. Convergence can often occur in a very small number of iterations due to the ability to make large configuration changes in a single step. In this case, there are no compositions that compete for the same part interpretation, and two distinct solutions can be extracted from a single labeling of the sampler. Continued iterations of the sampler will likely generate the same two solutions, but with potentially interchanged labels, which is shown between iteration 2 and 3 in the figure. Finding trunk proposals from line segments is sampled in exactly the same way.

Leg proposals are extracted from the sampling results, and the process is repeated to form leg pairs. Finally, the root of the AND-OR graph is reached, which composes the full elephant from proposals of leg pairs, head, body, and trunk, which is illustrated in figure 4.22. In this case, there is more competition between leg pair proposals to group with the body and head, which have only a single proposal. In this case there are two alternative interpretations of the elephant. In order for the sampler to jump between these interpretations it must

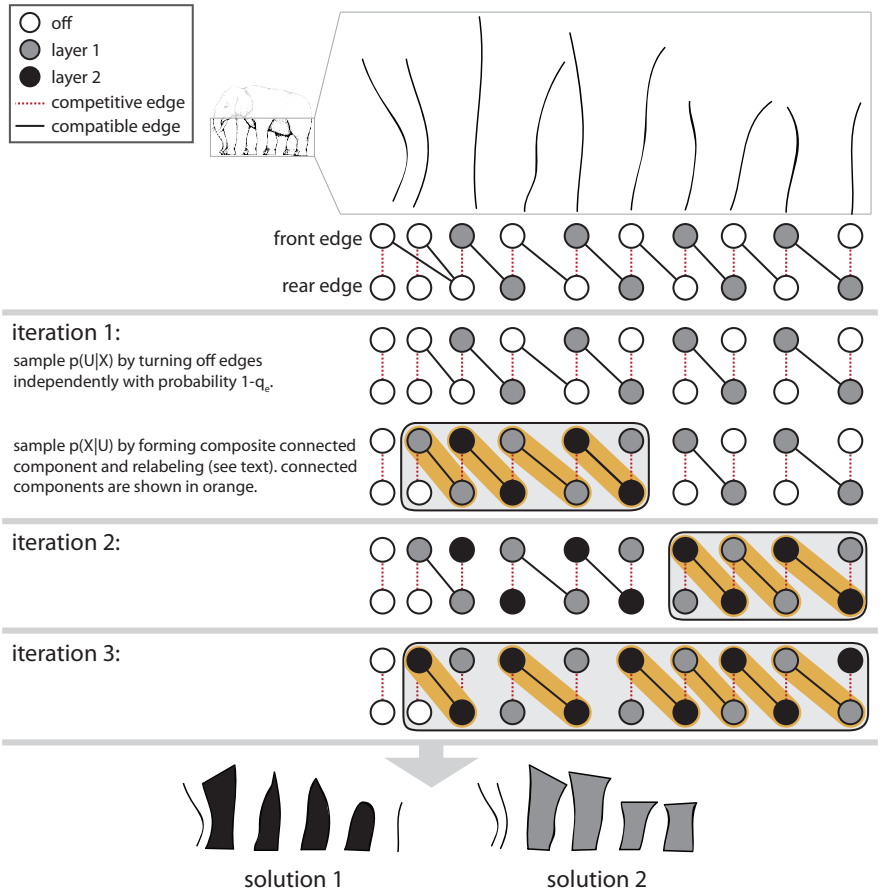

Figure 4.21: Finding legs from lines: composite cluster sampling is used to group line interpretations together to form legs in a manner that encourages compatible interpretations to be grouped, and discourages incompatible ones.

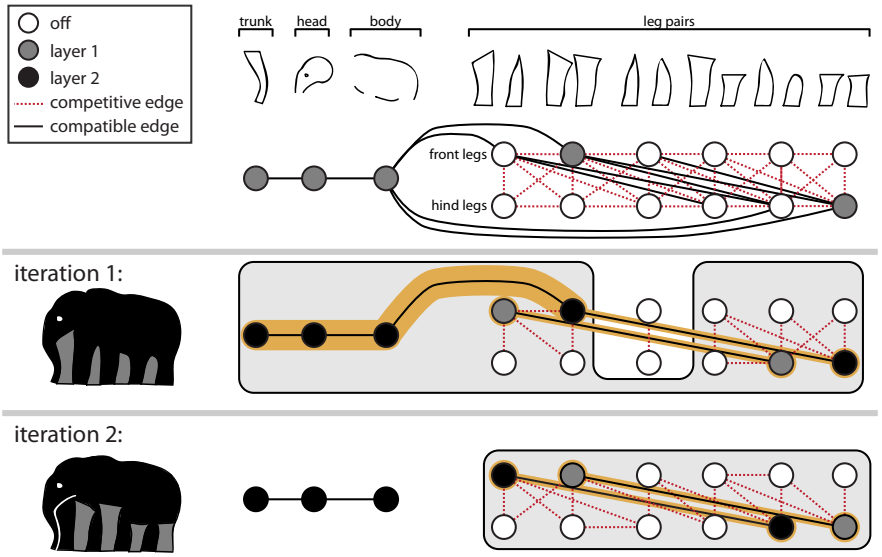

Figure 4.22: Finding elephants from parts: when the algorithm finally reaches the root node in the AND-OR graph, the cluster sampling output correspond to scene interpretations. In this case, there is competition between the body part for the two different leg interpretations, and the sampler will jump back and forth between these interpretations on successive iterations. These two interpretations are shown in iteration 1 and 2.

break the connections between the body and legs in the $p(U|X)$ step, the legs will then cluster and exchange labels and finally merge back with the body in the $p(X|U)$ step. This is illustrated in iteration 2 of figure 4.22

4.5.6 Human image experiments

For conducting experiments on images, we use a 6-part upper-body model consisting of head, torso, and two arm segments for each arm. Arm appearance models are learned using Active Basis templates, described in Section 4.2.1. Detecting parts such as the arm segments are particularly unreliable, we employ a top-down

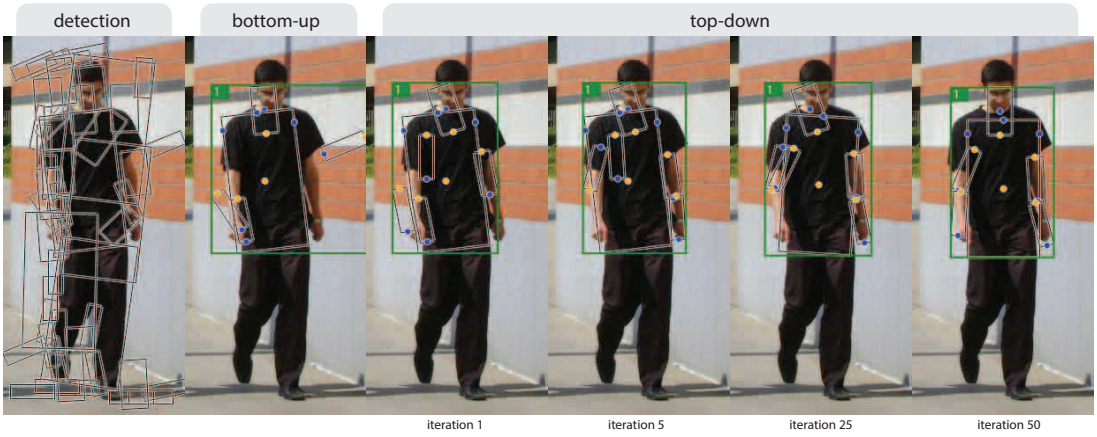

Figure 4.23: When populating the candidacy graph with unreliable part detectors, some parts such as the arm segments will often be missing using bottom-up proposals alone. This is addressed by adding a top-dow prediction step to help find missing part proposals. Here, only the top scoring candidate is shown after being composed bottom-up, after which the top-down Gibbs sampling is run for 50 iterations.

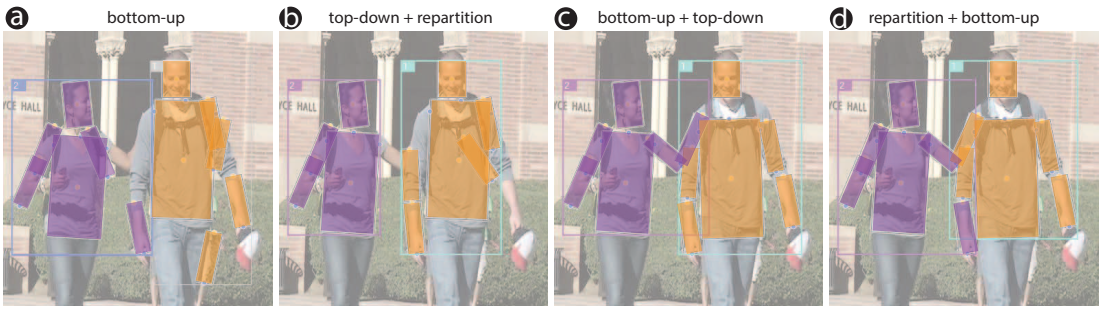

Figure 4.24: Intermediate results on an ambiguous image. The algorithm converges to a state where each person is placed on different layers with the ambiguous parts, being the arm segments between them, jumping between layers.

prediction step after the bottom-up detection to help find parts initially missed. We illustrate one iteration of the bottom-up + top-down process in Figure 4.23, which shows the highest scoring candidate being composed and sampled.

A running example on a multi-person image is shown in Figure 4.24, with ambiguity in the arms. This example is run using a Potts model with 3 layers, parts in layer 0 are not drawn, layer 1 is purple, and layer 2 is orange. Due to poor detection results on the arm segments, the initial bottom-up proposals in (a) are quite poor, but mostly corrected after top-down prediction in (b). After repartitioning, some of the poorly matching parts are turned *off* and replaced by new top-down predictions in (c). In this case, the sampler converges with each person on different layers, and the ambiguous arm jumping between layers, shown in (c) and (d).

In Figure 4.25, we show four iterations on a crowd example. One of the individuals in this case has his arms occluded by his neighbors. Again, we use a Potts model with 3 layers. The part colors in this case correspond to the 4 top object proposals, and the bounding box around the proposals designates the layer assignment. After the first iteration, the algorithm finds a stable localization of

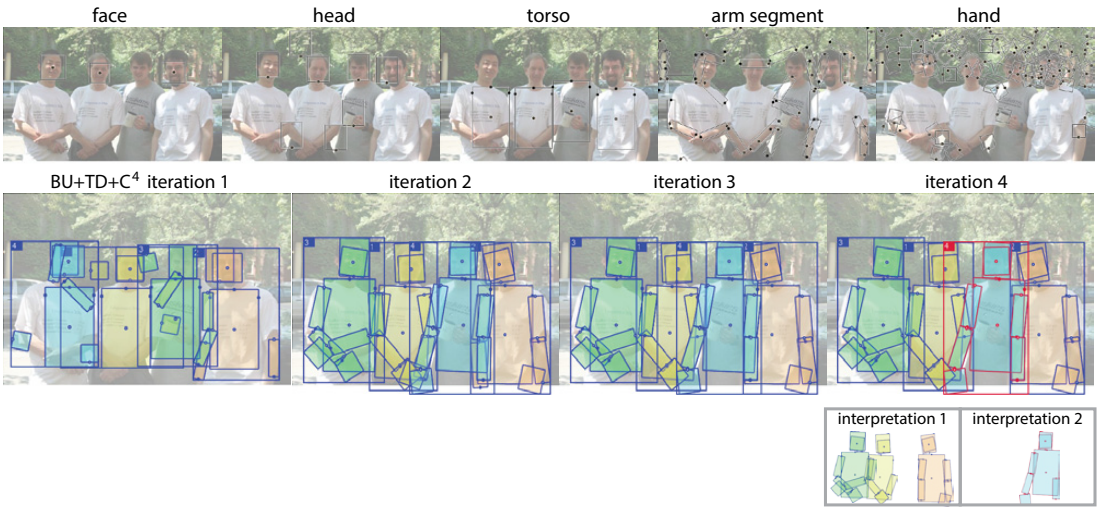

Figure 4.25: Parsing ambiguous group images. The top rows contain the initial bottom-up part detections. The bottom row contains results from four iterations of bottom-up composition, top-down prediction, and C^4 sampling. The top 4 object candidates are drawn with different colors, and the color of their bounding box indicates the assigned layer in the Potts model. After four iterations, the algorithm moves the conflicting body to another layer thus resolving the ambiguity.

most of the parts, and at the fourth iteration eventually moves the ambiguous individual to its own layer and thus resolving the ambiguity.

CHAPTER 5

Discriminative Methods

In this chapter we focus on learning the grammar from a discriminative perspective. The general approach is to learn the model by minimizing the empirical risk of the posterior directly, as opposed to maximizing the data likelihood in the generative case.

There are several differences in the construction of the discriminative grammar from our generative approach. First, the appearance models are no longer based on Active Basis or HIT, which are sparse template models learned individually through pursuit. Instead, the appearance templates are now dense and all parts are learned jointly during training. Second, the grammar productions are structured as a dependency grammar which instead links parts together into a compositional chain or tree structure as described in Section 3.3.2. The choice of using the dependency grammar over a coarse-to-fine phrase-structure grammar is motivated by the datasets we intend to evaluate on. The PARSE (Figure 2.4) and Leeds (Figure 2.5) datasets are prominent benchmark datasets in the literature, for which nearly all modern pose estimation techniques are compared on. Both of these datasets are composed primarily of highly deformed poses that are often crumpled or contorted, and not particularly suited towards coarse-to-fine modeling. Furthermore, the dependency grammar is more efficient to compute, which is significant in keeping the model training time within practical limits. The parameterization of the model is otherwise identical to the generative case.

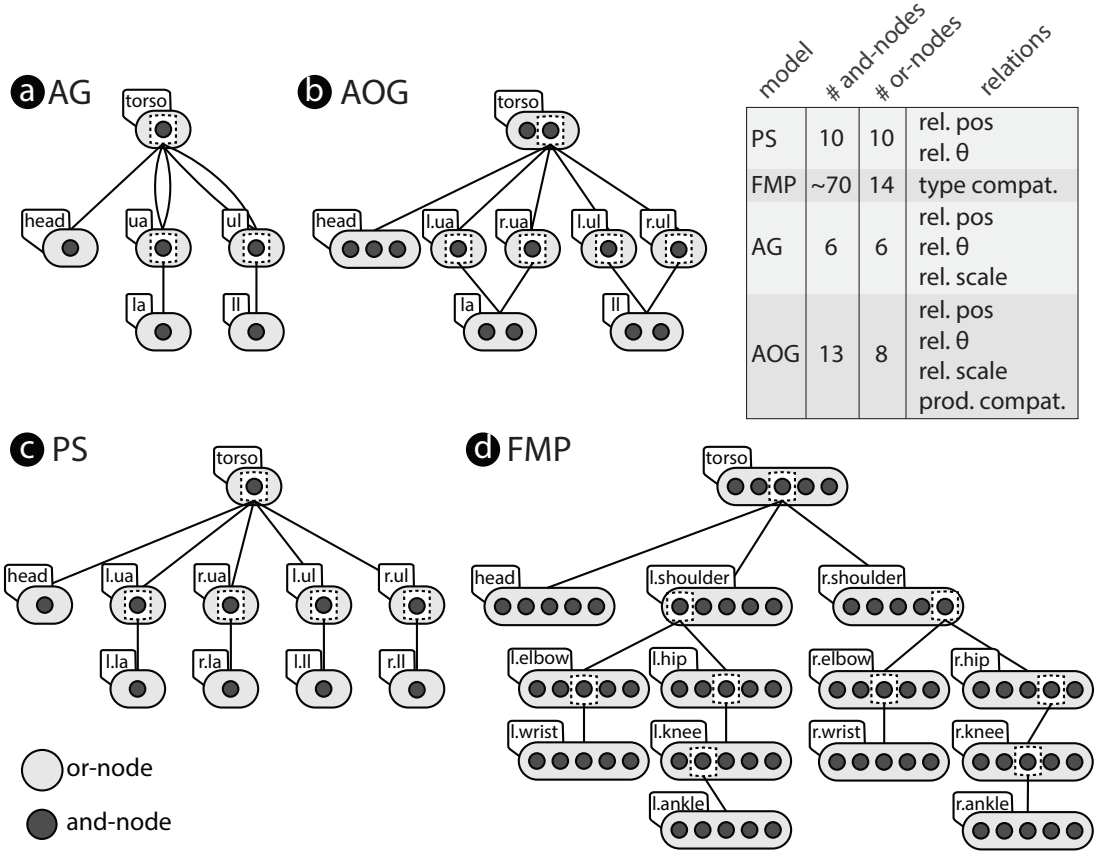

Figure 5.1: Several common models for pose estimation are shown using our and-or graph notation. And-nodes represent distinct part appearance models, while or-nodes can be treated as a local mixtures of and-nodes. Edges represent the contextual relations between parts, which are specified for each model using the table on the right. Pictorial structures [FH05] (c) has a fixed structure with no shared parts, and uses conventional articulation relations over relative position and orientation. The flexible mixture-of-parts model [YR11] (d) emulates articulation with a large number of orientation-specific parts and mixtures, using relations only between mixture selections (types). Our baseline and-graph model (a) has similar structure and relations with PS but shares parts between left and right sides and uses relative scale relations. Our final and-or-graph model (b) extends (a) by utilizing several part variants, and compatibility relations between variants (productions).

For the following experiments, we define two models called AG (AND graph) and AOG (AND-OR graph). The AG model is the baseline model that only defines a single grammar production for each part, and therefore has no structural variability because the production selections are deterministic. We use this model to define a baseline case for comparing more complicated grammars and evaluating the difference in performance. The AG model is rooted at the torso, which has dependency relations to head, arms, and legs, which further branch out to the extremities. This model is very similar to the conventional pictorial structures (PS) [FH05] model, with the exception that we are still using shared parts and have a richer geometric representation. The AOG model expands on the baseline model by adding several productions to each of the parts. To illustrate these models and compare against other common models used in the literature, Figure 5.1 represents each model as an AND-OR structure. The table on the right indicates the state space of each of the parts – effectively the continuous nature of model, whereas the AND-OR structure describes the discrete nature.

5.1 Appearance features and region segmentation

The appearance templates for each AND-node production are based on the Histogram of Oriented Gradient (HOG) feature descriptor of Dalal and Triggs [DT05]. The conventional implementation of HOG divides the image into a grid of cells, each cell typically being 8×8 pixels. The image gradients of each pixel in the cell are quantized into a fixed number of orientations, and pooled into a histogram. Lastly, these histograms are locally normalized using the gradient histograms of neighboring cells in a scheme they call block normalization. The limitations of computing HOG in this manner is that the features can only be evaluated on the cell boundaries, in this case every 8 pixels, and there is no sim-

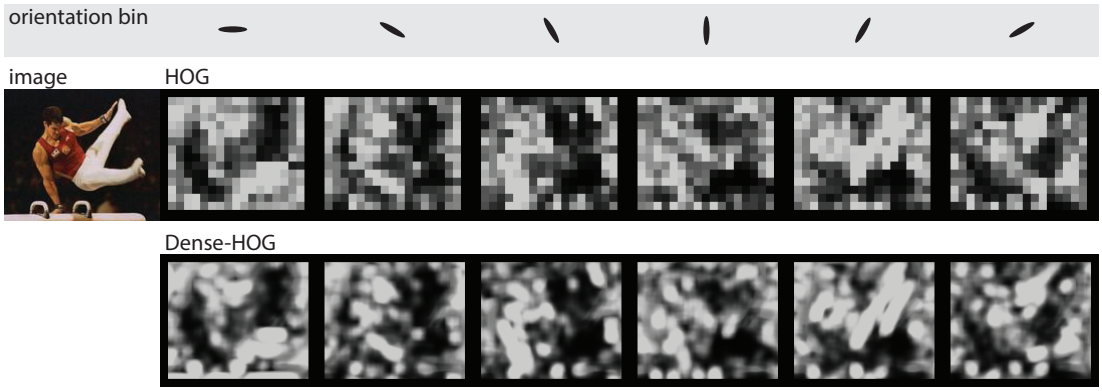

Figure 5.2: Histogram of oriented gradients (HOG) vs. Dense-HOG feature responses. The conventional implementation of HOG pools gradients from fixed cells in the image, resulting in the block structure shown in the top row. This representation is not conducive to computing part responses at arbitrary locations or orientations. To overcome this, our Dense-HOG implementation performs the same pooling, but centered around every pixel in the image to produce the response maps on the bottom row.

ple method to extract features for rotated parts. To address these limitations, we utilize the same grid cell pooling and block normalization, but compute the feature for a cell centered around every pixel in the image. To illustrate this, Figure 5.2 visualizes the conventional HOG response on the top row by rendering the value of each orientation bin in its own map. Note that the value of the feature for a given orientation is fixed within each 8×8 cell. The Dense-HOG feature we compute is visualized in the same manner on the bottom row, which now has smooth transitions between cells.

Each production defines an appearance template that specifies where to extract features responses from the image for a given part state, as illustrated in Fig.5.3. To keep the feature length reasonable, the Dense-HOG feature map is subsampled within the part boundaries, typically every 6 pixels. To compute part appearances responses at different scales, a fixed-sized template is applied to different levels of the image pyramid. When parts are rotated, features are extracted by rotating the template, then bin-shifting the gradient histograms to compensate for the rotation.

The gradient features of HOG only capture local edge phenomena, however, and cannot distinguish if those edges are on the boundary or interior of a region. Furthermore, local normalization is necessary to gain robustness to varying lighting conditions, but will also amplify the edge strength in flat or textured regions increasing the likelihood of a part spuriously scoring highly within these amplified background edges. As an example, Figure 5.4(d) shows the response map of a trained part template using only edge features, for which there is a considerable number of local maxima in the background clutter. To compensate for this effect, we introduce a region feature to the part appearance template to measure how strongly the interior region of the part stands out against its surrounding

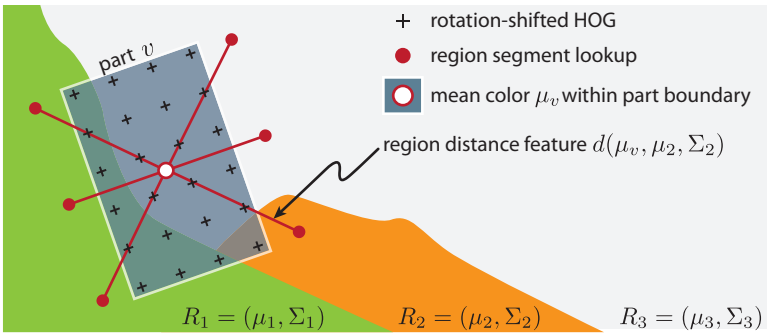

Figure 5.3: **Part appearance template**: The template utilizes features from both foreground and background. Foreground features use a rotation-shifted variant of HOG [DT05] collected along a uniform grid, as well as the mean color within the part boundary. Background features consist of distance measures between the mean part color and adjoining external region segments. The template defines multiple background sample points around the perimeter of the part, each of which retrieves the region segment that contains the point and compares it with the interior mean color.

background.

First, we present a background segmentation model. The image background is represented as a collection of large disjoint regions, where the appearance within each region is well explained using a multivariate Gaussian in $L^*u^*v^*$ color space. Furthermore, we assume that the background regions are large compared to the size of the foreground parts and treat the background process as independent of the foreground. This independence is chosen to avoid the intractable computation of reestimating the background segments for every part state.

Let Λ denote the pixel lattice of image I, which is partitioned into K disjoint regions $\bigcup_{i=1}^{K} \mathcal{R}_i = \Lambda$, $\bigcap_{i=1}^{K} \mathcal{R}_i = \emptyset$. The segmentation of the image is represented as $\mathcal{S} = (K, \{(\mathcal{R}_i, \mu_i, \Sigma_i);\ i = 1, 2, ...K\})$. Each region is assumed to be generated independently and normally distributed, thus the image likelihood is $p(I|\mathcal{S}) = \prod_{i=1}^{K} \mathcal{N}(\mu_i, \Sigma_i)$. A prior model encourages the number of regions to be small, region volumes to be large, and boundaries smooth:

$$p(\mathcal{S}) \propto p(K) \prod_{i=1}^{K} p(R_i) \propto \exp\left\{ -\lambda_0 K - \sum_{i=1}^{K} \mu \oint_{\partial R_i} ds + \gamma |R_i|^c \right\} \tag{5.1}$$

where ∂R_i represents the boundary around region R_i. The prior on the region volumes $e^{\gamma |R_i|^c}$ is a scale factor motivated from [MG01] and [TZ02] and related to how "busy" the image is. The optimal segmentation maximizes the posterior $p(\mathcal{S}|I) \propto p(I|\mathcal{S})p(\mathcal{S})$.

To learn the parameter for this model, we adopt data-driven MCMC approach of [TZ02] to maximize the segmentation posterior using $scale = 1.0, \gamma = 2.0, c = 0.9$. Please refer to the original work for a full explanation of the algorithm.

The foreground region consists of the pixels contained within the rectangular part boundaries of the part, and are modeled as a single mean color in $L^*u^*v^*$ and can be computed efficiently using integral images. The region contrast fea-

ture can now be defined as a distance measure between the foreground region, and the adjacent background region distributions. Because the background distribution are all modeled as a Gaussian distribution, the Mahalanobis distance is used.Given a foreground mean μ_v and a background region (μ_i, Σ_i), the distance is computed as

$$d(\mu_v, \mu_i, \Sigma_i) = (\mu_v - \mu_i)^\top \Sigma_i^{-1} (\mu_v - \mu_i). \tag{5.2}$$

This distance can be interpreted as the negative log-probability that the average pixel in the foreground region is generated by the background process. To account for the possibility that multiple background region segments can adjoin the part, the template defines multiple region features that are equally spaced around the part periphery, as shown by the circles in Fig.5.3. The output of the segmentation model is illustrated in Fig.5.4, as well as visualizations of template scores separated into individual edge and region response maps. The final score map would be the sum of these two maps.

The full appearance response vector $\phi^a(I, t, v)$ can now be computed as a concatenation of responses from each rotation-shifted gradient histogram feature, and region distance feature in the template. The appearance score is then

$$f^a(v, I) = \langle \lambda^a_\omega, \phi^a(I, t_\omega, v) \rangle. \tag{5.3}$$

5.2 Occlusion reasoning

Humans in natural scenes and general position exhibit a substantial amount of self-occlusion, typically of the arms and legs. Furthermore, humans often occur in cluttered scenes with many potential occluding objects, such as other people, cars, trees, etc. Without an explicit representation of occluded parts, the model will attempt to train appearance models for all parts as if they were fully visible.

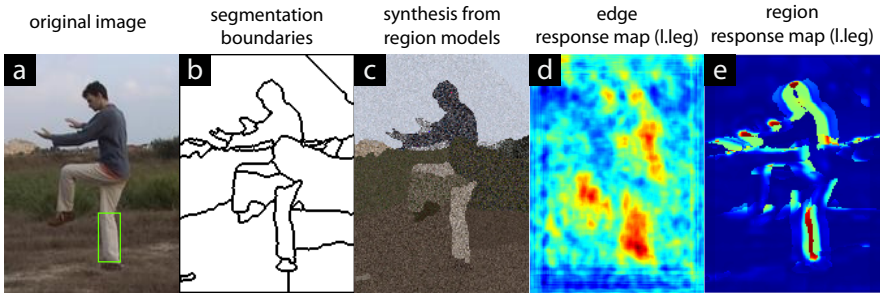

Figure 5.4: Appearance response maps for edge and region portions of the template. Segmented regions are shown in (b), and a resynthesized image sampled from the region models is shown in (c) to illustrate the model fit. Score maps from the trained model of the l.leg part in the vertical orientation are shown using only HOG features in (d) and only region distance features in (e). Due to local normalization, spurious foreground responses tend to appear particularly around textured regions, whereas the background feature is far more stable in these regions.

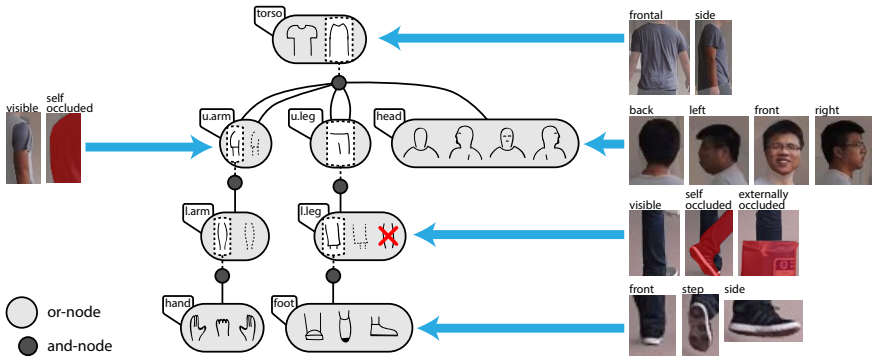

Figure 5.5: Incorporating part occlusions into the AOG. To avoid training appearance models with occluded confounders, the grammar can naturally define different productions for visible, self-occluded, and externally occuded parts.

This can be destructive, since we are effectively contaminating parts defined as true positives with corrupted examples where the appearance is of the occluder, and not the actual part.

Our grammar provides a natural representation for these parts by simply defining a new production for the self-occluded and externally occluded variants of these parts. This provides the occluded parts with their own geometries to learn the distribution of local pose configurations that are are correlated with occluded parts, as well as their own appearance models. These occluded appearance models are in actuality models of the occluder, however, and if the appearance of the occluder is truly independent of the occluded part geometry then the discriminative training will assign the part appearance weights to zero. We find this is often not the case, and observe that the model can often learn appearance features that can help localize occluded parts. In Figure 5.5, we illustrate an example AND-OR grammar with productions to encode different appearances, viewpoints, and both self-occlusion and external occlusion.

5.3 Inference

Exact inference of the model can be computed efficiently using a dynamic programming algorithm. Because we are using a dependency grammar in this case, the relation structure is more simplified than the phrase-structured case used in our generative grammar. In particular, for every production ($\alpha \rightarrow \beta$) the α part is always the proximal part, and the β parts are always the distal parts. This ensures a tree-structured model, for which the dynamic programming algorithm described in Section 4.3.3 can be used.

Because the dependency grammar relations are always star-structured, the DP

algorithm for this model can be further simplified from the algorithm described in Section 4.3.3. In particular, the function MaxAND maximizes constituent part states within the AND-node production, which previously contained a chain or tree structure of parts, all conditioned on the AND-node selection itself. In the dependency grammar case, there is exactly one relation that connects the root part of the AND-node with any of the dependent parts. We therefore do not need to recurse through the relational chain or tree of constituent nodes, and can simply optimize out the relation for each dependent part in one step. As a consequence, this avoids the repeated maximization over context-sensitive production selections between AND-node constituents (lines 6 and 7 in Algorithm 6), which are very costly to compute.

The basic unit of computation is computing a maximal score map for the proximal part of a production. Each of the distal parts are conditionally independent given the proximal part, and can be maximized individually. The maximal score map for part state v_i given production ω_i can be expressed recursively as

$$\begin{aligned} M(v_i|\omega_i) &= f^a_{\omega_i}(v_i, I) + f^{g1}_{\omega_i}(v_i) + f^{c1}_{\omega_i} \\ &+ \sum_{(v_i,v_j)\in R_{\omega_i}} \max_{v_j} \left[f^{g2}_{\omega_i}(v_i, v_j) + f^{c2}_{\omega_i}(v_i, v_j) + M(v_j|\omega_j) \right]. \end{aligned} \tag{5.4}$$

Although the production for part v_i is fixed, we must maximize over the full state of the distal parts v_j, including the distal production. The maximization over positions (x_j, y_j) can be computed very efficiently using distance transforms [FH04] that have linear complexity in the number of positions. The maximization over scale s_j, orientation θ_j, and production ω_j each require quadratic time to compute. The state space for these remaining variables is still quite small, however, and the computation is tractable.

To infer the maximal scoring parse, we recurse through the grammar start-

ing from the root symbol s_0. Terminal symbols have no distal parts, and their maximal score maps consist of only the appearance and unary potentials. Once the maximal score maps are computed for every production, the maximal parse score can be obtained by maxing over all productions that have the root symbol as the proximal part

$$\max_{p_j \in P \text{ s.t. } \alpha_j = s_0} \max_{v_i} M(v_i | p_j). \tag{5.5}$$

The parse tree can be recovered by replacing the max operators with arg max and backtracking through the optimal state maps.

5.4 Learning

The score of a parse can always be expressed as the inner product of the full model weight vector and a response vector for the entire parse $f^{\mathcal{G}}(pg, I) = \langle \lambda, \phi(pg, I) \rangle$. The model weights λ parameterizes a family of parsers that output the maximal scoring parse $F^{\mathcal{G}}_{\lambda}(I) = \arg\max_{pg} f^{\mathcal{G}}(pg, I)$ for a given grammar. We define the learning task as the search for a weight vector such that the empirical risk of the associated parser is minimized, which is computed as the expected loss on the training dataset D. Let $\bar{pg}$ be the ground truth parse. The optimal weights are

$$\lambda^* = \arg\min_{\lambda} E_{(\bar{pg}, I) \sim D}[L(F^{\mathcal{G}}_{\lambda}(I), \bar{pg})] \tag{5.6}$$

The loss is defined on the structured output space of parses, and must measure the quality of a predicted parse against the ground truth parse. In a general grammar, these parses may have different structure or a different number of parts, making the formulation of such a loss sometimes difficult. All parses from the grammars we define here, however, have the same number of parts and the same branching structure which allows us to compute loss as the sum of part-wise terms. Our loss is motivated by the PCP evaluation metric [EF09], which

computes a score based on the proximity of the part endpoints to the ground truth endpoints. A part is typically considered detected when the PCP score is under 0.5. The loss function is

$$L(pg, \bar{pg}) = \frac{1}{|V(pg)|} \sum_{v \in V(pg)} \min\left(2 \cdot \text{dist}_{pcp}(v, \bar{v}), 1\right) \tag{5.7}$$

and is bounded between 0 and 1 taking the value 0 only when identical to the ground truth.

To make the learning computationally tractable, we instead minimize a convex upper bound to this loss using the so-called margin-scaled structural hinge loss from [TGK03], resulting in the following max-margin structural SVM objective function

$$\begin{aligned} &\min_{\lambda} \frac{1}{2}||\lambda||^2 + \frac{C}{|D|} \sum_{i=1}^{|D|} \xi_i \\ &s.t.\ \ \lambda^\top \left[\phi(\bar{pg}_i, I_i) - \phi(pg, I_i)\right] \geq L(pg, \bar{pg}_i) - \xi_i \\ &\forall pg \in \Omega_{\mathcal{G}}, \forall i. \end{aligned} \tag{5.8}$$

Due to the exponential number of constraints, it is intractable to minimize this expression directly. Instead, it is still provably efficient to solve this minimization incrementally by adding only the most violated constraints at each iteration, using the following maximization as the so-called separation oracle

$$\hat{pg}_i = \arg\max_{pg} \lambda^\top \phi(pg, I_i) + L(pg, \bar{pg}_i). \tag{5.9}$$

This maximization is commonly referred to as a loss-adjusted inference problem. The complexity of this maximization depends on the formulation of the loss function. This is the primary reason we choose an additive loss function, which can be incorporated into the existing inference algorithm in a relatively straightforward manner without impacting the computational complexity.

It can be more clearly seen that this objective is an upper bound for the risk by rearranging the terms of the most violated constraints $\xi_i \geq L(pg, p\bar{g}_i) + \lambda^\top \phi(p\hat{g}_i, I) - \lambda^\top \phi(p\bar{g}_i, I)$. Because the score of the parse $\lambda^\top \phi(p\bar{g}_i, I)$ can never be greater than the score of the maximal parse $\lambda^\top \phi(p\hat{g}_i, I)$, the right-hand-side of the expression can never be lower than the loss.

This minimization can be solved by a multitude of methods. A dual coordinate descent solver was implemented for [YR11], and the cutting plane method of [THJ04] is also commonly used. For our implementation, we maximize the dual objective using the QP solver of Franc and Hlavac [FH06].

One practical consideration is that when using a Gaussian potential function for the joint displacement, the learned parameters correspond to the inverse variance and must be positive. To prevent the optimizer from assigning negative weights to these parameters, we can implicitly introduce positivity constraints into the primal objective function in Equation 5.9. The primal constraints take the form $\lambda^\top z_j \geq b_j - \xi_i$ where z_j is the response difference from the separation oracle $\phi(p\bar{g}_i, I_i) - \phi(pg, I_i)$ and b_j is the loss. Let k be the index of the parameter we wish to impose a positivity constraint on. By initializing the optimizer with a set of constraints such that z_j is a unit vector selecting the k'th element of λ, and setting $b_j = 0$, we have the constraint $\lambda[k] \geq 0 - \xi_i$. This is a soft constraint, however, because the optimizer can always compensate for a negative $\lambda[k]$ by increasing the slack variable ξ_i. To effectively make this a hard constraint, the unit vector z_j can be scaled by a large constant.

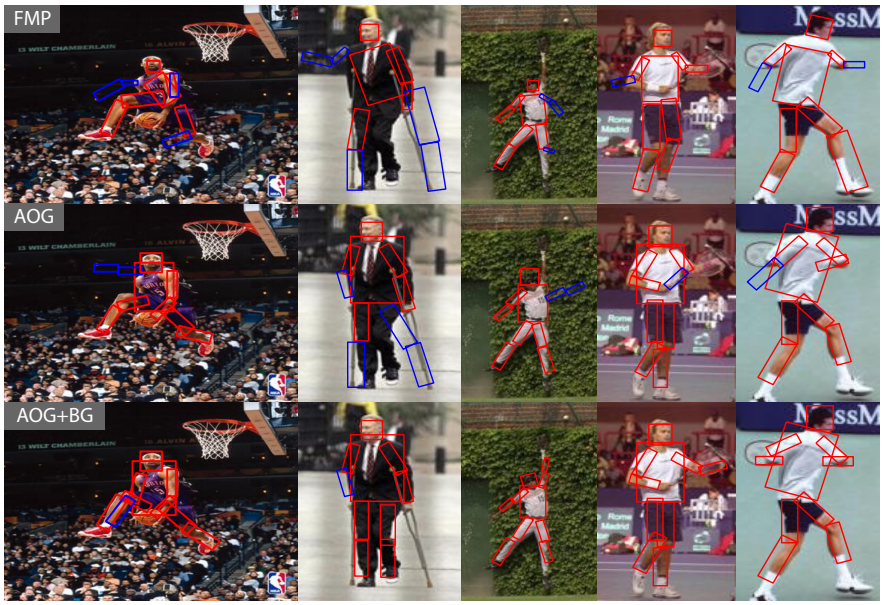

Figure 5.6: **Influence of region features**: A selection of results from the AOG and AOG+BG models compared with FMP [YR11]. Textured regions are problematic for both FMP and AOG models, leading to frequent spurious limb detections in these regions. The AOG+BG model includes terms to favor part regions that are distinct from their adjoining background process, and can correctly localize many of these parts.

5.5 Evaluation

We train and evaluate our method using three different grammar models to illustrate the impact on performance from the addition of reconfigurable parts as well as the background model. For all cases, we discretize the state space of the parts to be 25% of the image width and height, and use 24 part orientations.

AG: And-graph grammar is our baseline model, shown in Fig.4.7(a), and is the simplest possible model in our framework to represent the full articulated body. Each symbol has only one production, and all limb parts are shared between the left and right sides. This construction is equivalent to a pictorial structures model (Fig.4.7(a)) with shared parts.

AOG: And-or-graph grammar, shown in Fig.4.7(b), using productions { front, side } for torso, { left, front, right } for head, and { visible, occluded } for both l.arm and l.leg. Separate symbols are used for left and right u.leg as well as u.arm because side-specific features tend to be strong for these parts. The l.leg and l.arm symbols are still shared between sides.

AOG+BG: This is the same grammar as AOG, but with the addition of the background terms. These terms are only included on the the productions for l.arm and l.leg. To illustrate the influence of these features, Fig.5.6 shows several examples where the top scoring pose erroneously matches to a strong edge with poor region support, but is corrected when retraining with the background feature.

5.5.1 Protocol

Evaluation follows the PCP protocol described in Section 2.3. Although this is largely the standardized protocol, we point out two notable consistencies that can

significantly affect the calculated performance results. First is the existence of zero-length parts in the annotations, which are impossible to detect according to this metric. Second is the inconsistency or genuine ambiguity of labeling a limb as left or right. Both datasets that we evaluate on have multiple cases where the left/right annotations are inconsistent with the rest of the dataset. Even perfect results on these examples will still get the limb parts counted as wrong because the left limb is being evaluated against the right side and vice versa. To compensate for this, for each selected skeleton we exchange the left and right labels for arm and leg separately, and take the configuration that has the highest number of correctly localized parts. We mark results evaluated in this way with a † symbol, all other results are evaluated such that the performance results are comparable to published methods to the best of our knowledge.

5.5.2 Results on PARSE

Introduced by Ramanan [Ram06], and described in Section 2.2, this dataset contains 305 images from which the model is trained on the first 100, and evaluated on the remaining 205. For each part, we provide an additional annotation to indicate a production label. Performance on this dataset is tabulated in Table 5.1 along with all known recent published results. We observe a performance gain of 2.5% between the AOG and baseline AG model, and a 4.0% gain between AOG+BG and AG. Furthermore, the AOG+BG model outperforms the current published state-of-art for all parts individually, with an average improvement of 1.6%. Selected results with both successes and failures are shown in Figure 5.7. Common failure modes primarily consist of matching limbs to background and double-counting of limb appearances.

Dataset	Method	torso	head	u.leg	l.leg	u.arm	l.arm	avg
PARSE	[Ram06] (2006)	52.1	37.5	31.0	29.0	17.5	13.6	27.2
	[ARS09] (2009)	81.4	75.6	63.2	55.1	47.6	31.7	55.2
	[JE09] (2009)	77.6	68.8	61.5	54.9	53.2	39.3	56.4
	[SNH10] (2010)	91.2	76.6	71.5	64.9	50.0	34.2	60.9
	[JE10] (2010)	85.4	76.1	73.4	65.4	64.7	46.9	66.2
	[JE11] (2011)	87.6	76.8	74.7	67.1	67.4	45.9	67.4
	[TZN12] (2012)	97.1	92.2	85.1	76.1	71.0	45.1	74.4
	[YR11] (2011)	97.6	93.2	83.9	75.1	72.0	48.3	74.9
	[DR12] (2012)	-	-	-	-	-	-	77.4
	Ours (AG)	99.5	95.6	81.8	67.0	74.3	54.6	75.0
	Ours (AOG)	**100.0**	96.2	87.0	75.3	73.2	53.9	77.5
	Ours (AOG+BG)	99.5	**97.4**	**88.4**	**78.0**	**74.1**	**56.1**	**79.0**
	Ours (AOG+BG)†	99.5	97.4	89.2	78.3	74.6	56.9	79.5

Table 5.1: Benchmark evaluation results on the PARSE dataset. We evaluate our baseline model (AG), grammar model (AOG), and grammar model with background features (AOG+BG) against all known published results on this dataset. The † symbol indicates the use of a modified evaluation protocol, see text for details.

5.5.3 Results on Leeds

Introduced by Johnson and Everingham [JE10], and described in Section 2.2, this dataset consists of 1000 images each for training and testing. An additional 10,000 training images are also provided by [JE11] as an extension to the dataset, which we do not use. In the same manner as PARSE, we provide an additional production label to each part. Our AOG+BG model also outperforms the published state-of-art for all parts on this dataset, by an average gain of 9.2%. Our results are tabulated with performance results of recently published work in Table 5.2.

The contribution of the background feature is minimal on this dataset, however, which we believe may be attributed to the narrow crop margins and general lack of large background regions. Selected results with both successes and failures are shown in Figure 5.7. Common failure modes primarily consist of matching limbs to background and double-counting of limb appearances, similar to what we observe in the PARSE evaluation, but we also struggle to parse many of the extreme perspective examples.

Much of the published work on this dataset has been ambiguous on the exact evaluation protocol. In particular, whether the part detection criteria uses the average endpoint distance or the maximum endpoint distance, and whether the ground truth object bounding box was used as a filter to select the parse to evaluate as opposed to taking the globally highest scoring parse. We post results for each non-standard variant, shown in red on Table 5.2. These variations are described in detail in Section 2.3.

Dataset	Method	torso	head	u.leg	l.leg	u.arm	l.arm	avg
Leeds	[JE10] (2010)	78.1	62.9	65.8	58.8	47.4	32.9	55.1
	[TZN12] (2012)	95.8	87.8	69.9	60.0	51.9	32.9	61.3
	[JE11] (2011)	88.1	74.6	74.5	66.5	53.7	37.5	62.7
	Ours (AG)	98.4	92.8	81.2	69.8	61.9	38.2	69.3
	Ours (AOG)	**98.8**	92.7	**83.9**	**74.4**	64.0	41.1	71.8
	Ours (AOG+BG)	98.3	**92.7**	83.7	73.1	**66.0**	**41.4**	**71.9**
	Ours (AOG single-avg)	89.0	77.6	76.1	68.4	57.8	37.9	64.7
	Ours (AOG single-both)	86.1	71.8	70.9	64.8	49.6	32.3	59.5
	Ours (AOG match-both)	97.0	85.0	78.1	70.1	55.0	35.8	66.0
	Ours (AOG+BG)†	98.3	92.7	86.8	78.2	70.2	45.1	75.2

Table 5.2: Benchmark evaluation results on the Leeds dataset. We evaluate our baseline model (AG), grammar model (AOG), and grammar model with background features (AOG+BG) against recent published results on this dataset. The † symbol indicates the use of a modified evaluation protocol, see text for details.

Figure 5.7: Selected parsing results for PARSE and Leeds. Bounding boxes are drawn for each of the 10 parts considered for evaluation. Parts localized correctly are shown in red, and incorrectly in blue. The top two rows are examples of challenging poses with perfect localization scores from the PARSE and Leeds datasets respectively. The bottom row illustrates some of the failure modes on both datasets. Common failures are due to double counting, occlusion, and background confounders, which is compounded by extreme perspective, foreshortening, and crumpled poses.

CHAPTER 6

Conclusions

6.1 Summary

In this work we develop a framework for constructing and learning stochastic image grammar models for articulated objects. These models can compactly represent many of the complex geometric deformations and appearance variabilities seen in the human body using a semantically meaningful hierarchy. We propose two types of grammars, namely phrase-structured grammars and dependency grammars, that can decompose the body into sub-models while still preserving the articulated kinematics. The phrase-structure grammar represents the body as a coarse-to-fine decomposition, whereas the dependency grammar represents the body as a dependency tree of simple parts. We also provide two approaches to learning a probability distribution on the parses of these grammars, from a generative perspective using maximizing likelihood, and a discriminative perspective using empirical risk minimization. The central unit to both of these grammar models is the production, which represents a part of the body whether it be a primitive or composite part. The geometries of these parts are represented as an oriented bounding box, parameterized by its location, orientation, scale, and aspect ratio. Each production specifies an appearance model that describes the image structure within the part geometry. The productions also define the relations that control the geometry of its parts relative to each other, as well as

which productions can co-occur together. Both primitive and composite parts can appear in dramatically different modes due to clothing, viewpoint, or many other factors. The grammar productions are specifically designed to capture and represent these modes, which have their own distinct appearances and part geometries associated with them. We present several approximate and optimal parsing algorithms, and demonstrate superior pose estimation performance on several challenging datasets. Furthermore, we believe this framework can be taken beyond pose estimation, such as for modeling semantically meaningful attributes of the parts to infer object properties such as gender, or similarly encode local geometric cues into the grammar to infer human activity.

6.2 Contributions

We summarize our contributions to part-based modeling and human pose estimation from the following aspects:

1. The contextual hierarchy of the stochastic grammar model represents articulated pose, as well as semantically meaningful compositional information such as the type of clothing people are wearing, or what orientation they are facing.

2. We present learning strategies for a context-sensitive stochastic image grammar model from a generative MLE perspective, as well as a discriminative perspective using a structured-output SVM.

3. The dynamic programming algorithm we present can compute the globally optimal parse in under a second on modern computing hardware for the dependency grammar case.

4. We experimentally justify the use of multiple productions in the form of OR-nodes to represent and detect parts that are highly variant in either appearance or geometry by showing performance gains from detection experiments using appearance models of each part individually, as well as full parsing experiments.

5. Our human body grammar model significantly outperforms all known methods for human pose estimation on multiple modern and highly competitive benchmarks.

References

[AKP08] M. Aycinena, L. P. Kaelbling, and T. Lozano Perez. "Learning Grammatical Models for Object Recognition." In *Technical Report, AI Lab, Massachusetts Institute of Technology*, 2008.

[ARS09] M. Andriluka, S. Roth, and B. Schiele. "Pictorial structures revisited: People detection and articulated pose estimation." In *CVPR*, pp. 1014–1021, 2009.

[AT07] Yali Amit and Alain Trouvé. "Pop: Patchwork of parts models for object recognition." *International Journal of Computer Vision*, **75**(2):267–282, 2007.

[Bar03] Moshe Bar. "A cortical mechanism for triggering top-down facilitation in visual object recognition." *Journal of Cognitive Neuroscience*, **15**(4):600–609, 2003.

[BGP97] Elie Bienenstock, Stuart Geman, and Daniel Potter. "Compositionality, MDL priors, and object recognition." *Advances in neural information processing systems*, pp. 838–844, 1997.

[BKS10] M. Bergtholdt, J. Kappes, S. Schmidt, and C. Schnörr. "A Study of Parts-Based Object Class Detection Using Complete Graphs." *Int. J. Comput. Vision*, **87**:93–117, March 2010.

[BM09] L. Bourdev and J. Malik. "Poselets: Body Part Detectors Trained Using 3D Human Pose Annotations." In *International Conference on Computer Vision*, sep 2009.

[BZ05] A. Barbu and S. C. Zhu. "Generalizing Swendsen-Wang to Sampling Arbitrary Posterior Probabilities." *IEEE Trans. Pattern Analysis and Machine Intelligence*, **27**(8):1239–1253, aug 2005.

[CG98] Zhiyi Chi and Stuart Geman. "Estimation of probabilistic context-free grammars." *Computational Linguistics*, **24**(2):299–305, 1998.

[CGJ98] E. Charniak, S. Goldwater, and M. Johnson. "Edge-Based Best-First Chart Parsing." In *National Conference on Artificial Intelligence*, 1998.

[CJ05] Eugene Charniak and Mark Johnson. "Coarse-to-fine n-best parsing and MaxEnt discriminative reranking." In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pp. 173–180. Association for Computational Linguistics, 2005.

[CK00] Michael Collins and Terry Koo. "Discriminative reranking for natural language parsing." In *Proceedings of the Seventeenth International Conference on Machine Learning (ICML 2000)*, pp. 175–182. Citeseer, 2000.

[CXL06] H. Chen, Z. J. Xu, Z. Q. A. Liu, and S. C. Zhu. "Composite Templates for Cloth Modeling and Sketching." In *CVPR*, pp. I: 943–950, 2006.

[CZL07] Y. Chen, L. Zhu, C. Lin, A. L. Yuille, and H. J. Zhang. "Rapid Inference on a Novel AND/OR graph for Object Detection, Segmentation and Parsing." In John C. Platt, Daphne Koller, Yoram Singer, and Sam T. Roweis, editors, *NIPS*. MIT Press, 2007.

[DDS09] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. "ImageNet: A Large-Scale Hierarchical Image Database." In *CVPR09*, 2009.

[DR12] Chaitanya Desai and Deva Ramanan. "Detecting Actions, Poses, and Objects with Relational Phraselets." In *ECCV (4)*, pp. 158–172, 2012.

[DT05] N. Dalal and B. Triggs. "Histograms of Oriented Gradients for Human Detection." In *CVPR*, pp. I: 886–893, 2005.

[EF09] Marcin Eichner and Vittorio Ferrari. "Better Appearance Models for Pictorial Structures." In *BMVC*, 2009.

[EGW08] M. Everingham, L. J. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. *The PASCAL Visual Object Classes Challenge 2008 (VOC2008) Results*. Online, 2008.

[FCH08] R. Fan, K. Chang, C. Hsieh, X. Wang, and C. Lin. "LIBLINEAR: A Library for Large Linear Classification." *Journal of Machine Learning Research*, **9**:1871–1874, 2008.

[FE73] M. A. Fischler and R. A. Elschlager. "The Representation and Matching of Pictorial Structures." *IEEE Trans. Computer*, **22**(1):67–92, jan 1973.

[FH04] P. F. Felzenszwalb and D. P. Huttenlocher. "Distance Transforms of Sampled Functions." In *Cornell University*, 2004.

[FH05] P. F. Felzenszwalb and D. P. Huttenlocher. "Pictorial Structures for Object Recognition." *International Journal of Computer Vision*, **61**(1):55–79, January 2005.

[FH06] V. Franc and V. Hlavac. "A Novel Algorithm for Learning Support Vector Machines with Structured Output Spaces." *Research Report K333 22/06, CTU-CMP-2006-04*, 2006.

[FJZ08] V. Ferrari, M. Marin Jimenez, and A. Zisserman. "Progressive search space reduction for human pose estimation." In *CVPR*, pp. 1–8, 2008.

[FM10] P. F. Felzenszwalb and D. McAllester. "Object Detection Grammars." *Technical Report TR-2010-02, Dept. of Computer Science, University of Chicago*, 2010.

[FMR08] Pedro Felzenszwalb, David McAllester, and Deva Ramanan. "A discriminatively trained, multiscale, deformable part model." In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pp. 1–8. IEEE, 2008.

[FPZ03] R. Fergus, P. Perona, and A. Zisserman. "Object class recognition by unsupervised scale-invariant learning." In *CVPR*, pp. II: 264–271, 2003.

[Fri87] Jerome H Friedman. "Exploratory projection pursuit." *Journal of the American statistical association*, **82**(397):249–266, 1987.

[Fu86] K. S. Fu. "A Step Towards Unification of Syntactic and Statistical Pattern Recognition." *IEEE Trans. Pattern Anal. Mach. Intell.*, **8**(3):398–404, 1986.

[GFM11] Ross B. Girshick, Pedro F. Felzenszwalb, and David McAllester. "Object Detection with Grammar Models." In *NIPS*, 2011.

[HZ05] Feng Han and Song-Chun Zhu. "Bottom-up/top-down image parsing by attribute graph grammar." In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 2, pp. 1778–1785. IEEE, 2005.

[IF01] S. Ioffe and D. A. Forsyth. "Probabilistic Methods for Finding People." *International Journal of Computer Vision*, **43**(1):45–68, jun 2001.

[JE09] Sam Johnson and Mark Everingham. "Combining Discriminative Appearance and Segmentation Cues for Articulated Human Pose Estimation." In *ICCV Workshop on Machine Learning for Vision-based Motion Analysis*, 2009.

[JE10] Sam Johnson and Mark Everingham. "Clustered Pose and Nonlinear Appearance Models for Human Pose Estimation." In *BMVC*, pp. 1–11, 2010.

[JE11] Sam Johnson and Mark Everingham. "Learning effective human pose estimation from inaccurate annotation." In *CVPR*, pp. 1465–1472, 2011.

[JG06] Y. Jin and S. Geman. "Context and Hierarchy in a Probabilistic Image Model." In *CVPR*, pp. II: 2145–2152, 2006.

[JM08] H. Jiang and D. R. Martin. "Global pose estimation using non-tree models." In *CVPR*, pp. 1–8, 2008.

[LC04] M. W. Lee and I. Cohen. "Proposal maps driven MCMC for estimating human body pose in static images." In *CVPR*, pp. II: 334–341, 2004.

[Mar82] David Marr. *Vision: A computational investigation into the human representation and processing of visual information.* W.H. Freeman publ., 1982.

[MG01] David Mumford and Basilis Gidas. "Stochastic models for generic images." *Quarterly of applied mathematics*, **59**(1):85–112, 2001.

[MPP01] A. Mohan, C. P. Papageorgiou, and T. Poggio. "Example-Based Object Detection in Images by Components." *IEEE Trans. Pattern Analysis and Machine Intelligence*, **23**(4):349–361, April 2001.

[MRE04] G. Mori, X. F. Ren, A. A. Efros, and J. Malik. "Recovering Human Body Configurations: Combining Segmentation and Recognition." In *CVPR*, pp. II: 326–333, 2004.

[MZ93] Stephane G Mallat and Zhifeng Zhang. "Matching pursuits with time-frequency dictionaries." *Signal Processing, IEEE Transactions on*, **41**(12):3397–3415, 1993.

[OKS78] Yu-ichi Ohta, Takeo Kanade, and Toshiyuki Sakai. "An analysis system for scenes containing objects with substructures." In *Proceedings of the Fourth International Joint Conference on Pattern Recognitions*, pp. 752–754, 1978.

[Pea84] Judea Pearl. "Heuristics: intelligent search strategies for computer problem solving." 1984.

[PZ10] J. Porway and S. C. Zhu. "C4: Exploring Multiple Solutions In Graphical Models by MCMC." *PAMI*, 2010.

[Ram06] Deva Ramanan. "Learning to parse images of articulated bodies." In *NIPS*, pp. 1129–1136, 2006.

[RBM05] X. F. Ren, A. C. Berg, and J. Malik. "Recovering Human Body Configurations Using Pairwise Constraints between Parts." In *ICCV*, pp. I: 824–831, 2005.

[SB06] L. Sigal and M. J. Black. "Measure Locally, Reason Globally: Occlusion-sensitive Articulated Pose Estimation." In *CVPR*, pp. II: 2041–2048, 2006.

[SIS03] L. Sigal, M. Isard, B. H. Sigelman, and M. J. Black. "Attractive people: Assembling loose-limbed models using non-parametric belief propagation." *NIPS*, 2003.

[SJT10] B. Sapp, C. Jordan, and B. Taskar. "Adaptive pose priors for pictorial structures." In *CVPR*, pp. 422–429. IEEE, 2010.

[SNH10] Vivek Kumar Singh, Ram Nevatia, and Chang Huang. "Efficient inference with multiple heterogeneous part detectors for human pose estimation." In *Computer Vision–ECCV 2010*, pp. 314–327. Springer, 2010.

[SS07] P. Srinivasan and J. B. Shi. "Bottom-Up Recognition and Parsing of the Human Body." In *CVPR*, pp. 1–8, 2007.

[SS11] Min Sun and Silvio Savarese. "Articulated part-based model for joint object detection and pose estimation." In *ICCV*, pp. 723–730, 2011.

[SZ12] Zhangzhang Si and Song-Chun Zhu. "Learning Hybrid Image Templates (HIT) by Information Projection." *IEEE Trans. Pattern Anal. Mach. Intell.*, **34**(7):1354–1367, 2012.

[TCY05] Z. W. Tu, X. R. Chen, A. L. Yuille, and S. C. Zhu. "Image Parsing: Unifying Segmentation, Detection, and Recognition." *International Journal of Computer Vision*, **63**(2):113–140, jul 2005.

[TF10] Duan Tran and David Forsyth. "Improved Human Parsing with a Full Relational Model." In *ECCV (4)*, pp. 227–240, 2010.

[TGK03] Benjamin Taskar, Carlos Guestrin, and Daphne Koller. "Max-Margin Markov Networks." In *NIPS*, 2003.

[THJ04] Ioannis Tsochantaridis, Thomas Hofmann, Thorsten Joachims, and Yasemin Altun. "Support vector machine learning for interdependent and structured output spaces." In *ICML*, 2004.

[TZ02] Zhuowen Tu and Song Chun Zhu. "Image Segmentation by Data-Driven Markov Chain Monte Carlo." *IEEE Trans. Pattern Anal. Mach. Intell.*, **24**(5):657–673, 2002.

[TZN12] Yuandong Tian, C. Lawrence Zitnick, and Srinivasa G. Narasimhan. "Exploring the Spatial Hierarchy of Mixture Models for Human Pose Estimation." In *ECCV (5)*, pp. 256–269, 2012.

[VJS05] P. Viola, M. J. Jones, and D. Snow. "Detecting Pedestrians Using Patterns of Motion and Appearance." *International Journal of Computer Vision*, **63**(2):153–161, jul 2005.

[WSF07] Y. N. Wu, Z. Z. Si, C. Fleming, and S. C. Zhu. "Deformable Template As Active Basis." In *ICCV*, pp. 1–8, 2007.

[WZ11] Tianfu Wu and Song-Chun Zhu. "A numerical study of the bottom-up and top-down inference processes in and-or graphs." *International journal of computer vision*, **93**(2):226–252, 2011.

[YR11] Y. Yang and D. Ramanan. "Articulated Pose Estimation with Flexible Mixtures-of-Parts." In *CVPR*, 2011.

[ZCL08] L. Zhu, Y. H. Chen, Y. Lu, C. X. Lin, and A. L. Yuille. "Max Margin AND/OR Graph learning for parsing the human body." In *CVPR*, pp. 1–8, 2008.

[Zha10] R. Zhang. "Adjusting Active Basis Model by Regularized Logistic Regression." http://www.stat.ucla.edu/~ywu/AB/adjust.html, 2010.

[ZLL06] J. Y. Zhang, Y. X. Liu, J. B. Luo, and R. T. Collins. "Body Localization in Still Images Using Hierarchical Models and Hybrid Search." In *CVPR*, pp. II: 1536–1543, 2006.

[ZM06] Song Chun Zhu and David Mumford. "A Stochastic Grammar of Images." *Foundations and Trends in Computer Graphics and Vision*, **2**(4):259–362, 2006.

[ZWM98] S. C. Zhu, Y. N. Wu, and D. Mumford. "Filters, Random-Fields and Maximum-Entropy (Frame): Towards a Unified Theory for Texture Modeling." *International Journal of Computer Vision*, **27**(2):107–126, mar 1998.