STRUCTURE PREDICTION FOR HUMAN PARSING

BY

DUAN DUC TRAN

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Computer Science
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2011

Urbana, Illinois

Doctoral Committee:

Professor David Forsyth, Chair & Director of Research
Professor Narendra Ahuja
Professor Derek Hoiem
Professor Deva Ramanan, University of California, Irvine

ABSTRACT

This thesis shows that structure prediction is well-suited for detecting and parsing people in images (and videos) due to the advantage of learning local part appearance models jointly with relationships between body parts. In detecting people, this method can deal with hard cases, for example, a person mounting a bicycle, that are uncommon in the training data and can cause current person detectors to fail. This thesis demonstrates a pedestrian finder which first finds the most likely human pose in the window using a discriminative procedure trained with structure learning on a small dataset, then presents features based on that configuration to an SVM classifier. This thesis shows, using the INRIA Person dataset, that estimates of configuration significantly improve the accuracy of a discriminative pedestrian finder.

This thesis shows quantitative evidence that a full relational model of the body performs better at upper body parsing than the standard tree model, despite the need to adopt approximate inference and learning procedures. The method uses an approximate search for inference, and an approximate structure learning method to learn. This thesis compares this method to state of the art methods on a dataset prepared at UIUC (which depicts a wide range of poses), on the standard Buffy dataset, and on the reduced PASCAL dataset published recently. Results suggest that the Buffy dataset over emphasizes poses where the arms hang down, and that leads to generalization problems.

Despite the superior performance of a full relational model to a tree structure model, its practical use is still limited because it must deal with the high complexity in inference. This thesis shows a method to boost a parser with poselet pruners. The method first develops a cascade of hierarchical poselet pruners to prune the search space to a small set of part states and then builds a hierarchical poselet parser to find part locations on the pruned set. Experiments on the UIUC Sport dataset shows that the poselet pruners can effectively prune away more than 99.6% of unlikely part states to about 500 states per part. This small set of part states allows the use of advanced appearance models for better parsers. The method achieves performance comparable to state-of-the-art methods' while improves the speed of finding part locations several times.

To my parents, Luong Tran and Hue Nguyen

my dear wife, Thanh-Nhan Nguyen

and my beloved daughter, Han Tran-Nguyen.

ACKNOWLEDGMENTS

I want first to thank my advisor, David Forsyth for his invaluable guidance, support and encouragement during my PhD life. It is really fortunate to be his student. His deep and wide knowledge, intelligence made every discussion inspirational and interesting.

I also thank Prof. Derek Hoiem, Prof. Deva Ramanan, Prof. Nadrendra Ahuja for serving on my committee and for providing the insightful comments on the thesis.

I thank the UIUC Vision Group: Ryan White, Nicloas Loeff, Alex Sorokin, Du Tran, Ali Farhadi, Gang Wang, Selen Pehlivan, Ian Endres, Varsha Hedau, Zicheng Liao, Yang Wang, Amin Sadeghi for many great and fun chats that made my research life at UIUC memorable. Thanks to Yang Wang for a great collaboration and providing many helps.

I thank to the Exploratory Computer Vision Group at IBM Research, Hawthorne, NY where I had a wonderful time during the internship. It was a pleasure to work with Dr. Sharat Pankanti, Dr. Rogerio Feris and friends I met there: Daniel Vaquero, Haowei Liu, Hyunggon Park.

I thank all friends I have met and worked with during my PhD time: Du Tran, Quang Do, Gang Wang, Yang Wang, Vivek Srikuma, Hoang Nguyen. Thanks to the Vietnamese friends at UIUC: Kien Nguyen, Loc Bui, Loan Vo, Nghia Nguyen's family, Nam Pham's family, Huy Bui's family and many others who have made my life at UIUC more cheerful. And lastly, my heartful thanks to my family for being always beside me.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Detecting and parsing people are problems of very interest of vision community recently. In detecting people, we want to detect and localize areas, bounding boxes, containing people in an image (or videos) or, say, there are no persons in it. In human parsing, not only we want to localize a region of interested (ROI) containing a person, but also we must produce an accurate representation of the body configuration. These accurate human body parts localization greatly benefits human activity recognition; for example, the ROI might be produced by a detector, but we must know what the arms are doing to label the activity.

For detecting people, template matching is one of the most successful techniques, in which each hypothesis scanning window is tested if it has a person or not. Different types of machineries and features were studied such as a Support Vector Machine (SVM) applied to wavelet expansion ([1] and [2]); a neural network applied to stereoscopic reconstructions [3]; chamfer matching to a hierachy of contour templates [4] and [5]; a likelihood threshold applied to a random field model [6]; an SVM applied to spatial wavelets stacked over four frames to give dynamical cues [2]; a cascade architecture applied to spatial averages of temporal differences [7]; and a very successful HOG features by [8]. However, these methods process features in an unstructrured manner which do not take into account semantic body parts and their relations. They tend to work on common cases (appear most likely in

the dataset) and fail on uncommon cases (e.g. persons mounting on bikes). We conjecture that parsing helps detection where reasoning about a person layout of parts (even though it is not necessarily perfectly accurate) enriches the signal near body parts and improves the accuracy of detecting people. We do it by a two-step strategy: first, for each window, we estimate the configuration of the best person available in that window by a structure prediction method; we then extract features for that window conditioned on the configuration estimate, and pass these features to a support vector machine classifier to make the final decision on the window.

Now come to the demand of accurate human parsing, we must correctly localize body segments. The representation produced is usually a stick figure, or a box model, but may be image regions or joint locations. All representations encode the configuration of body segments. It is usual to represent pairwise spatial relations between locations structured into a kinematic tree, so that dynamic programming can be used for inference [9, 10]. The joint relations encoded by the kinematic tree model are important, but there are other important relations. Limbs on the left side of the body usually look like those on the right. This cue should be important, because limbs are genuinely difficult to detect, particularly in the absence of an appearance model. Inference difficulties occur when one encodes relations between all pairs of segments, because finding the best parse now becomes max-cut. Approximate inference on sets of extended image segments can produce good parses for difficult images [11]. However, there is no evidence comparing the benefits of a full model against the cost of approximate inference. In this thesis, we explore the advantages of representing a full set of relations for human parsing. We apply structure prediction to jointly learn local appearance and pairwise weights. We show strong quantitative evidence that the advan-

tages of representing a full set of relations between segments outweigh the costs of approximate inference and approximate learning. We demonstrate the method on upper body parsing, and show results on Buffy and Pascal dataset [12], and on a new dataset where the prior on body configuration is quite weak.

Being cast as a structured output problem, human parsing presents a difficulty to find the best structures because the search spaces is huge. In a typical example, there are about $10,000$ states per part (a full body is often represented by 10 parts as shown in Figure 4.2) which results in 10^8 evaluations of a pairwise part relation. This issue also prevents the use of sophisticated appearance models (which are known to be very useful for better performance [13]) because evaluating pairwise appearance relations between related parts leads to intractable searches as visualized in Figure 4.3. For this reason, people tend to choose models that leads to simple searches, for example the tree-based models as in pictorial structures [14]) which take advantages of fast inference by dynamic programing. Complex full relational models are carefully designed with approximate inference [15] to make them tractable.

To promote practical uses of full relational models, we must make inference faster. One approach is to develop good strategies to prune down search spaces. A good pruner eliminates part states that are unlikely to be in the correct structures leaving as few remaining states as possible without wrongly removing the correct states. Some recent work has proposed very good pruning strategies. Mori *et al.* [11] used over-segmentation to generate limb hypotheses. Ferrari *et al.* [12] used upper-body detectors and foreground/background segmentations to narrow down the possible areas of limbs around the body candidates. Felzenszwalb *et al.* [16] proposed a cas-

cade model by using early stopping. Tran and Forsyth [15] pruned the search space using local searches. Recently, Sapp *et al.* [17] developed a cascade of pictorial structure-based pruners to progressively prune states from low resolution to high resolution. In this thesis, we design a series of poselet pruners in a cascaded fashion of three different types of pruner from simple single part pruners to complex tree-based pruners. Our pruners work on the actual state space resolution. First, we use simple and very fast pruners to quickly prune away more than half of the part states in early stages. Then more accurate yet more complex pruners are used to get a small number of remaining states. We show that our pruners can prune down up to 99% of part states while retaining about 90% of correct parts (at $PCP_{0.5}$ [1]) in the remaining search spaces. This set of states is small enough to allow approximate inference of full relational models with complex appearance models to parse highly accurate body parts.

The structure of the thesis is organized as follows: Chapter 1 presents an introduction of the problems of detecting and parsing people and pruning the search space that we attempt to solve. We then present in detail the technical and experiments we have done for each problem in the three following chapters. In particular, chapter 2 presents the problem of detecting pedestrian where we show parsing helps to improve detection. Chapter 3 presents the problem of parsing human parts using full relational models. Chapter 4 presents a method of building a cascade of poselet pruners to improve parsers in both inference time and performance. Chapter 5 is about the conclusions and the future directions.

[1] PCP - Percentage of Correctly estimated Parts: a criteria to evaluate part estimates (see details in Section 4.8)

CHAPTER 2

DETECTING PEDESTRIAN WITH STRUCTURE PREDICTION

This chapter presents a method to build a pedestrian detector using parsing. Fair discriminative pedestrian finders are now available. In fact, these pedestrian finders make most errors on pedestrians in configurations that are uncommon in the training data, for example, mounting a bicycle. This is undesirable. However, the human configuration can itself be estimated discriminatively using structure learning. This chapter demonstrates that parsing helps improve a pedestrian finder. The method first finds the most likely human pose in the window using a discriminative procedure trained with structure learning on a small dataset. The method then presents features based on that configuration to an SVM classifier. Our results suggest, using the INRIA Person dataset, that estimates of configuration significantly improve the accuracy of a discriminative pedestrian finder.

2.1 Introduction

Very accurate pedestrian detectors are an important technical goal; approximately half-a-million pedestrians are killed by cars each year (1997 figures, in [18]). At relatively low resolution, pedestrians tend to have a characteristic appearance. Generally, one must cope with lateral or frontal views of a walk. In these cases, one will see either a "lollipop" shape — the torso is wider than the legs, which are together in the stance phase of the walk — or a "scissor" shape — where the legs are swinging in the walk. This encourages the use of template matching. Early template matchers include: support vector machines applied to a wavelet expansion ([1], and variants described in [2]); a neural network applied to stereoscopic reconstructions [3]; chamfer matching to a hierachy of contour templates [4]; a likelihood threshold applied to a random field model [6]; an SVM applied to spatial wavelets stacked over four frames to give dynamical cues [2]; a cascade architecture applied to spatial averages of temporal differences [7]; and a temporal version of chamfer matching to a hierachy of contour templates [5].

One of the most successful static template matcher is due to Dalal and Triggs [8]. Their method is based on a comprehensive study of features and their effects on performance for the pedestrian detection problem. The method that performs best involves a histogram of oriented gradient responses (a HOG descriptor). This is a variant of Lowe's SIFT feature [19]. Each window is decomposed into overlapping blocks (large spatial domains) of cells (smaller spatial domains).In each block, a histogram of gradient directions (or edge orientations) is computed for each cell with a measure of histogram "energy". These cell histograms are concatenated into block histograms followed by normalization which obtains a modicum of illumination

invariance. The detection window is tiled with an overlapping grid. Within each block HOG descriptors are computed, and the resulting feature vector is presented to an SVM. Dalal and Triggs show this method produces no errors on the 709 image MIT dataset of [1]; they describe an expanded dataset of 1805 images. Furthermore, they compare HOG descriptors with the original method of Papageorgiou and Poggio [1]; with an extended version of the Haar wavelets of Mohan *et al.* [20]; with the PCA-Sift of Ke and Sukthankar ([21]; see also [22]); and with the shape contexts of Belongie *et al.* [23]. The HOG descriptors outperform all other methods.

A key difficulty with pedestrian detection is that detectors must work on human configurations not often seen in datasets. For systems to be useful, they cannot fail even on configurations that are very uncommon — it is not acceptable to run people over when they stand on their hands. There is some evidence (figure 2.1) that less common configurations present real difficulties for very good current pedestrian detectors (our reimplementation of Dalal and Triggs' work [8]).

Figure 2.1. *Configuration estimates result in our method producing fewer false negatives than our implementation of Dalal and Triggs does. The figure shows typical images which are incorrectly classified by our implementation of Dalal and Triggs, but correctly classified when a configuration estimate is attached. We conjecture that a configuration estimate can avoid problems with occlusion or contrast failure because the configuration estimate reduces noise and the detector can use lower detection thresholds.*

2.2 Configuration and Parts

Detecting pedestrians with templates most likely works because pedestrians appear in a relatively limited range of configurations and views (e.g. "Our HOG detectors cue mainly on silhouette contours (especially the head, shoulders and feet)" [8], p.893). It appears certain that using the architecture of constructing features for whole image windows and then throwing the result into a classifier could be used to build a person-finder for arbitrary configurations and arbitrary views only with a major engineering effort. The set of examples required would be spectacularly large, for example. This is unattractive, because this set of examples implicitly encodes a set of facts that are relatively easy to make explicit. In particular, people are made of body segments which individually have a quite simple structure, and these segments are connected into a kinematic structure which is quite well understood.

All this suggests finding people by finding the parts and then reasoning about their layout — essentially, building templates with complex internal kinematics. The core idea is very old (see the review in [24]) but the details are hard to get right and important novel formulations are a regular feature of the current research literature.

Simply identifying the body parts can be hard. **Discriminative approaches** use classifiers to detect parts, then reason about configuration [20]. **Generative approaches** compare predictions of part appearance with the image; one can use a tree structured configuration model [14], or an arbitrary graph [25]. If one has a video sequence, part appearance can itself be learned [26, 27]; more recently, Ramanan has shown knowledge of articulation properties gives an appearance model in a single image [28]. **Mixed**

approaches use a discriminative model to identify parts, then a generative model to construct and evaluate assemblies [29, 30, 31]. **Codebook approaches** avoid explicitly modelling body segments, and instead use unsupervised methods to find part decompositions that are good for recognition (rather than disarticulation) [32].

Our pedestrian detection strategy consists of two steps: first, for each window, we estimate the configuration of the best person available in that window; second, we extract features for that window conditioned on the configuration estimate, and pass these features to a support vector machine classifier, which makes the final decision on the window.

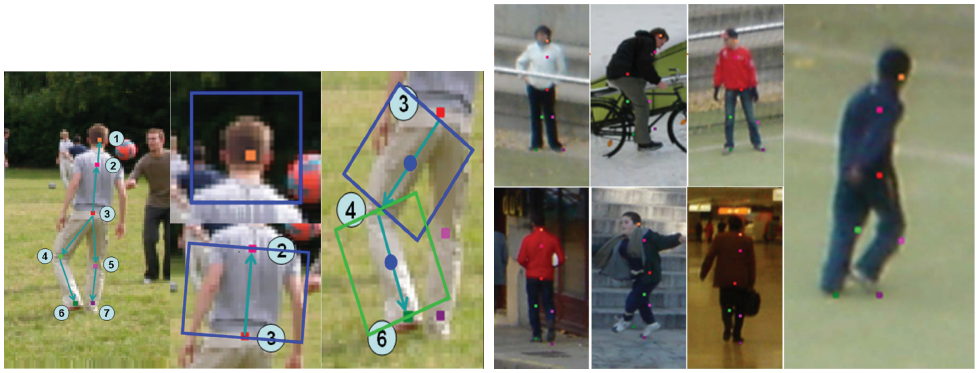

Figure 2.2. This figure is best viewed in color. *Our model of human layout is parametrized by seven vertices, shown on an example on the* **far left**. *The root is at the hip; the arrows give the direction of conditional dependence. Given a set of features, the extremal model can be identified by dynamic programming on point locations. We compute segment features by placing a box around some vertices (as in the* **head**), *or pairs of vertices (as in the* **torso** *and* **leg**). *Histogram features are then computed for base points referred to the box coordinate frame; the histogram is shifted by the orientation of the box axis (section 3.2.4) within the rectified box. On the* **far right**, *a window showing the color key for our structure learning points; dark green is a foot, green a knee, dark purple the other foot, purple the other knee, etc. Note that structure learning is capable of finding distinction of left legs (green points) and right legs (pink points). On the* **center right**, *examples of configurations estimated by our configuration estimator after 20 rounds of structure learning to estimate* $\mathbf{W}$.

2.3 Configuration Estimation and Structure Learning

We are presented with a window within which may lie a pedestrian. We would like to be able to estimate the most likely configuration for any pedestrian present. Our research hypothesis is that this estimate will improve pedestrian detector perfomance by reducing the amount of noise the final detector must cope with — essentially, the segmentation of the pedestrian is improved from a window to a (rectified) figure. We follow convention (established by [10]) and model the configuration of a person as a tree model of segments (figure 2.2), with a score of segment quality and a score of segment-segment configuration. We ignore arms because they are small and difficult to localize. Our configuration estimation procedure will use dynamic programming to extract the best configuration estimate from a set of scores depending on the location of vertices on the body model.

However, we do not know which features are most effective at estimating segment location; this is a well established difficulty in the literature [24]. **Structure learning** is a method that uses a series of correct examples to estimate appropriate weightings of features relative to one another to produce a score that is effective at estimating configuration [33, 34]. We will write the image as $\mathcal{I}$; coordinates in the image as $\mathbf{x}$; the coordinates of an estimated configuration as $\mathbf{y}$ (which is a stack of 7 point coordinates); the score for this configuration as $\mathbf{W}^T\mathbf{f}(\mathcal{I}, \mathbf{x}; \mathbf{y})$ (which is a linear combination of a collection of scores, each of which depends on the configuration and the image).

For a given image $\mathcal{I}_0$ and known $\mathbf{W}$ and $\mathbf{f}$, the best configuration estimate is

$$\arg\max_{\mathbf{y} \in \mathbf{y}(\mathcal{I}_0)} \mathbf{W}^T\mathbf{f}(\mathcal{I}_0, \mathbf{x}; \mathbf{y})$$

and this can be found with dynamic programming for appropriate choice of

$\mathbf{f}$ and $\mathbf{y}(\mathcal{I}_0)$. There is a variety of sensible choices of features for identifying body segments, but there is little evidence that a particular choice of features is best; different choices of $\mathbf{W}$ may lead to quite different behaviours. In particular, we will collect a wide range of features likely to identify segments well in $\mathbf{f}$, and wish to learn a choice of $\mathbf{W}$ that will give good configuration estimates.

We choose a loss function $L(\mathbf{y}_t, \mathbf{y}_p)$ that gives the cost of predicting $\mathbf{y}_p$ when the correct answer is $\mathbf{y}_t$. Write the set of n examples as $\mathcal{E}$, and $\mathbf{y}_{p,i}$ as the prediction for the i'th example. Structure learning must now estimate a $\mathbf{W}$ to minimize the hinge loss as in [35]

$$\frac{1}{2} \parallel \mathbf{W} \parallel^2 + \frac{1}{n} \sum_{i \in examples} \beta_i \xi_i$$

subject to the constraints

$$\forall i \in \mathcal{E},\ \mathbf{W}^T \mathbf{f}(\mathcal{I}_i, \mathbf{x}; \mathbf{y}_{t,i}) + \xi_i \geq \max_{\mathbf{y}_{p,i} \in \mathbf{y}(\mathcal{I}_i)} (\mathbf{W}^T (\mathcal{I}_i, \mathbf{x}; \mathbf{y}_{p,i}) + L(\mathbf{y}_{t,i}, \mathbf{y}_{p,i}))$$

At the minimum, the slack variables ξ_i happen at the equality of the constraints. Therefore, we can move the constraints to the objective function, which is:

$$\frac{1}{2} \parallel \mathbf{W} \parallel^2 + \frac{1}{n} \sum_{i \in examples} \beta_i (\max_{\mathbf{y}_{p,i} \in \mathbf{y}(\mathcal{I}_i)} (\mathbf{W}^T (\mathcal{I}_i, \mathbf{x}; \mathbf{y}_{p,i}) + L(\mathbf{y}_{t,i}, \mathbf{y}_{p,i})) - \mathbf{W}^T \mathbf{f}(\mathcal{I}_i, \mathbf{x}; \mathbf{y}_{t,i}))$$

Notice that this function is convex, but not differentiable. We follow Ratliff *et al.* [35], and use the subgradient method (see [36]) to minimize. In this case, the derivative of the cost function at an extremal $\mathbf{y}_{p,i}$ is a subgradient (but not a gradient, because the cost function is not differentiable everywhere).

2.4 Features

There are two sets of features: first, those used for estimating configuration of a person from a window; and second, those used to determine whether a person is present conditioned on the best estimate of configuration.

2.4.1 Features for Estimating Configuration

We use a tree structured model, given in figure 2.2. The tree is given by the position of seven points, and encodes the head, torso and legs; arms are excluded because they are small and difficult to identify, and pedestrians can be identified without localizing arms. The tree is rooted at hips, and the arrows give the direction of conditional dependence. We assume that $torso, leftleg, rightleg$ are conditionally independent given the root (at the hip).

The feature vector $\mathbf{f}(\mathcal{I}, \mathbf{x}; \mathbf{y})$ contains two types of feature: appearance features encode the appearance of putative segments; and geometric features encode relative and absolute configuration of the body segments.

Each **geometric feature** depends on at most three point positions. We use three types of feature. First, the length of a segment, represented as a 15-dimensional binary vector whose elements encode whether the segment is longer than each of a set of test segments. Second, the cosine of the angle between a segment and the vertical axis. Third, the cosine of the angle between pairs of adjoining segments (except at the lower torso, for complexity reasons); this allows the structure learning method to prefer straight backs, and reasonable knees.

Appearance features are computed for rectangles constructed from pairs of points adjacent in the tree. For each rectangle, we compute His-

togram of Oriented Gradient (HOG) features, after [8]. These features have a strong record in pedestrian detection, because they can detect the patterns of orientation associated with characteristic segment outlines (typically, strong vertical orientations in the frame of the segment for torso and legs; strong horizontal orientations at the shoulders and head). However, histograms involve spatial pooling; this means that one can have many strong vertical orientations that do not join up to form a segment boundary. This effect means that HOG features alone are not particularly effective at estimating configuration.

To counter this effect, we use the local gradient features described by Ke and Sukthankar [21]. To form these features, we concatenate the horizontal and vertical gradients of the patches in the segment coordinate frame, then normalize and apply PCA to reduce the number of dimensions. Since we want to model the appearance, we do not align the orientation to a canonical orientation as in PCA-SIFT. This feature reveals whether the pattern of a body part appears at that location. The PCA space for each body part is constructed from 500 annotated positive examples.

2.4.2 Features for Detection

Once the best configuration has been obtained for a window, we must determine whether a person is present or not. We do this with a support vector machine. Generally, the features that determine configuration should also be good for determining whether a person is present or not. However, a set of HOG features for the whole image window has been shown to be good at pedestrian detection [8]. The support vector machine should be able to distinguish between good and bad features, so it is natural to concatenate

the configuration features described above with a set of HOG features. We find it helpful to reduce the dimension of the set of HOG features to 500, using principal components. We find that these whole window features help recover from incorrect structure predictions. These combined features are used in training the SVM classifier and in detection as well.

2.5 Results

Dataset: We use INRIA Person, consisting of 2416 pedestrian images (1208 images with their left-right reflections) and 1218 background images for training. For testing, there are 1126 pedestrian images (563 images with their left-right reflections) and 453 background images.

Training structure learning: we manually annotate 500 selected pedestrian images in the training set examples. We use all 500 annotated examples to build the PCA spaces for each body segment. In training, each example is learned to update the weight vector. The order of selecting examples in each round is randomly drawn based on the differences of their scores on the predictions and their scores on the true targets. For each round, we choose 300 examples drawn (since structure learning is expensive). We have trained the structure learning on 10 rounds and 20 rounds for comparisons.

Quality of configuration estimates: Configuration estimates look good (figure 2.2). A persistent nuisance associated with pictorial structure models of people is the tendency of such models to place legs on top of one another. This occurs if one uses only appearance and relative geometric features. However, our results suggest that if one uses absolute configuration features as well as different appearance features for left and right legs (implicit in the structure learning procedure), the left and right legs are iden-

tified correctly. The conditional independence assumption (which means we cannot use the angle between the legs as a feature) does not appear to cause problems, perhaps because absolute configuration features are sufficient.

Bootstrapping the SVM: The final SVM is bootstrapped, as in [8]. We use 2146 pedestrian images with 2756 window images extracted from 1218 background images. We apply the learned structure model to generate on these 2416 positive examples and 2756 negative examples to train the initial SVM classifier. We then use this classifier to scan over 1218 background images with step side of 32 pixels and find hard examples (including false positives and true negatives of low confidence by using LibSVM [37] with probability option). These negatives yield a bootstrap training set for the final SVM classifier. This bootstrap learning helps to reduce the false alarm significantly.

Testing: We test on 1126 positive images and scan 64x128 image windows over 453 negative test images, stepping by 16 pixels, a total of 182, 934 negative windows.

Scanning rate and comparison: Pedestrian detection systems work by scanning image windows, and presenting each window to a detector. Dalal and Triggs established a methodology for evaluating pedestrian detectors, which is now quite widely used. Their dataset offers a set of positive *windows* (where pedestrians are centered), and a set of negative images. The negative images produce a pool of negative *windows*, and the detector is evaluated on detect rate on the positive windows and the false positive per window (FPPW) rate on the negative windows. This strategy — which evaluates the detector, rather than the combination of detection and scanning — is appropriate for comparing systems that scan image windows at approximately the same high rate. Current systems do so, because the detectors

require nearly centered pedestrians. However, the important practical parameter for evaluating a system is the false positive per image (FPPI) rate. If one has a detector that does not require a pedestrian to be centered in the image window, then one can obtain the same detect rate while scanning fewer image windows. In turn, the FPPI rate will go down even if the FPPW rate is fixed. To date, this issue has not arisen, because pedestrian detectors have required pedestrians to be centered.

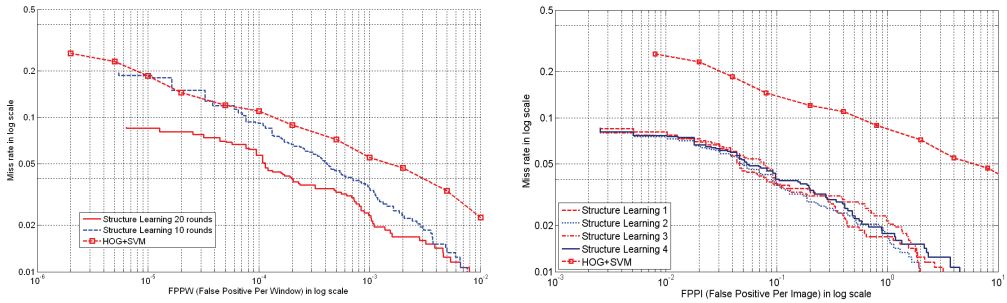

Figure 2.3. Left: *a comparison of our method with the best detector of Dalal and Triggs,* on the basis of FPPW rate. *This comparison ignores the fact that we can look at fewer image windows without loss of system sensitivity. We show ROC's for a configuration estimator trained on 10 (blue) and 20 (red) rounds of structure learning. With 20 rounds of structure learning, our detector easily outperforms that of Dalal and Triggs.* **Right:** *a comparison of our method with the best detector of Dalal and Triggs,* on the basis of FPPI rate. *This comparison takes into account the fact that we can look at fewer image windows (by a factor of four). However, scanning by larger steps might cause a loss of sensitivity. We test this with a procedure of replicating positive examples, described in the text, and show the results of four runs. The low variance in the detect rate under this procedure shows that our detector is highly insensitive to the configuration of the pedestrian within a window. If one evaluates on the basis of false positives per image — which is likely the most important practical parameter — our system easily outperforms the state of the art. (Originally, we also compared with [38]; however, they later corrected their result due to an experiment setup error which did not outperform Dalal and Triggs' method. Therefore, we do not show comparisons with theirs in this plot)*

2.5.1 The Effect of Configuration Estimates

Figure 2.3 compares our detector with that of Dalal and Triggs on the basis of detect and FPPW rates. It shows that our detector outperforms Dalal and Triggs's detector. Moreover, we scan images at steps of 16 pixels (rather

Figure 2.4. *In color, original positive examples from the INRIA test set; next to each, are three of the replicates we use to determine the effect on our detection system of scanning relatively few windows, or, equivalently, the effect on our detector of not having a pedestrian centered in the window. See section 2.5.1, and figure 2.3.*

than 8 pixels for Dalal and Triggs). This means that we scan four times fewer windows than they do. If we can establish that the detect rate is not significantly affected by big offsets in pedestrian position, then we expect a large advantage in FPPI rate.

We evaluate the effect on the detect rate of scanning by large steps by a process of sampling. Each positive example is replaced by a total of 256 replicates, obtained by offsetting the image window by steps in the range -7 to 8 in x and y (figure 2.4). We now conduct multiple evaluation runs. For each, we select one replicate of each positive example uniformly at random. For each run, we evaluate the detect rate. A tendency of the detector to require centered pedestrians would appear as variance in the reported detect rate. The FPPI rate of the detector is not affected by this procedure, which evaluates only the spatial tuning of the detector.

Figure 2.3 compares system performance, combining detect and scanning rates, by plotting detect rate against FPPI rate. We show four evaluation runs for our system; there is no evidence of substantial variance in detect rate. Our system shows a very substantial increase in detect rate at fixed FPPI rate.

2.6 Discussion

There is a difficulty with the evaluation methodology for pedestrian detection established by Dalal and Triggs (and widely followed). A pedestrian detector that tests windows cannot find more pedestrians than there are windows. This does not usually affect the interpretation of precision and recall statistics because the windows are closely packed. However, in our method, because a pedestrian need not be centered in the window to be detected, the windows need not be closely packed, and there is a possibility of undercounting pedestrians who stand too close together. We believe that this does not occur in our current method, because our window spacing is narrow relative to the width of a pedestrian.

Part representations appear to be a natural approach to identifying people. However, to our knowledge, there is no clear evidence to date that shows compelling advantages to using such an approach (e.g. the review in [24]). We believe our method does so. Configuration estimates appear to have two important advantages. First, they result in a detector that is relatively insensitive to the placement of a pedestrian in an image window, meaning one can look at fewer image windows to obtain the same detect rate, with consequent advantages to the rate at which the system produces false positives. This is probably the dominant advantage. Second, configuration estimates appear to be a significant help at high specificity settings (notice that our method beats all others *on the FPPW criterion* at very low FPPW rates). This is most likely because the process of estimating configurations focuses the detector on important image features (rather than pooling information over space). The result would be that, when there is low contrast or a strange body configuration, the detector can use a somewhat lower detection threshold for the same

FPPW rate. Figure 2.1 shows human configurations detected by our method but not by our implementation of Dalal and Triggs; notice the predominance of either strange body configurations or low contrast. Structure learning is an attractive method to determine which features are discriminative in configuration estimation, and it produces good configuration estimates in complex images.

CHAPTER 3

PARSING HUMAN FIGURES WITH RELATIONAL MODELS

In chapter 2, we show that parsing helps improve detection. This chapter presents how to make parsing better with a full relational model. The method shows quantitative evidence that a full relational model of the body performs better at upper body parsing than the standard tree model, despite the need to adopt approximate inference and learning procedures. The method uses an approximate search for inference, and an approximate structure learning method to learn. We compare our method to state of the art methods on the UIUC people dataset (which depicts a wide range of poses), on the standard Buffy dataset, and on the reduced PASCAL dataset published recently. Our results suggest that the Buffy dataset over emphasizes poses where the arms hang down, and that leads to generaliza- tion problems.

3.1 Introduction

In human parsing, we have a region of interest (ROI) containing a person, perhaps produced by a detector, and we must produce an accurate representation of the body configuration. This problem is an important part of activity recognition; for example, the ROI might be produced by a detector, but we must know what the arms are doing to label the activity. The representation produced is usually a stick figure, or a box model, but may be image regions or joint locations. All representations encode the configuration of body segments.

It is usual to represent pairwise spatial relations between locations structured into a kinematic tree, so that dynamic programming can be used for inference [9, 10]. The joint relations encoded by the kinematic tree model are important, but there are other important relations. Limbs on the left side of the body usually look like those on the right. This cue should be important, because limbs are genuinely difficult to detect, particularly in the absence of an appearance model. Even the strongest recent methods have difficulty detecting forearms (e.g. [39], 32%, p8). Inference difficulties occur when one encodes relations between all pairs of segments, because finding the best parse now becomes max-cut. Approximate inference on sets of extended image segments can produce good parses for difficult images [11]. However, there is no evidence comparing the benefits of a full model against the cost of approximate inference.

In this chapter we explore the advantages of representing a full set of relations for human parsing. We show strong quantitative evidence that the advantages of representing a full set of relations between segments outweigh the costs of approximate inference and approximate learning. We concentrate

Figure 3.1. *A tree model of the upper body represents only relations that can be organized as a tree (always the kinematic relations in the natural tree,* **top left***). By doing so, it omits relations that are important; for example, the left arm typically looks like the right arm. A full model (***bottom left** *— we have indicated only some of the relations omitted by the tree model) encodes these relations, at the cost of approximate inference and approximate training. There is qualitative evidence in the literature that full models can yield good parses [40]; in this chapter, we show quantitative evidence on two datasets that a full model offers strong advantages over a tree model. On the* **right***, we show parses derived from a tree model (***top***) and a full model (***bottom***); note that the appearance constraints in the full model often help arms to be in the right place. This is confirmed by our quantitative data.*

on upper body parsing, and show results on Buffy and Pascal dataset [12], and on a new dataset where the prior on body configuration is quite weak.

3.1.1 Related Work

For segments with known appearance, inference is by dynamic programming (the "pictorial structure model" [14]). A similar approach can be applied to informative local patches [41], or to joint locations using velocities at the joints, assuming known activity [42]. If segment appearance is unknown, it can be recovered from motion sequences [43], or by an iterative procedure of estimating appearance, then configuration, etc. [28]. The iterative procedure can produce good parses, but can fail if the search starts poorly. These methods can be costly, because the search space is a discretization of all possible segment configurations. Improvements result from estimating head location, then pruning the search space [12]; and from tuning the initial appearance model with spatial priors on segment locations and respecting likely interactions between segment appearance models (the upper arm is often the same color as the upper body) [13]. Alternatively, local segment scores can be computed by appearance independent segment detectors; edge based limb detectors give limited performance [44], but a combination of HOG and color segmentation features beats the original iterative process [45].

There has been considerable experimental tuning of various tree models. A ten-body segment model (head, upper body, two for each arm and two for each leg) is now universal. The standard scoring procedure regards a prediction correct if its endpoints lie within 50% of the ground truth segment length from the true positions; this score is fairly generous, because body segments are long. Nonetheless, high scores are difficult to get. Ramanan [28] has

published a widely used dataset for full body localization, on which the best method (Andriluka *et al.*, 2009 [39]) gets 55.2%. Ferrari *et al.* [12] published another for upper body localization, on which the best method (Eichner *et al.*, 2009 [13]) gets 80.3% in its best configuration on Buffy dataset. On a selected subset of PASCAL challenge images, this method gets 72.3%. The strongest methods all use carefully constructed and tuned segment detectors, with search space pruning.

The tree model presents real difficulties: limb occlusions seem to be correlated; tree models tend to prefer parses that superimpose both arms or both legs on one promising set of image segments; and a tree model cannot encode the tendency of the left arm (resp. leg) to look like the right arm (resp. leg). Correlated limb occlusions can be dealt with using a mixture of trees without complicating inference [46]. Doubled limbs can be controlled with "repulsive" edges [13, 47], or by converting the energy function into a posterior probability, drawing samples, and using some process to decide which is best (for example, rejecting parses where arms overlap) [14]. An alternative, which requires approximate inference, is to require that the model covers more of the image pixels [48].

Another important difficulty is determining which poses to work with. In our opinion, the performance of a parser should be as close to pose-independent as possible. That is, the parser should be tested (if not trained) on a dataset with a rich selection of poses at approximately even frequencies. This is complicated, because natural data sources often have strong biases — as we shall see, TV actors in stills tend to have their arms by their sides. The result is very strong effects due to the prior, which can cause generalization problems. For these reasons, we have collected a further dataset that emphasizes rich upper body configurations.

3.2 Method

Our approach defines a search space in the image using a set of tuned body segment detectors. We then build an energy model that is regressed against actual loss for a set of parses of each training image. Our model scores appearance and spatial relations between all pairs of segments in the image. We then find the best parse by an approximate maximization procedure.

We have detectors for upper body, head and arm segments. Our detectors do not distinguish between upper and lower arms. We must choose a label (head, upper body, left/right upper/lower arm, null) for each response. For image $\mathcal{I}$, we build a scoring function $C(L;\mathcal{I})$ which evaluates a labelling L of the responses. We consider only labellings that are consistent, in the sense that we do not attempt to label head detector responses as upper bodys, etc. Write S_i for the i-th body segment in the model, D_j for the j-th detector response in the image, and $L(S_i)$ for the image segment labelled S_i by L. Our energy is a linear combination of unary and binary features, which we write as

$$C(L;\mathcal{I}) = \sum_{i \in \text{features}} w_i \phi_i(L;\mathcal{I}) = \mathbf{W}^T \Phi(L;\mathcal{I})$$

where each feature ϕ_i is either a unary feature (yielding $\phi_i = \phi_i(S_j, L(S_j);\mathcal{I})$) or a binary feature (yielding $\phi_i = \phi_i(S_j, S_k, L(S_j), L(S_k);\mathcal{I})$). We do *not* require that the set of binary features form a tree, but represent all pairs. Our features measure both spatial and appearance relations (section 3.2.4). The scoring function can be converted to an energy by $E(L) = -C(L)$; a probability model follows, though we do not use it.

3.2.1 Searching a full energy model

Finding the best labelling involves solving a general zero-one quadratic form subject to linear constraints, and there is no exact algorithm. While approximate algorithms for MRF's could be applied, most labels are null and there is only one instance of each non-null label, meaning that expansion moves are unlikely to be successful. We use an approximate search procedure that relies on the proven competence of tree models.

The upper body detector is quite reliable, so there are relatively few false positives. This means we can search for a configuration at each upper body, then take the overall best configuration. Because tree models are quite reliable, we can use specialised tree models to produce arm candidates on each side of each given upper body, then evaluate all triples of right arm-torso-left arm. Finally, we use a local search to improve segments.

Obtaining arm candidates: We need to obtain candidates for left (resp. right) arm that have a good chance of being in the final configuration. We can do so by simplifying the cost function, removing all terms apart from those referring to upper body, left (resp. right) upper arm and left (resp. right) lower arm. The resulting simplified cost function is easily maximised with dynamic programming. We keep the top 300 candidates found this way for each side.

Building good triples: We now have 300 candidates each for left (resp. right arm), and a set of head candidates. We obtain the top five triples by exhaustive evaluation of the whole cost function.

Polishing with local search: Limb detectors chatter, because they must respond to contrast at limb edges and limbs are narrow; it is usual to see more than one response near a segment. To counteract this effect, we

polish each of the top five triples. We repeatedly fix five segments and search for the best candidate for the sixth, stopping when all segments have been visited without change. We now report the best of the polished five triples for each upper body.

Detection: We report the parse associated with the best upper body detector response. In principle, one could parse multiple people in an image by keeping all such parses, applying a threshold to the cost function, and using non-maximum suppression (to control chatter at the upper body detector); since most images in evaluation datasets contain single people, and since our focus is on "hard parses", we have not investigated doing so.

Complexity: With 6 human parts in the model, the exact solution will cost $O(T * H * LUA * LLA * RUA * RLA)$ where T, H are torso and head detections, LUA, LLA and RUA, RLA are left upper, lower arms (resp. right upper and lower arms) detections. While T and H are small (less than 10 each), LUA, LLA, RUA, RLA are quite large (normally more than 100 each after pruning by the closeness to the torso), this complexity is practically intractable. However, our approximate solution has complexity $O(T * H * LA * RA) --LA, RA$: numbers of full left (resp. right arms) that we keep top 300 for each). This complexity is tractable, and though it is an approximation, it still proves its benefit of improving the performance. In fact, Our implementation in C just takes around 5 seconds for one parsing on a computer of Xeon 2.27HGz.

3.2.2 Training a full energy model

We wish to train the energy model so that detections using that model are as good as possible. **Structure learning** is a method that use a series of

correct examples to estimate appropriate weightings of features relative to one another to produce a score that is effective at estimating configuration (in general [33, 34]; applied to parsing [49]). For a given image $\mathcal{I}$ and known $\mathbf{W}$ the best labelling is

$$\begin{array}{c}\arg\max \\ L \in L(\mathcal{I})\end{array} \mathbf{W}^T\Phi(L;\mathcal{I})$$

though we cannot necessarily identify it. We choose a loss function $\mathcal{L}(L_p, \hat{L})$ that gives the cost of predicting L_p when the correct answer is $\hat{L}$. Write the set of n examples as $\mathcal{E}$, and $L_{p,i}$ as the prediction for the i'th example. Structure learning must now estimate a $\mathbf{W}$ to minimize the hinge loss as in [35, 50, 33]

$$\min \lambda\frac{1}{2} \parallel \mathbf{W} \parallel^2 + \frac{1}{n} \sum_{i \in examples} \xi_i$$

subject to the constraints

$$\begin{array}{r} \forall i \in \mathcal{E},\ \mathbf{W}^T\Phi(\hat{L};\mathcal{I}_i) + \xi_i \geq \\ \begin{array}{c}\max \\ L_{p,i} \in L(\mathcal{I}_i)\end{array} (\mathbf{W}^T\Phi(L_{p,i};\mathcal{I}_i) + \mathcal{L}(L_{p,i}, \hat{L}_i)) \\ \xi_i \geq 0 \end{array}$$

At the minimum, we can choose the slack variables ξ_i to make the constraints equal. Therefore, we can move the constraints to the objective function, which is:

$$\begin{array}{r} \lambda\frac{1}{2} \parallel \mathbf{W} \parallel^2 + \\ \frac{1}{n} \sum_{i \in examples} \begin{array}{c}\max \\ L_{p,i} \in L(\mathcal{I}_i)\end{array} \left(\begin{array}{c} \mathbf{W}^T\Phi(L_{p,i};\mathcal{I}_i) + \\ \mathcal{L}(L_{p,i}, \hat{L}_i) - \mathbf{W}^T\Phi(\hat{L};\mathcal{I}_i) \end{array} \right) \end{array}$$

Notice that this function is convex, but not differentiable. We use the cutting-plane method of [50], as implemented in SVM-Struct package[1]. Our approach is:

Start: we initialize $\mathbf{W}$, and prepare a pool of candidate labellings $\mathcal{C}_i^{(0)}$ for each example image using the search of section 3.2.1. Then, iterate multiple rounds of

1. for each example, compute the best (most violated constraint) labelling $L_{p,i} = \arg\max_{L \in \mathcal{C}_i^{(k)}} \mathbf{W}^T \Phi(L; \mathcal{I}_i) + \mathcal{L}(L_{p,i}, \hat{L}_i)$.

2. pass these labellings to SVM-Struct to form cutting planes to update $\mathbf{W}$.

This procedure will stop until there are no violated labelling found (in this case, the ground truth labelling is the highest score) or no significant change in the objective value when updating $\mathbf{W}$. We observe that the learning converges after 50-60 iterations.

3.2.3 Part detectors

We have detectors for upper body, head and arm segments, but do not distinguish between upper and lower arms for a total of three part detectors. The detectors are oriented, and we use a total of 25 orientations of $(-180^o..180^o)$ for arm detectors and 13 orientations of $(-90^o..90^o)$ for head detector and upper body detector. We use code from Felzenszwalb [2] to compute HOG features [51] for upper body and head detectors. Arm segment detectors use HOG features and self-similarity features (from [52], using the implementation of V. Gulshan). This detector does not distinguish between upper and

[1] $http://svmlight.joachims.org/svm_struct.html$
[2] $http://people.cs.uchicago.edu/\ pff/latent/$

Detector	Size	Features	EER
Upper body	80x80	HOG	0.096 +/-0.005
Head	56x56	HOG	0.123+/0.012
Lower arm	30x30	HOG, SSIM	0.249+/-0.068

Table 3.1. Summary of part detectors. Equal error rates (EER) are computed with 5-fold cross validation. The lower arm detector is not comparable to others as it tends to be dataset dependent. We operate the detectors at 92% recall and given a upper body candidate we keep the 300 best lower arm responses.

lower arms, because locally they are similar in appearance. Upper arms can be difficult to detect, because there may be little contrast between the segment and the body. To overcome this difficulty, we also use a *virtual* upper arm detector, obtained by joining points nearby the top of the upper body segment to the elbows of nearby lower arm segments.

Lower arms can be strongly oriented (i.e. long and thin), and our arm detector may respond more than once to a lower arm in a lateral view. Extending the support of the detector does not help, unless one searches an impractical number of orientations. We deal with this by expanding the search space: we add new lower arms to the pool of detector responses, made by fusing nearby arm detections at the same orientation.

All part detectors are linear SVM trained on cropped parts from our dataset and from some of Buffy_s5e3 dataset. We bootstrap upper body and head detectors on a subset of background images, and lower arm detector on subset training images (regions outside the subject box). Table 3.1 summarizes part detector parameters.

3.2.4 Features

We use a binning scheme, after [28]. Binning takes a feature such as distance and quantizes the range to a set of discrete bins, then sets the bin into which

a value falls to be one and all others zero. We find it helpful to antialias by splitting the vote among nearby bins.

Unary features are the detector score at the detection labelled with a part label (converted to a probability using the method of [53]), and a binned vector representing the part length. For virtual upper arms, we have no detector score and instead use the value of the detector response at the lower arm used to create the virtual upper arm.

Binary features are different for different pairs of parts. We use six parts: upper body (from chest to navel), head (from top forehead to chin), left upper arm (LUA), left lower arm (LLA), right upper arm (RUA), and right lower arm (LLA). For each pair, we compute features from *distance*, *appearance*, *angle*, or *overlap*, according to the scheme of table 3.2.

Distance features for a pair of segments consist of a binned vector representing distance between endpoints, concatenated with the actual distance. The **comparative appearance feature** is formed from a set of appearance vectors. The appearance vectors consist of normalized color histograms, normalized Gabor filter histograms [54], and a histogram of textons [55]. For each type of appearance vector, we compute the χ^2 distance between the vectors corresponding to the two segments to be compared. For speed, integral images of appearance features are precomputed over reoriented images. **Angle features** are given by a binned angle vector representing signed angle from segment 1 to segment 2 in the range $(-90^o..90^o)$ for the head-torso pair, and $(-180^o..180^o)$ for all others. **Overlap features** give the ratio of endpoint distances to segment length, with the ratio computed for each segment. There are a total of 707 features.

Parts	Upper body	Head	LUA	LLA	RUA
Upper body	-	-	-	-	-
Head	D,A,N,O	-	-	-	-
LUA	D,A,N,O	A,O	-	-	-
LLA	D,A,N,O	A,O	D,A,O	-	-
RUA	D,A,N,O	A,O	A,O	A,O	-
RLA	D,A,N,O	A,O	A,O	A,O	D,A,O

Table 3.2. This table shows pairwise features to be computed. [D]: distance binning, [A]: appearance difference, [N]: angle, [O]: overlap

3.3 Experimental results

We compare a full model to a tree model on three datasets, described below. The full model is trained as above. The tree model is trained in the same way, but with the weights of features representing relations not in the tree clamped at zero. Inference (and so training) of the tree does not require a polishing step, because dynamic programming is exact. The tree is the usual kinematic tree (figure 3.1).

3.3.1 Dataset

We describe results on three datasets. The first is the Buffy dataset of [12], in various partitions. This dataset has little variation in layout (figure 3.2). The second is the subset of Pascal images marked up and released by [13]. Human parsing results are usually intended to drive activity recognition, which is at its most interesting when the body takes unusual postures. Methods that work well on a dataset with a strong spatial bias may do so because (say) they are particularly good at some common poses; such methods may not be useful in practice. For this reason, we have created a third dataset of 593 images (346 training, 247 test), marked up with stick figures by hand. This dataset is built to have aggressive spatial variation in configuration

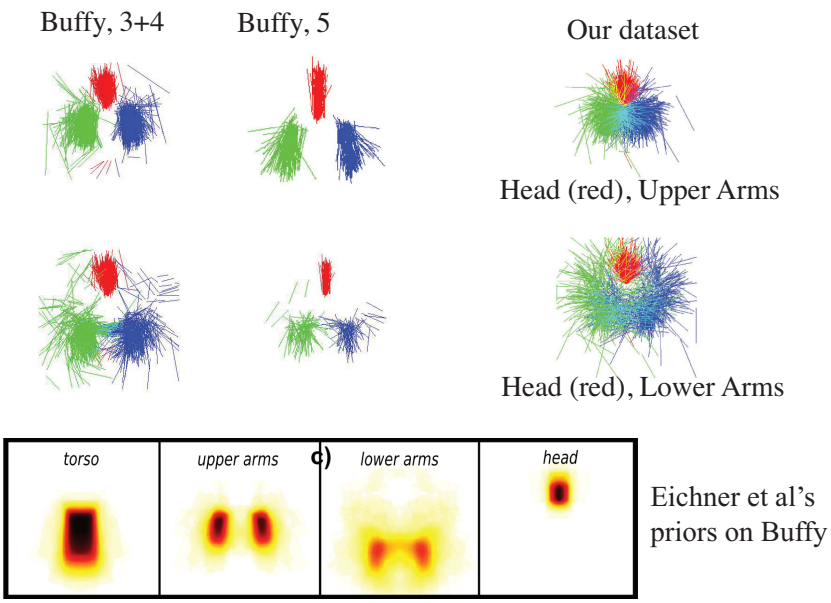

Figure 3.2. *In the Buffy dataset, upper arms hang by the sides of the body, and lower arms mostly do so as well. This very strong spatial prior can overcome contributions by other parts of a model, but impedes generalization.* **Above:** *Scatter plots of head and upper arm (***top row***) or lower arm (***bottom row***) sticks with respect to fixed upper body position for the Buffy 3 and 4 ground truth, Buffy 5 ground truth, and our ground truth. Notice how compact the prior configuration is for the Buffy datasets. Our dataset emphasizes a wide range of body configurations.* **Below:** *Part of figure 1, from [13], for reference, showing the location priors derived in that work for the Buffy dataset; again, note the highly compact prior.*

(figure 3.2).

3.3.2 Results

We follow convention and measure performance with PCP (Percentage of Correctly estimated body Parts). In this method, a segment is correct if its endpoints lie within 50% of the length of the ground truth from the annotated location [12]. Since our method produces one parse for each upper body detector response, we apply non-maximum suppression to the score, to prevent effects from multiple nearby upper body detector responses. As in Eichner *et al.* [13], we evaluate PCP only for stickmen whose upper body response overlaps the correct upper body.

On Buffy and Pascal, our method obtains 62.3% and 62.7%, respectively

(compare 80.3% and 72.3%, Eichner *et al.* [13]). However, there are two difficulties with these dataset (especially for Buffy), both potentially quite serious. First, there is little variation in pose. Figure 3.2 shows a scatter plot of ground truth head and arm segments for overlaid upper body segments. Head, upper arm and lower arm segments all have relatively little scatter — most figures are upright. Second, the contrast for many frames is relatively low. Both issues suggest that careful detector engineering will produce improvements in performance by overcoming contrast difficulties. Detector engineering is a valuable contribution which is responsible for all advances on the buffy dataset, but it will not identify better or worse modelling techniques. Because the spatial configuration varies so little in the Buffy dataset, comparisons of modelling techniques on this dataset should be approached with caution.

On all three datasets, the full model significantly outperforms the tree model (table 3.3). This is most likely because appearance consistency constraints between upper arms help overcome relatively low contrast at the boundary. Typical results suggest that improvements occur because consistency in appearance (the left arm must look like the right) is a cue that helps parse, and possibly because the model is spatially more rigid than a tree model (figure 3.5). The value of these cues outweighs the cost of approximate inference and approximate learning. Our parser can be configured as a detector by applying non-maximum suppression to the parse score and thresholding.

Model		Test set		
		our_test	Buffy_s5e256	Pascal
	Train set			
Full model	our_train	0.663	0.623	0.627
	Buffy_s5e256_sub	0.583	0.676	0.625
	Buffy_s5e3&Pascal_sub	0.613	0.628	
Tree model	our_train	0.608	0.552	0.565
	Buffy_s5e256_sub	0.545	0.629	0.599
	Buffy_s3&Pascal_sub	0.565	0.596	
Eichner [13]	Buffy_s5e2to6	0.557		0.675
	Buffy_s5e34&Pascal	0.559	0.801	

Table 3.3. *Average PCP over body segments for a full model; for a tree model; and for Eichner and Ferrari [13], who use a tree model with a location prior to recover appearance. Performance of the full model is much better than performance of tree models, except for the model of Eichner and Ferrari applied to Pascal or to buffy_s256. However, for all models, training on Buffy_256 leads to strong generalization problems (performance on Buffy_256 is much better than performance on other test sets), most likely because of the quite strong bias in arm location. We believe that the very strong performance of Eichner and Ferrari on Buffy_s256 should be ascribed to the effects of that bias. Buffy_s5e3&Pascal appears to contain a similar, but smaller, bias (compare training on this and testing on Buffy_s256 with training on this and testing on our_test). We do not have figures for Eichner and Ferrari's method trained on Buffy_s2to6 and tested on Buffy_s256. Note that: Buffy_s5e256_sub and Buffy_s5e3&Pascal_sub are subsets of 150 examples randomly chosen for each dataset.*

Figure 3.3. *Examples of stick-figure, upper body parses of figures in our dataset produced by the full model trained on our dataset* **top** *row, our tree model* **top-center** *and the code of Eichner et al.***bottom center** *(trained on buffy_2to6) and* **bottom** *(trained on buffy_3&4 and pascal), all applied to our dataset. Red: upper body; Green: head; Blue-Purple: left upper/lower arm; Green-Yellow: right upper-lower arm. Note doubled arms produced by the tree model and a tendency for Eichner et al.to produce hanging arms, most likely a result of the strong geometric prior in their training datasets.*

3.4 Discussion

We have shown quantitative evidence that a full relational model of the body performs better at upper body parsing than the standard tree model, despite the need to adopt approximate inference and learning procedures. We have obtained our results on a new dataset where there is extensive spatial variation in body configuration. Our results suggest that appearance consistency constraints help localize upper arms better.

Figure 3.4. *Examples of stick-figure, upper body parses of figures in the buffy produced by the full model trained on ours* **top** *row, and our tree model trained on ours* **bottom**. *Red: upper body; Green: head; Blue-Purple: left upper/lower arm; Green-Yellow: right upper-lower arm. Note doubled arms produced by the tree model, and the strong tendency for poses to have hanging arms.*

Figure 3.5. *Examples of stick-figure, upper body parses of figures in our dataset produced by the full model trained on our dataset* **top** *row, our tree model* **top-center** *and the code of Eichner et al.***bottom center** *(trained on buffy_2to6) and* **bottom** *(trained on buffy_3&4 and pascal), all applied to our dataset. Red: upper body; Green: head; Blue-Purple: left upper/lower arm; Green-Yellow: right upper-lower arm. Note doubled arms produced by the tree model and a tendency for Eichner et al.to produce hanging arms, most likely a result of the strong geometric prior in their training datasets.*

CHAPTER 4

BOOSTING HUMAN PARSERS WITH POSELET PRUNERS

Full relational models prove to be better than tree structure models for human parsing problems, but these models must deal with high complexity in inference, especially if one wants to take advantages of complex appearance models. Huge search spaces mostly prevent practical uses of full relational models. This chapter demonstrates that full relational models can be tractable once we have a good pruning strategy. We propose a series of pruning models from simple part pruners to complex pruners with tree models which are able to prune down 99.6% states per part to about 500 states per part without loosing performance. Using complex appearance models in full relational models makes significantly improvements on the localization of most parts. We compare our method to state-of-the-art methods on the challenging datasets UIUC Sport.

4.1 Introduction

Human body parsing problems, where one wants to localize torso, head, upper arms, lower arms, upper legs and lower legs, have increasingly interested the computer vision community in recent years. Parsing has proven important in many other computer vision tasks such as person detection, action recognition. There have been substantial advances in performance on human parsing problems recently [56, 57, 15, 13, 17, 39], yet very accurate and fast human parsing is still a challenging problem.

Human parsing is usually cast as a structured output problem, where the search spaces is huge, making it difficult to find the best structures. In a typical example, there are about $100 * 100 * N_{poselet}$ ($N_{poselet}$ is the number of orientations for primitive parts) states per part (a full body is often represented by 10 parts as shown in Figure 4.2). This results in $N_{poselet}^2 * 10^8$ evaluations of a pairwise part relation. This high computation issue prevents the use of sophisticated appearance models (which are known to be very useful for better performance [13, 17]) because evaluating pairwise appearance relations between related parts leads to intractable searches as visualized in Figure 4.3. For this reason, people tend to choose models that lead to low computation in search. A well-known tree-based model as in pictorial structures [14] is widely used for the benefit of fast inference by dynamic programing. Complex full relational models are carefully designed with approximate inference [15] to make them tractable. Despite the use of approximation methods, Tran & Forsyth [15] have shown evidence that full relational models are better than tree-based models.

A crucial step to promote the practical use of full relational models is to make inference faster. One approach is to develop good strategies to prune

down search spaces. A good pruner eliminates part states that are unlikely to be in the correct structures leaving as few remaining states as possible without wrongly removing the correct states. Some recent work has proposed very good pruning strategies. Mori *et al.* [11] used over-segmentation to generate limb hypotheses. Ferrari *et al.* [12] used upper-body detectors and foreground/background segmentations to narrow down the possible areas of limbs around the body candidates. Felzenszwalb *et al.* [16] proposed a cascade model by using early stopping. Tran and Forsyth [15] pruned the search space using local searches. Recently, Sapp *et al.* [17] developed a cascade of pictorial structure-based pruners to progressively prune states from low resolution to high resolution. We also design a series of pruners in a cascaded fashion but use three different types of pruner from simple part pruners to complex tree-based pruners. Our pruners work on the actual state space resolution. First, we use simple and very fast pruners to quickly prune away more than 75% of the part states. Then more accurate yet more complex pruners are used to get a small number of remaining states. We show that our pruners can prune down up to 99.6% to leave about 400-500 states per part in the final pruned space. This set of states is small enough to allow approximate inference of highly relational models with complex appearance models to parse highly accurate body parts.

While other methods that use only primitive parts in designing pruners as well as parsers, our methods use context from large parts to support pruning decisions. Bourdev *et al.* [58] introduced a new concept *poselets* localizing objects such as pedestrians. They marked up object 3D configurations and clustered part examples of similar 3D configurations into groups to define poselets. We then developed a hierarchical poselet model to apply to human body parsing (Wang *et al.* [57]). In that work, we define human parts as

either primitive parts or large chunks of body (full arms, torso-arms, torso-head, whole legs, etc., see Figure 4.2). Parts having similar 2D configuration are clustered into groups called *poselets*. Examples of two poselets of the part **legs** are shown in Figure 4.1. Large parts provide contextual information to guide pruning small parts (e.g. the upper left arm gets support from the left arm). [57] has shown the usefulness of large parts in the later case for building a parser. We also show that our pruners are more effective when augmented by large parts.

Figure 4.1. *Examples of two poselets of the part legs. Each row are patches of the same poselet. The last column is the HOG templates of the poselets. This figure is from Wang et al. [57].*

Our pruner cascade has three pruner types. The first pruner type is part pruners. We call them *part pruners* because they use simple structure models of only parts without relations. In this model, there are no relations between parts. This makes the inference extremely fast because finding the best structure now becomes searching the best state for each part independently. These pruners filter out unlikely 2D location states (mid-points) of that part in the image. Traditionally, it is usual to use part detectors as *part pruners* with a corresponding global pruning threshold. This has been shown to be inefficient for small parts such as half limbs ([17]). Instead, we augment our part pruners with their related large parts. For each part, we train a linear

function of its poselet responses and the poselet responses from its related large part (approximate at the same location) to score high on correct states and low otherwise. Using this support from large parts, our part pruners are very efficient compared to those using traditional part detectors. An intuitive explanation is that large parts have stable evidence (shape and appearance) to help disambiguate small parts. For example, a *lower arm pruner* can easily confuse any rectangular shape with a good part and so will most likely keep it in the search space. In the context of a full arm, a pruner can safely reject these false lower arms if there is no evidence of a full arm nearby.

The second pruner type has a more complex structure: a tree-based structure of both primitive parts and large parts (see Figure 4.6). These pruners also work to prune unlikely 2D location states. We call them *2D-tree-based pruners*. While *part pruners* get only appearance support from their related large parts, *2D-tree-based pruners* get both appearance and spatial support from other parts. The pruning score of a part at a particular 2D state is computed by a weighted combination of appearance (unary) scores and spatial (binary) scores with the best states of other parts in a joint configuration. These are *max-marginal* scores, of Sapp *et al.* [17]. These joint configuration scores help to recover part states having weak part evidence if they have strong spatial relation evidence. Computing max-marginal scores can be done quickly by dynamic programing on the tree structure of the models. Moreover, it also take the advantage of reduced state spaces after part pruners.

The third pruner type is *enhanced tree-based pruners*. We use the same tree structures as used in *tree-based pruners* for this type of pruner, except it works on the full representation of part states. A part state is now represented by a 2D location of its midpoint and a poselet index. This full representation

is also the actual representation of part states for the search. For the primitive parts, poselet indexes are the actual part orientations. After the first two pruner types, the remaining states are 2D locations of part midpoints. We then enumerate each 2D part state by all part poselet indexes to construct full representation of part states. The enhanced tree-based pruners keep pruning this set of states until they leave a small number of states for each part for the search. These pruners tend to focus on eliminating unlikely part poselets having lower scores in joint configurations with other parts. Also, we add the prior of part poselets and pairwise poselet co-occurrences to the pruning scores. This prior has shown to be useful in [56] in which their notion of parts are small patches of body joints. In our case, these priors are even stronger because large parts are distinctive (from background clutters) to define a small set of dependent small parts. For example, a particular full arm poselet tends to co-occur with a few poselets of upper arms and lower arms. We build a cascade of pruners from these three pruner types. This turns out to be very effective in ruling out unlikely part states. In the challenging dataset UIUC Sport [1], our cascade can prune more than 99.6% of part states while still achieving 54.9% of $PCP_{0.2}$ of lower arm part states and 76.5% of $PCP_{0.2}$ of torso part states.

We finally demonstrate the effectiveness of our pruning strategy by building a highly relational model of human parser on the remaining set of states. Our human parser uses both primitive parts and large parts as in the models of Wang *et al.*[57]. However, Wang's model is not capable of using appearance models because it searches on unpruned part state spaces. Our pruned part state spaces are small enough to do it. We show significant improvements compared to Wang's method on the UIUC Sport dataset even though

[1] Introduced by Wang *et al.* in [57]

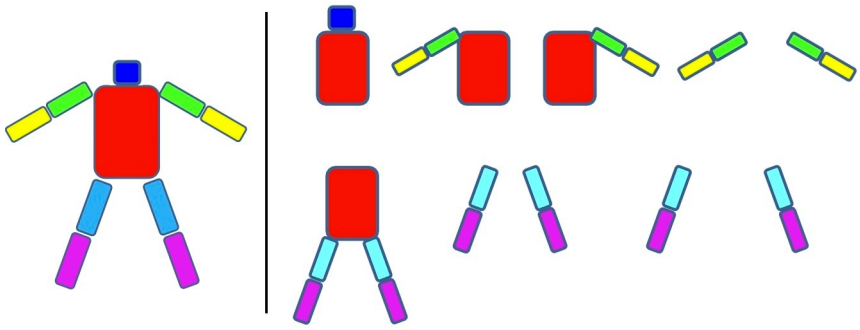

Figure 4.2. *A typical 10-part representation of human body parts (left-most), where each part is represented by either in a stick format of two endpoints or a mid-point format of a part mid-point location and orientation. Right figures are large parts which are big chunks of body. In total, there are 20 parts including both primitive parts and large parts. Also, the whole body part is considered as a large part in the model ([57]).*

we only use color consistency models for left/right limbs (see Table 4.4 for the comparisons to other methods).

4.2 Related Work

Our work is closely related to a line of research on using part-based models for human pose estimation. These methods model the configuration of a human figure as an assembly of parts connected in some fashion. Some early representative work includes the "cardboard people" [59] and pictorial structures [14].

For computational efficiency reasons, most work [14, 39, 28] on part-based models assumes a tree-structured model, e.g. the kinematic tree. However, the computational advantage of tree-structured model comes with a cost – they do not adequately model the full set of relationships between body parts. Certain important relationships (e.g. color symmetry between left/right limbs) are ignored in the tree models. There has been some effort to build models beyond trees, e.g. loopy graphs [60, 40, 61] , mixtures of trees [46, 62]. Most recently, we have demonstrated the effectiveness of using

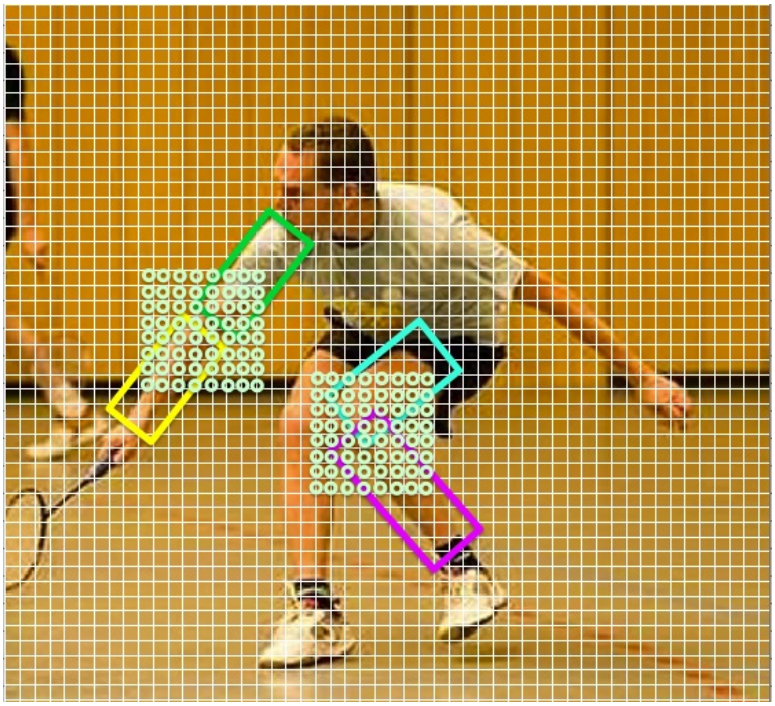

Figure 4.3. *State space of a half limb part. For a typical image of* $400 * 400$ *and grid size of 4 pixels, there are* $100 * 100 * N_p$ *states (*N_p *is the number of part poselets, ranging from 8 to 15). For each binary potential between two related parts, we must do about* $N_p^2 * 10^8$ *evaluations. There are more evaluations if we use complex appearance models in the binary potentials. This computation is intractable for the search of the best structures.*

a fully connected graph representing the complete set of relationships among body parts (Tran and Forsyth [15]). The advantage of using a full relational model is that we can incorporate more sophisticated appearnce features (e.g. to encourage color symmetry between left/right limbs) that have been proved to be useful.

The key challenge of using full relational models is how to peform learning and inference efficiently. One viable solution is to reduce the search space. For example, Mori *et al.* [11] used over-segmentation to generate limb hypotheses. Ferrari *et al.* [12] used upper-body detectors and foreground/background segmentations to narrow down the possible areas of limbs around upper body candidates. Felzenszwalb *et al.* [16] proposed a cascade model by using early stopping. Tran and Forsyth [15] pruned the search space

using local searches. The limitation of these pruning approaches is that the pruning strategy is disconnected from the human pose estimation algorithm. Recently, Sapp *et al.* [17] proposed a cascaded coarse-to-fine pruning method which integrates learns how to prune efficiently. However, at coarse resolutions in early stages, part evidence is not strong due to low resolution and therefore pruners will be more likely to miss correct part states in early stages. Our pruners work on actual state space resolutions to ensure that we always use actual part evidences which are important factors of the pruning models. Moreover, our pruners use large parts as contextual information to help resolving ambiguities of small parts.

Previous approaches (e.g. [17, 15]) try to directly prune the search space of body segments (e.g. torso, head, half-limbs). In this thesis, we would like to argue that pruning at this level of details is inherently difficult, since there are many things (e.g. rectangular shape such as windows, buildings) in an image that look like body segments. Instead, we want to prune the search space by exploiting information from larger body parts. Our work is inspired by the hierarchical poselet approach for human parsing by Wang *et al.* [57]. In their work, they use large body parts (e.g. one part can be torso+legs) as contextual information to guide the search of small parts (e.g upper left leg). The advantage of using large parts is that they typically distinctive from the background clutters. In this work, we use the same intuitation and try to do pruning using large body parts.

The cascade classifier model by Viola and Jones [63] has been a powerful device for object detection problems. Its advantage is to build a strong classifier from a series of fast and low-cost classifiers. Each classifier in the cascade can quickly reject easy negative examples in early stages. Weiss and Taskar [64] proposed a similar cascade approach for pruning search state spaces. In

their work, they developed *structured prediction cascades* of structure models to progressively filter unlike states in the search space using max marginals. Sapp *et al.* [17] then employed this model to human parsing problems. They built a sequence of tree structured models to prune unlikely part states from coarse to fine resolutions of the search space. The final tree structured parser is capable of exploiting rich appearance models for the pairwise part relations because the pruned search space is small enough. We also follow the same spirit of structured prediction cascades. Our pruners in the cascade have three main differences to those of Sapp's. First, Sapp's models use only tree structures while our pruners have two main types of structure models: simple structures (only one part) in early stages and increasingly complex tree structure models later on. Simple structures are easy to built yet effectively filter out more than half of the states. Second, our pruners work on actual state resolutions while Sapp's work on coarse-to-fine resolutions. We argue that missing a state at early stages in coarse resolution would cause missing many states in later stages. Third, our pruners get support from large parts as contextual information. Experiments show that our remaining state spaces contain relatively good part states close to ground truth states (Table 4.2).

4.3 Hierarchical Poslets

Bourdev *et al.* [65] first introduced *poselets*. In their work, they marked up 3D body joints and clustered part examples based on the similarity of their 3D configurations. Each cluster defined *a poselet*. They trained a HOG-SVM clasifier for each poselet. Outputs from posetlet classifiers are used to vote for object candidates (e.g. pedestrian) in the image by a Hough voting

scheme. However, this voting scheme does well on localizing objects (such as detecting pedestrian) but not enough for human parsing because human parsing problems are highly structured outputs.

Recently, we developed a hierarchical poselet model for parsing human body parts (Wang *et al.* [57]). Our notions of parts are not only primitive parts (head, torso, half limbs) but also large parts - large chunk of areas of human body (whole body, torso-head, left/right arms, etc., see Figure 4.1 for all large parts). We also cluster part examples of similar shape into *poselets* and train an $HOG - template$ for each poselet (see Figure 4.4). A relational model of 20 parts (both primitive parts and large parts, see Figure 4.8) is built for body part estimation. Our argument is that large parts provide contextual information that is helpful for detecting and localizing small parts. This should extend to pruning. For example, upper left arm states will be filtered out if there is no evidence of the left arm poselets nearby.

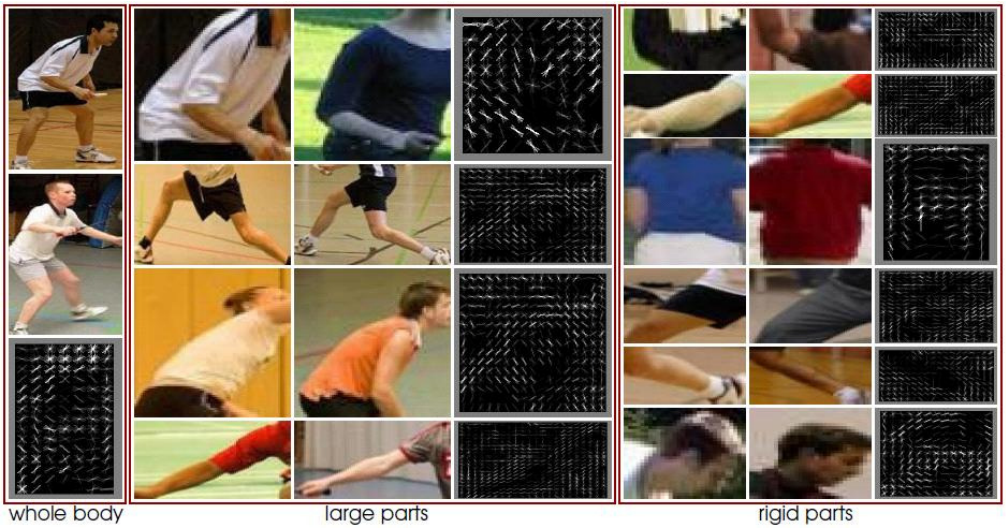

Figure 4.4. *This figure shows examples of part poselets and templates. The left figure is for one poselet of the whole body part and its template (bottom). Each row in the middle figure is a poselet of a large part and its template (right column). The right figures are poselets of rigid parts (one for each row) and the templates (right column) with respect to the poselets. This figure is from Wang et al. [57].*

Large parts as contextual information for pruners: We use large

parts in pruning models. We build pruners that prune both large parts and primitive parts, except large parts do not require high accuracy. By relaxing the pruning criteria (see Section 4.8), reasonably good large part segments in the remaining states help to produce good pruning results of primitive parts.

There are two ways that we use the support from large parts. For part pruners (no spatial relations with other parts), we add the poselet responses of the related large parts at the same 2D locations as augmenting unary features. For tree-based pruners, large parts will be nodes in the relational models. Primitive parts now get spatial support from large parts. Because large parts are distinctive (strong local evidence), they help remove more unlikely primitive part states and recover likely primitive part states having weak evidence.

Structured hierarchical models for parsers: We organize parts in a hierarchical fashion and build a relational model with each part as a node in the graph (Figure 4.8). At the top level is the *whole body part*. Further down are smaller parts and primitive parts. There are links among related parts to express their spatial/appearance relations. At first, we built the parser on the full (unpruned) set of part states [57] and on the small (pruned) set of part states. On the full set of states, we are not capable of using pairwise appearance relations because the number of states per part is too big. After reducing the search space, we can use the pairwise appearance models. Our experiments show their usefulness in localizing parts with the superior performance.

Why we need pruners: We want to build a good parser using a full relational model. Finding the optimal pose configuration involves solving the equation $L^* = \arg\max_L C(L; \mathcal{I}; \mathcal{W})$. This is a standard MAP inference problem in undirected graphical models. Since the graph in Figure 4.8 is

not a tree, we use loopy belief propagation (LBP). Suppose each l_i can take K different values, the computational complexity of LBP is $O(K^2)$. If we naively apply LBP, K is approximately 10,000 for each part. This is obviously too slow. In order to speedup the process, various algorithmic tricks (e.g. distance transform [14]) have been developed. However, these algorithmic tricks usually only handle specific forms of pairwise potential functions. In particular, they require that the pairwise potentials do not depend on the image.

In our work, we would like to exploit richer forms of pairwise potentials, including the ones that depend on the image. The color symmetry of left/right limbs is a representative example of such pairwise potentials. Luckily, after running the pruners in section 4.5, each part typically only has about 500 states, i.e. $K \approx 500$. This allows us to run LBP in a brute force way with $O(K^2)$ complexity. In our experiment, finding L^* in typical examples takes very fast with color symmetry pairwise potentials.

4.4 Problem settings

We summarize the concepts and the notations used in later formulations in Table 4.1. We model the pruners and parsers by relational models $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the set of *nodes* (e.q. body parts) and $\mathcal{E}$ is the set of edges (each edge says two connected nodes (parts) having some relations (spatial and/or appearance). The relational structures might be different depending on types of pruners and parsers. In practice, we use three structures. For *parsers*, we use a hierarchical loopy graph of M=20 nodes (Figure 4.8). For *tree-based pruners*, we use the same hierarchical structure as in parsers, but we drop loopy edges to make it a tree (Figure 4.6), because tree structures allow fast

inference while being powerful enough to build pruners. For *part pruners*, $\mathcal{G}$ has no spatial/appearance relations between parts.

4.4.1 Generalized model formulation

Our main approach is to learn compatibility functions for *pruners* and *parsers* by max-margin approaches. The parameters of these compatibility functions are learned by the appropriate loss functions depending on whether they are pruners or parsers. For pruners, we require the compatibility scores at the ground truth part configurations are not smaller than some pruning thresholds while they should be the highest scores compared to other part configurations for pruners. We learn the parameters by minimizing the empirical loss functions on a set of N labeled examples $\{\mathcal{I}^n\}, n = 1, .., N$. Each example is an image containing a person being marked up with each end points for 10 body parts. The examples of large parts are automatically extracted by the bounding boxes containing all parts involved. Given an example $\mathcal{I}$ and a configuration of parts $L = l_i, i = 1..M$, a generalized compatibility function is a summation of unary and pairwise potentials as follows:

$$C(L; \mathcal{I}; \mathcal{W}) = \sum_{i \in \mathcal{V}} w_i^\top \Phi_i(l_i; \mathcal{I}) + \sum_{i,j \in \mathcal{E}} w_{ij}^\top \Phi_{ij}(l_i, l_j; \mathcal{I}) \tag{4.1}$$

where $\mathcal{V}$ and $\mathcal{E}$ represents the vertices and edges in the graph structure respect to the each pruners and parsers. $\mathcal{W} = [w_i \ w_{ij}]$ are the parameters of the function to be learned. $\Phi_i(l_i; \mathcal{I})$ is the local evidence of part i. $\Phi_{ij}(l_i, l_j; \mathcal{I})$ is the spatial relations and/or appearance relations between part i and part j.

The compatibility functions have the same scoring characteristics for all models. They should score high on correct part states and low on unlikely part states. Moreover, the compatibility functions can reduce to a single

Concepts and Notations
primitive parts - rigid parts as in traditional definition. They are torso, head, left/right upper arms, left/right lower arms, left/right upper legs, left/right lower legs.
large parts - large chunk area of human body. They are whole body, torso-head, torso-larm (left arm), toro-rarm (right arm), torso-leg, larm (left arm), rarm (right arm), lleg (left leg), rleg (right leg).
$M = 20$ - number of body parts (both large parts and primitive parts).
p_i - body part i, where $i = 1, .., M$.
$p_{i,j}$ - poselet index j of part i, where $j = 1, .., M_i$.
$\mathcal{G} = (\mathcal{V}, \mathcal{E})$ - relational graph model where $\mathcal{V}$ - the set of *nodes* (each body part is a node in the graph) and $\mathcal{E}$ - the set of edges (each edge links two related nodes).
$\mathcal{L}$ - the current set of part states.
$\mathcal{L}_i$ - the current set of states of part i.
$l = \{l_i\}_{i=1}^{M}$ - a configuration of body parts.
W - learned paramters of a corresponding pruning model.

Table 4.1. *The table of definitions and notations used in the formulations of pruning and parsing models.*

unary terms in some cases as for part pruners because there are no pairwise relations in that models. Inference on the models usually involves finding the best structure (aka MAP assignment) in the search state space as follows:

$$L^* = \arg\max_{L \in \mathcal{L}} C(L; \mathcal{I}; \mathcal{W}) \tag{4.2}$$

For the tree structures, the exact solution can be found efficiently by dynamic programing while non-tree structures require an approximation such as loopy belief propagation.

4.5 Pruners

4.5.1 A cascade of structured pruners

Instead of building a single fine pruner that might suffer similar issues of high inference complexity as building a parser, we build a cascade of structured pruners inspired by Weiss *et al.* [64] and Sapp *et al.* [17]. Each level in the cascade, called *a structured pruner*, is a reasonably good and fast pruner to progressively filter unlikely part states using *max-marginal scores* and *data-specific thresholds*. The *max-marginal score* of a part at a given state is the max joint score with the best states of other parts. This joint score is more accurate in evaluating a part state because it considers the joint configuration of all body parts. Moreover, the threshold to prune states of a part in an example is computed based on a linear combination of the max-marginal score and the mean of marginal scores on that example [64]. This data-specific thresholds contrast with a single global threshold (one for each part) which is hard to tune.

Assume a structured pruner is represented by a relational model $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and uses the same compatibility function as in Equation 4.1. Given an image $\mathcal{I}$ and a configuration of parts $l = \{l_i\}_1^M$, the max-marginal score of part i at the state l_i is defined as follows:

$$C(l_i; \mathcal{I}; \mathcal{W}) = \max_{l' \in \mathcal{L}} \{C(l'; \mathcal{I}; \mathcal{W}) : l'_i = l_i\} \tag{4.3}$$

where $\mathcal{L}$ is the current state space. This max-marginal score corresponds to fixing part i at state l_i and searching for the best states of other parts. The joint score helps to recover state l_i when it has weak local evidence but good overall configuration arrangement. We also define the best MAP assignment

as:

$$C^*(\mathcal{I};\mathcal{W}) = \max_{l \in \mathcal{L}} \{C(l;\mathcal{I};\mathcal{W})\} \tag{4.4}$$

The max-marginal scores are used to decide to prune part states if they are below a threshold $t(\mathcal{I}) = t(\mathcal{I}, \mathcal{W}, \alpha)$. State l_i of part i is pruned if $C(l_i;\mathcal{I};\mathcal{W}) < t(\mathcal{I})$. As defined in [64], $t(\mathcal{I})$ is the convex linear combination of the MAP assignment and the mean max-marginal score:

$$t(\mathcal{I}, \mathcal{W}, \alpha) = \alpha C^*(\mathcal{I};\mathcal{W}) + (1-\alpha)\frac{1}{M}\sum_{i=1}^{M}\frac{1}{|\mathcal{L}_i|}\sum_{l \in \mathcal{L}_i} l_i \in \mathcal{L}_i C(l_i;\mathcal{I}) \tag{4.5}$$

where $\alpha \in [0, 1]$ is the parameter to control the pruning behavior which is the trade-off between the pruning efficiency and performance accuracy. The higher value α is, the more states will be pruned but we might loose more correct part states and vice versa. This convex formulation allows us to optimize the parameters $\mathcal{W}$ at a given α and then tune α to handle the balance between remaining states and pruned states. In practice, we fix α at *zero* during the training to learn the parameter $\mathcal{W}$. The α for the testing are chosen on the validation set. The training set and the validation set exchange their roles to train each level of pruner in the cascade. This procedure has been described in Sapp *et al.* [17] to avoid overfitting.

4.5.2 Learning structured pruners

We learn the parameter $\mathcal{W}$ by using a max-margin approach. To do so, we try to minimize the hinge loss of the incorrectly pruned part states over the

example set $X = \{(\mathcal{I}^n, l^n)\}_{n=1}^N$ as follows:

$$\min \lambda \frac{1}{2} \parallel \mathcal{W} \parallel^2 + \frac{1}{N} \sum_{n \in examples} \xi_n(l^n, \mathcal{I}^n; \mathcal{W}) \tag{4.6}$$

where $\xi_n(l^n, \mathcal{I}^n; \mathcal{W}) = \max\{0, 1 + t^n(\mathcal{I}^n, \mathcal{W}, \alpha) - C(l^n; \mathcal{I}^n; \mathcal{W})\}$ is the hinge loss to measure the margin between the scores of the MAP assignment and the ground truth. This essentially learns $\mathcal{W}$ to ensure the ground truth scores are above the thresholds at least a margin of 1. We apply the gradient descent method to update $\mathcal{W}$ from violated examples of having $\xi_n(l^n, \mathcal{I}^n; \mathcal{W}) > 0$. The update of $\mathcal{W}$ at iteration $t+1$ will be:

$$\mathcal{W}^{t+1} \leftarrow \mathcal{W}^t + \eta(-\lambda \mathcal{W}^t + \nabla_{\mathcal{W}^t}) \tag{4.7}$$

$$\nabla_{\mathcal{W}^t} = \Phi(l; \mathcal{I}) - \alpha\Phi(l^*; \mathcal{I}) - (1-\alpha)\frac{1}{M}\sum_i \frac{1}{\mid \mathcal{L}_i \mid} \sum_{l_i \in \mathcal{L}_i} \Phi(l_i; \mathcal{I}) \tag{4.8}$$

where η is the learning rate, $\Phi(l; \mathcal{I})$ is the feature vector at the ground truth configuration l of the example $\mathcal{I}$, $\Phi(l^*; \mathcal{I})$ is the feature vector of the MAP assignment found with respect to $\mathcal{W}^t$.

Sapp *et al.* [17] built their cascade of structured pruners by a coarse-to-fine resolutions of the search state. The initial resolution is 10x10x12 (the first two dimensions are 2D location and the third is the part orientation). They double the resolution after each level until they reach the actual search state resolution of 80x80x24. In contrast, we build structured pruners in the cascade at the actual search state resolution. Note that, while we use the same 2D location for the first two dimensions, our third dimension is the *poselet index* where it is also the part orientation for the primitive parts (see details in Section 4.7.1). Our cascade has three types of pruners. First,

part pruners have no edges in the relational structure model. Hence, the compatibility function has no pairwise potentials which makes the finding of MAP assignments extremely fast. For the unary potentials, the feature vector $\Phi(l_i, \mathcal{I})$ of part i and state location l_i is augmented by the poselet responses of its related large part at location l_i (e.g. the upper arms get support from the full arms). Though l_i is not the right location for the related part, its poselet responses are strong enough. These simple pruners apply to the early stages of large search spaces to prune down more than 75% states per part with little effort. The remaining set of states after this type of pruners are 2D part locations (x_i, y_i).

The second type of pruner is a tree-based structure model, called a *tree-based pruner*. The structure models now are hierarchical tree structures where we add edges between related parts (see Figure 4.6). Now, we can use the full formulation of the compatibility function of 4.1 with pairwise potentials. However, inference can be done fast by dynamic programing because this is a tree structure. After this type of pruner, the remaining set of states are still 2D part locations.

Finally, we build the third type of pruners of tree-based structure models, called *enhanced tree-based pruners*. These pruners work on pruning on the full state representation of (x_i, y_i, z_i), where (x_i, y_i) is a part mid-point and z_i is a part poselet index. For the primitive parts, the poselet indexes are the actual orientations. After these stages, the set of remaining states are small enough to run a human body parser to localize body parts.

4.5.3 Part pruners

For these pruners, the part state representation is 2D locations, which is the mid-point locations of parts. Given an example $\mathcal{I}$, a configuration of parts $l = \{l_i = (x_i, y_i)\}_{i=1}^{M}$, the compatibility function of Equation 4.1 will reduce to the summation of unary potentials because there are no edges between parts:

$$C_1(L; \mathcal{I}; \mathcal{W}) = \sum_{i \in \mathcal{V}} w_i^\top \Phi_i(l_i; \mathcal{I}) \tag{4.9}$$

where $\Phi_i(l_i; \mathcal{I})$ is now the feature vector at location l_i consisting of the poselet responses of part i and the support part of part i. E.g, the left arm is the support part of the left upper arm.

Because there are no pairwise relations, finding the MAP assignments according to Equation 4.9 can be done quickly by searching the best state for each part.

$$C^*(L; \mathcal{I}; \mathcal{W}) = \max_{l \in \mathcal{L}} \sum_{i \in \mathcal{V}} w_i^\top \Phi_i(l_i; \mathcal{I}) = \sum_{i \in \mathcal{V}} \max_{l_i \in \mathcal{L}_i} w_i^\top \Phi_i(l_i; \mathcal{I}) \tag{4.10}$$

where $\mathcal{L}_i$ is the current state space of part i.

The max-marginal score of part i at a state l_i now become as follows:

$$C_1(l_i; \mathcal{I}; \mathcal{W}) = w_i^\top \Phi_i(l_i; \mathcal{I}) + \sum_{j \in \mathcal{V} \setminus i} \max_{l_j \in \mathcal{L}_j} w_j^\top \Phi_j(l_j; \mathcal{I}) \tag{4.11}$$

Learning the parameters of this compatibility function can be done as in Section 4.5.2. However, we can do the update of parameters in Equation 4.8 in a batch scheme by the average of the gradients of all examples. This is feasible because solving the MAP score (Equation 4.10) and the max-

marginal score (Equation 4.11) requires just linear time in the part state space. The batch update scheme guarantees to find the optimal solution for the problem in Equation 4.6.

$$\mathcal{W}^{t+1} \leftarrow \mathcal{W}^t + \eta(-\lambda\mathcal{W}^t + \nabla_{\mathcal{W}^t}) \tag{4.12}$$

$$\nabla_{\mathcal{W}^t} = \frac{1}{N}\sum_{n=1}^{N}\left(\Phi(l^n;\mathcal{I}^n) - \alpha\Phi(l^*;\mathcal{I}^n) - (1-\alpha)\frac{1}{M}\sum_{i=1}^{M}\frac{1}{|\mathcal{L}_i|}\sum_{l_i\in\mathcal{L}_i}\Phi(l_i;\mathcal{I}^n)\right)$$

We build three levels of this type of part pruner. Our observation is that more levels are not very helpful because this model has simple structure. Therefore, if we use these pruners to prune too many states, we will wrongly remove good states (see table 4.2 for details of the reduction rates). After these three levels, more than 75% states are filtered (the reduction rate of large parts are even higher). The next tree-based pruning models are more complex and are capable of pruning more unlikely part states.

4.5.4 Tree-based pruners

We model this type of pruner using hierarchical tree structures. The model has $M = 20$ parts (Figure 4.6) where the root is at *torso-head* and the edges are links between related parts. This is a simplified structure of the loopy relational model used in parsers (Figure 4.8) by dropping edges in the loops to make it a tree structure. The tree models still have the efficiency of inference and the strength for a good pruner.

Given an example $\mathcal{I}$ and a configuration $l = \{l_i\}_{i=1}^{M}$ of parts in the current state space, we now can use the full formulation of the compatibility function of Equation 4.1. Note that a state of part i is still represented by $l_i = (x_i, y_i)$,

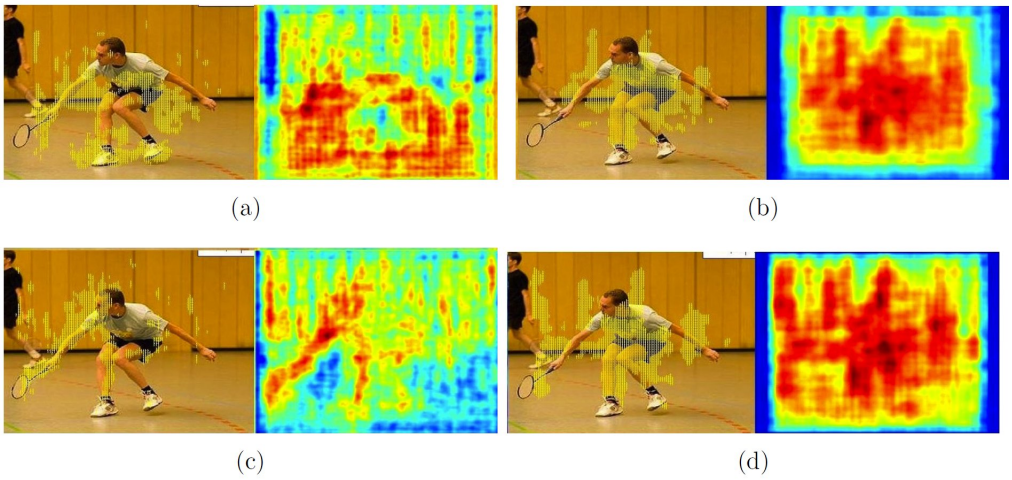

Figure 4.5. *An example of remaining states+heatmap by part pruners with support from large parts (top) vs. part pruners without support from big parts (bottom). Each figure shows the remaining states (left) and the heatmap of part state confidence (state confidence scores are visualized by colors. Red, yellow, cyan, blue are respect to high, medium and low confidence scores). Figures (a)+(c) are for right lower arm and figures (b)+(d) are for the right lower leg. Part pruners are able to prune away at least 75% unlikely states. The remaining states concentrate around the correct location while remaining states produced by pruners without support from large parts scattered all over the example. This evidence indicates that large parts are helpful in resolving ambiguity of small parts. Note that, the right lower arm pruner is augmented by the whole right arm part and the right lower leg pruner is augmented by the whole right leg part (best viewed in color).*

a 2D-location in the example.

We apply the procedure in Section 4.5.2 to learn the parameters of the tree-based pruners. We build four levels of tree-based pruners. The remaining states are 2D-state representation of part mid-points. As shown in the experiments (Section 4.8, there are about 300-500 remaining states for each part after these levels. We could stop at these levels and apply a human parser to find parsing configuration. However, finding a parse will involve searching over all part poselets (ranging from 8 poselets for the torso to 15 poselets for the half limbs). These are still large state spaces (approximately 2500-4000 states per part). Therefore, we extend tree-based pruners to work on the full representation of part states (2D locations and poselet indexes).

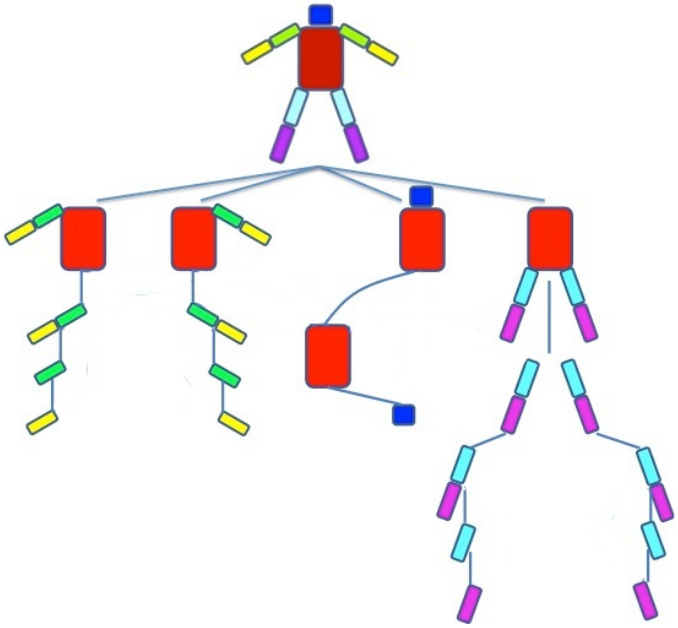

Figure 4.6. *A tree-based structure of hierarchical poselet pruning models of M=20 parts consists of both large parts and primitive parts. Each part is a node in the tree where the root is at the torso-head part. Blue edges are links between related parts. The tree structure has an advantage of fast inference by dynamic programing.*

4.5.5 Enhanced tree-based pruners

We now take the poselet index into the part state to make a full representation. Each state part i is a triple of $(x_i, y_i, z_i), (x_i, y_i)$ is the mid-point part location and z_i is the poselet index. For primitive parts (half limbs, torso, head) poselet indexes are the actual part orientations. They will be used to infer the endpoints of the final parsing result. Basically, we use the same tree structures for this type of pruners as in Figure 4.6.

We add part poselet priors and poselet co-occurence priors to the compatibility function of 4.1. Yang *et al.* [56] have demonstrated the usefulness of these priors for human parsing problems though their notion of parts are small parts (a patch around body joints). In our case, these priors are more important because our parts are large (e.g. an up-right poselet torso-head normally matches up with an upright torso).

Let $b_i(z_i)$ be the posetlet prior of poselet z_i of part i, and $b_{ij}(z_i, z_j)$ be the

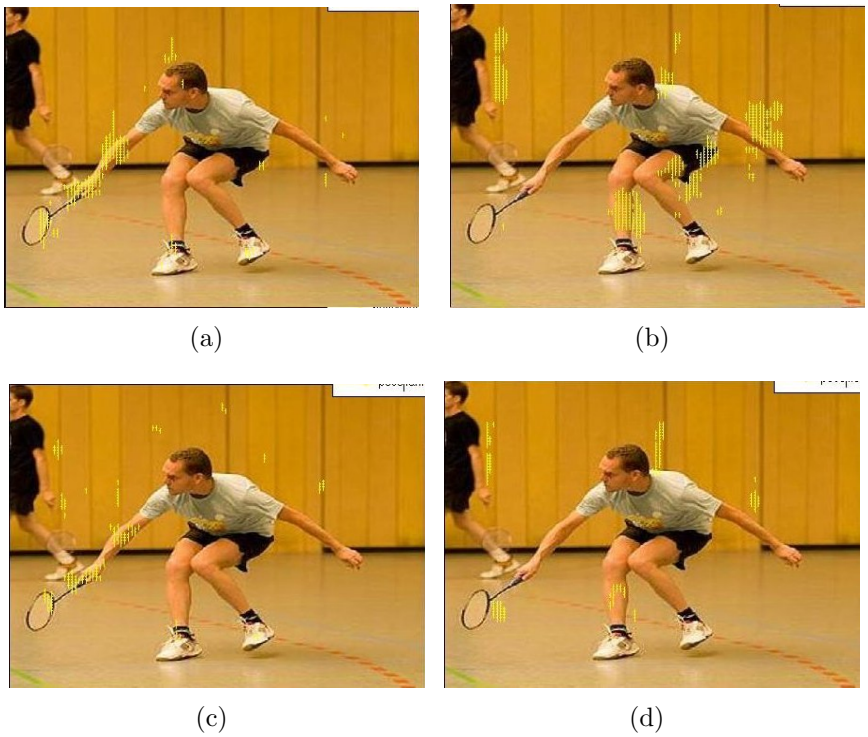

Figure 4.7. *These figures show the remaining states of left lower arm and left lower leg after the tree-based pruners (4.7(a) and 4.7(b)) and after the enhanced tree-based pruners (4.7(c) and 4.7(d)). Once part pruners have done, tree-based pruners are applied to keep takeing away unlikely part states (more than 95%) and leave a small number of 2D part states (around 300-500 2D locations per part). However, when adding one more dimension of the part poselet index to the search spaces (8 to 15 poselets per part), the number of states per part is still relatively large (2400-4000 actual states per part). We perform enhanced tree-based pruners to work on the full state representation to prune to about 500 states per part. This set of states is small enough to execute a parser.*

co-occurence prior of poselet z_i, z_j of part i and part j. Now, the compatibility function of Equation 4.1 is become as follow:

$$C_b(l; \mathcal{I}; \mathcal{W}_T) = B(l) + \sum_{i \in \mathcal{V}} w_i^\top \Phi_i(l_i; \mathcal{I}) + \sum_{i,j \in \mathcal{E}} w_{ij}^\top \Phi_{ij}(l_i, l_j; \mathcal{I}) \qquad (4.13)$$

$$B(l) = \sum_{i \in \mathcal{V}} b_i(z_i) + \sum_{i,j \in \mathcal{E}} b_{ij}(z_i, z_j) \qquad (4.14)$$

Level	torso	head	upper arm	lower arm	upper leg	lower leg
Part 1	55.0	51.8	54.8	52.6	55.7	54.2
Part 2	79.8	76.4	72.1	71.0	70.7	69.2
Part 3	88.8	79.0	75.6	75.0	76.5	77.9
Tree 1	91.4	92.8	89.7	88.1	90.5	91.3
Tree 2	95.8	94.5	92.9	91.8	93.4	94.1
Tree 3	97.1	95.6	94.7	94.2	95.1	95.3
Tree 4	97.5	95.8	95.1	95.0	95.3	95.4
E. tree 1	99.4	99.4	99.3	99.4	99.4	99.3
E. tree 2	99.6	99.6	99.6	99.6	99.6	99.6

Table 4.2. *This table shows reduction rates for primitive part after each level of the cascade of pruners. Part levels are part pruners and Tree levels are tree-based pruners and Enhanced tree levels (E. tree) are advanced tree-based pruners. Pruners are sequentially applied to prune on the current set of states. Each row shows the percentage of part states are reduced up to that pruner level. For advanced tree pruners we tune the thresholds such that about 99.6% of part states are removed after all these levels. This ensures the number of part states are small enough for the parser. Note that, the initial state resolution is* $100 * 100 * N_{poselets}$.

where $\mathcal{V}$ and $\mathcal{E}$ represents the vertices and edges in the graph shown in Figure 4.6, respectively, $l = \{l_i = (x_i, y_i, z_i)\}_{i=1}^{M}$ is a configuration of parts.

We also apply the procedure in Section 4.5.2 to learn the parameters for these pruners, except we use the compatibility function of Equation 4.13. We build three levels of this *enhanced tree-based pruners*. After these levels, we have only a small set of states (about 500 states per part). This set is small enough to build a parser using appearance models. Table 4.2 and 4.3 show the reduction rates after each level of the cascade and accuracy of the remaining set of states.

4.6 Parser

Given small numbers of remaining part states after pruning, we now can apply a body parser to localize parts. We use the hierarchical poselet models by Wang *et al.* [57] in which we represent the configuration of the human pose

Level	torso	head	upper arm	lower arm	upper leg	lower leg
Part 1	93.5	88.5	86.8	80.5	91.5	89.2
Part 2	91.8	82.7	82.5	76.8	87.0	86.1
Part 3	88.3	81.8	80.1	75.9	86.3	83.5
Tree 1	85.9	76.4	73.7	70.0	80.9	78.6
Tree 2	82.1	72.6	69.2	65.2	77.9	75.1
Tree 3	80.5	70.8	67.4	63.5	73.0	71.4
Tree 4	79.8	69.3	65.5	61.9	72.4	70.9
E. tree 1	77.2	62.5	58.3	55.6	68.4	65.8
E. tree 2	76.5	61.2	57.2	54.9	67.7	64.1

Table 4.3. *This table shows $PCP_{0.2}$ rates for oracle primitive part states after each level of the cascade of pruners at $PCP_{0.2}$ (E. trees are Enhanced trees). There is more pruning as you go down the table. Oracle states are best states chosen from the pools of the current remaining part states (see details in section 4.8.1). After applying pruners, torso gets 75.6% $PCP_{0.2}$ and lower arms get 54.9% $PCP_{0.2}$ (Note that, Sapp's pruner gets 54% $PCP_{0.2}$ for lower arm on Buffy dataset).*

using a 20-part model shown in Figure 4.8. The black edges in Figure 4.8 represents constraints between pairs of parts. Give a new image $\mathcal{I}$, we need to find the configuration l_i for the i-th part ($i = 1, 2, ..., 20$). Here $l_i = (x_i, y_i, z_i)$, where (x_i, y_i) corresponds to the 2D location in the image, and z_i indicates the poselets index for the i-th part. We measure the compatibilty between an image $\mathcal{I}$ and a pose configuration $L = (l_1, l_2, ..., l_{M=20})$ as the summation of unary and pairwise potentials:

$$C(L; \mathcal{I}; \mathcal{W}) = \sum_{i \in \mathcal{V}} w_i^\top \phi_i(l_i; \mathcal{I}) + \sum_{i,j \in \mathcal{E}} w_{ij}^\top \phi_{ij}(l_i, l_j) \tag{4.15}$$

where $\mathcal{V}$ and $\mathcal{E}$ represents the vertices and edges in the graph shown in Figure 4.8, respectively.

The remaining set of states left by the cascade of pruners are small about 500 states for each part. We now can exploit richer forms of pairwise potentials that depend on the image. We use the representative color symmetry

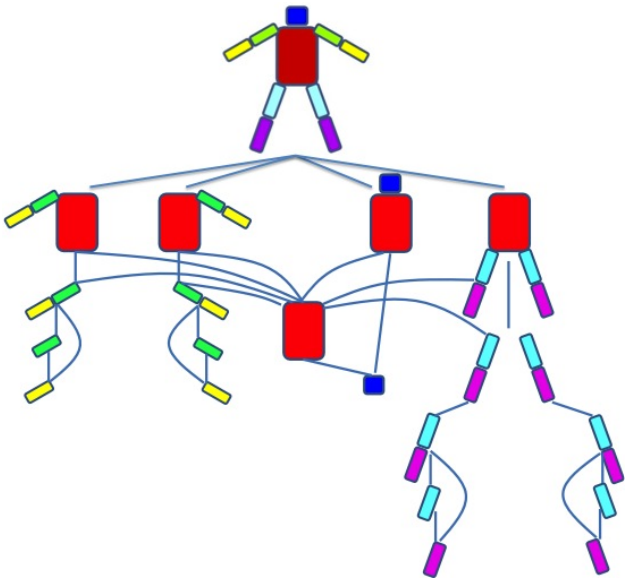

Figure 4.8. *This is the relational graph model of the part-based representation of human poses which is after the hierarchical poselet model of Wang et al. [57]. In total, there are 20 parts. Each part is a node in the graph. Blue edges are links between related parts in the model. We do not consider full relational models in order to reduce inference complexity, but the model is more sophisticated than the tree-structured kinematic part models (e.g. [28]) used in the literature.*

of left and right limbs for the pairwise appearance relations. Precomputing those pairwise potentials allows us to solve the MAP assignment of Equation 4.15 quickly.

4.7 Implementation

4.7.1 State space resolution

The 2D-location state space is on image grid of 100x100 (about 4 pixels for each grid cell size). This 2D resolution is the same in all levels of pruners. For enhanced poselet pruners, we expand to the full state resolution by adding the part poselet index dimension. Poselet dimension sizes vary between 8 to 15 depending on parts. For the primitive parts, poselet indexes indicate the actual part orientations. At the beginning there are 10,000 2D-states per part

(w.r.t the grid size 100x100). Part pruners prune and tree-based pruners can prune away more than 95% states leaving about 300-500 2D-states. We then expand the current set of 2D states to full resolution states by enumberating all poselet indexes. Then , the enhanced tree-based pruners are applied to filter to about at most 500 states per part. The parser now searches for the best structures in the remaining set of states.

4.7.2 Unary features

For all models (pruners and parsers), we use part poselet responses as unary features. Poselet responses are precomputed for all examples from pre-trained poselet templates (using SVM on HOG features). For part pruners, the unary features at each part are augmented with the poselet responses from supporting parts at the same part location. For examle, the augmenting unary features of the left upper arm part are poselet responses of the left arm. These additional unary features represent the context of left arm evidence to support the left upper arm.

An alternative way is to use original HOG features as unary features and directly learns weight parameters for unary features. This might improve the disrimination of the models because poselet responses can be considered as feature reduction which might loose some discrimination; however, we find that using poselet responses are good enough for pruners and parsers.

4.7.3 Pairwise features

For tree-based pruners and enhanced tree-based pruners, there are no appearance pairwise relation features. We only use pairwise spatial features between connect parts. We use binning scheme distances of between part

mid-points for better capturing spatial relations among them (after [28]). For enhanced tree-based pruners, we represent features for pairwise relations of part poselets by a vector of size *#poselet_part_i * #poselet_part_j*. The corresponding index in the vector of a pair of poselets z_i, z_j will be 1, others are zeros. For the parser, we use color appearance models for related primitive parts to capture the similarity of symmetric parts (e.g. limbs on the left are similar to limbs on the right). We precompute color histograms and use χ^2 distances between color histograms of related parts as appearance features. We could use more extensive appearance features but there will be more computational. Color symmetry relations are good enough for the purpose to capture the appearance compatibility between left/right limbs.

4.8 Experiments

4.8.1 Evaluation methodology

Evaluation of parsers: We follow standard PCP criteria (Percentage of Correctly estimated Part) of evaluation parsing accuracy [12]. An estimated part is correct if its two endpoints to the ground truth endpoints lie within 50% of the ground truth length.

Evaluation of pruners: A good pruner removes most states, without pruning the right answer. We evaluate our pruning by checking whether the pruned state space still contains a part within PCP 0.2 of the right answer (given by an oracle parser). The oracle parser will choose the best state for each part in the current state space. This essentially gives an upper bound performance of the parsing results on the pruned space. At each level of pruners, we evaluate the reduction rate for primitive parts up to that level

(Table 4.3) and the corresponding $PCP_{0.2}$ by an oracle parser (Tabel 4.2).

4.8.2 Datasets

We run experiments on a challenging UIUC Sports by [57] which have large pose variations. This dataset is an extension of UIUC People ([15]) by collecting more sport images from Internet. There are totally 1299 examples of more than 20 sport categories including: badmintons, acrobatics, cycling, American football, croquet, hockey, figure skating, soccer, golf, horseback riding, rugby, etc. We randomly devide into halves for training (650 examples) and for testing (649 examples) as in [57]. Each example is annotated with 14 body joints (then being converted into 10 body parts, each is marked up by two end points). Table 4.9 shows some typical examples of the dataset.

We will show the benefits of pruning search space through the parsing results. Our parsing models are in the same approach of [57], except that we can use appearance models.

4.8.3 Experiments of our cascade of pruners

We demonstrate the effectiveness of our cascade of pruners on UIUC Sport dataset. We can reduce 99.6% of part states (leaving about 300-500 states per part) while still achieving high $PCP_{0.2}$ rate. At a given level, we compute the $PCP_{0.2}$ rate for the best states. In particular, table 4.2 after each level in the cascade is applied, and table 4.3 shows the corresponding $PCP_{0.2}$ rate for that level of pruner. For example, *part pruners* can prune away at least 75% of 2D states and *tree-based pruners* then prune up to 95% states. Finally, *advanced tree-based pruners* will perform to prune on full resolution states to more than or equal to 99.6%. In the final pruned space, oracle lower arms

Figure 4.9. *Examples of the dataset UIUC Sport collected from the Internet by [57]. The dataset contains examples of more than 20 sport categories which depict a wide range of poses. Each example is annotated with 10 parts (two end points for each part). There are 1299 examples devided into halves for for training and for testing).*

Figure 4.10. Some typical parsing results on the dataset UIUC Sports by our parser. Note that, the parser is trained and test on the pruned space of examples. This qualitative results show that the pruners still retain good states for the parser to produce good outputs.

still achieve 54.9% and oracle torso achieves 76.5% at $PCP_{0.2}$. Torso can perform much better than other parts because it is big and more stable to localize than other parts.

4.8.4 Experiments of human parsing

We compare our parsing results with other state of the art methods on UIUC Sport dataset. In table 4.4, we show that our results outperform [57] for all body parts. This is an evidence of the benefit of using appearance models on the pruned space. Our parser also performs better on most of body parts comparing to [28] and [39]. Figure 4.10 shows some typical parsing examples by our method.

Method	Torso	upper leg	lower leg	upper arm	lower arm	head
[28]	28.7	7.3	19.2	7.5	20.6	12.9
[39]	71.5	43.7	30.9	28.8	16.3	63.3
[57]	75.3	49.2	39.5	25.2	11.2	47.5
Our method	82.0	52.6	45.5	30.6	12.5	49.0

Table 4.4. *The comparisons of parsing results of our method vs other methods on UIUC Sport dataset. The percentage of correctly estimated parts (PCP) over primitive human body parts (upper/lower legs, upper/lower arms are average of two values of left and right parts). Note that the improvement of our method vs. [57] shows the benefits of using appearance models on pruning spaces.*

4.9 Conclusions

We have shown in this chapter that a good pruning strategy can make a hierarchical poselet parser feasible to use complex appearance models. We have built a cascade of poselet pruners using both large parts and primitive parts. In the pruners, large parts prove to be useful context to support small parts. There are 3 types of pruners in the cascade: simple part pruners, tree-based pruners and enhanced tree-based pruners. The pruner cascade effectively prunes away more than 99.6% part states to about 500 states per part. We have demonstrated an improvement of the hierarchical poselet parser on the pruned set using color appearance models compared to [57] and two other state-of-the-art methods on a challenging UIUC Sport dataset.

CHAPTER 5

CONCLUSIONS

We have demonstrated in this thesis that structure prediction is an attractive method for the problems of people detection and people parsing in images (and videos). For detection, we have shown that using structure prediction to estimate body configuration helps to improve the accuracy of detecting people in images. We have also demonstrated quantitatively that a full relational model of the body performs better at upper body parsing than the standard tree model, despite the need to adopt approximate inference and learning procedures. We have presented a method of building a cascade of pruners to reduce the search space of finding the best structures in parsing. The method builds a pruner using hierarchical poselets of both large parts and primitive parts. Finally, we have shown a significant improvement of the hierarchical poselet parser on the pruned set of part states where we can use complex appearance models.

Our contributions are as follows:

1. We have first developed a structure prediction method applying to the problem of people detection.

2. We have first demonstrated quantitatively that full relational model of the body performs better at upper body parsing than the standard tree model, despite the need to adopt approximate inference and learning procedures.

3. We have first developed poselet pruners using both large parts and primitive parts to build a cascade of pruners to prune the search space. The recuded search space makes the hierarchical poselet parser feasible to use complex appearance models.

Our future directions are as follows:

1. Improving appearance models: We believe better appearance models should help better localize arms and legs. Though clothes and skin textures vary significantly across different subjects, they tend to be consistent within an image and differs from background. We are attempting to better learn the consistency of body segments which takes into account the surrounding background texture. These models should be tractable for highly relational models where inference is expensively computational.

2. Improving pruning performance: better pruning methods are very important where leaving as few states per part as possible while keeping high quality of part states in the remaining set. This benefits inference in a full relational model because inference will be fast in a small set of part states. An alternative is to improve inference to be faster and more accurate.

3. Extending to tracking people in videos: Jointly tracking and parsing human subjects in videos are important for human activity recognition. In short-term tracking, people rarely change their appearance (i.e. clothes), and good body part appearance models would help to determine if subjects in different frames are the same person. We believe that our approach is applicable for this task.

REFERENCES

[1] C. Papageorgiou and T. Poggio, "A trainable system for object detection," *Int. J. Computer Vision*, vol. 38, no. 1, pp. 15–33, June 2000.

[2] C. Papageorgiou and T. Poggio, "A pattern classification approach to dynamical object detection," in *ICCV*, 1999, pp. 1223–1228.

[3] L. Zhao and C. Thorpe, "Stereo- and neural network-based pedestrian detection," *Intelligent Transportation Systems*, vol. 1, no. 3, pp. 148–154, September 2000.

[4] D. Gavrila, "Pedestrian detection from a moving vehicle," in *ECCV*, 2000, pp. II: 37–49.

[5] M. Dimitrijevic, V. Lepetit, and P. Fua, "Human body pose recognition using spatio-temporal templates," in *ICCV workshop on Modeling People and Human Interaction*, 2005.

[6] Y. Wu, T. Yu, and G. Hua, "A statistical field model for pedestrian detection," in *CVPR*, 2005, pp. I: 1023–1030.

[7] P. Viola, M. Jones, and D. Snow, "Detecting pedestrians using patterns of motion and appearance," *Int. J. Computer Vision*, vol. 63, no. 2, pp. 153–161, July 2005.

[8] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *CVPR*, 2005, pp. I: 886–893.

[9] S. Ioffe and D. Forsyth, "Finding people by sampling," in *ICCV*, 1999, pp. 1092–1097.

[10] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient matching of pictorial structures," in *CVPR*, 2000.

[11] G. Mori, X. Ren, A. Efros, and J. Malik, "Recovering human body configuration: Combining segmentation and recognition," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2, 2004, pp. 326–333.

[12] V. Ferrari, M. Marín-Jiménez, and A. Zisserman, "Progressive search space reduction for human pose estimation," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2008.

[13] M. Eichner and V. Ferrari, "Better appearance models for pictorial structures," in *IEEE International Conference on Computer Vision*, 2009.

[14] P. F. Felzenszwalb and D. P. Huttenlocher, "Pictorial structures for object recognition," *International Journal of Computer Vision*, vol. 61, no. 1, pp. 55–79, January 2005.

[15] D. Tran and D. Forsyth, "Improved human parsing with a full relational model," in *European Conference on Computer Vision*, 2010.

[16] P. F. Felzenszwalb, R. B. Girshick, and D. McAllester, "Cascade object detection with deformable part models," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2010.

[17] B. Sapp, A. Toshev, and B. Taskar, "Cascaded models for articulated pose estimation," in *European Conference on Computer Vision*, 2010.

[18] D. Gavrila, "Sensor-based pedestrian protection," *Intelligent Transportation Systems*, pp. 77–81, 2001.

[19] D. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Computer Vision*, vol. 60, no. 2, pp. 91–110, November 2004.

[20] A. Mohan, C. Papageorgiou, and T. Poggio, "Example-based object detection in images by components," *PAMI*, vol. 23, no. 4, pp. 349–361, April 2001.

[21] Y. Ke and R. Sukthankar, "Pca-sift: a more distinctive representation for local image descriptors," in *CVPR*, 2004, pp. II: 506–513.

[22] K. Mikolajczyk and C. Schmid, "A performance evaluation of local descriptors," *PAMI*, 2004, accepted.

[23] S. Belongie, J. Malik, and J. Puzicha, "Shape matching and object recognition using shape contexts," *PAMI*, vol. 24, no. 4, pp. 509–522, April 2002.

[24] D. Forsyth, O.Arikan, L. Ikemoto, J. O'Brien, and D. Ramanan, "Computational studies in human motion 1: Tracking and animation," *Foundations and Trends in Computer Vision*, 2006, in press.

[25] M. P. Kumar, P. H. S. Torr, and A. Zisserman, "Extending pictorial structures for object recognition," in *Proceedings of the British Machine Vision Conference*, 2004.

[26] D. Ramanan, D. Forsyth, and A. Zisserman, "Strike a pose: Tracking people by finding stylized poses," in *CVPR*, 2005.

[27] D. Ramanan and D. Forsyth, "Using temporal coherence to build models of animals," in *Proc. ICCV*, 2003.

[28] D. Ramanan, "Learning to parse images of articulated bodies," in *Advances in Neural Information Processing Systems*, vol. 19, 2006, pp. 1129–1136.

[29] R. Ronfard, C. Schmid, and B. Triggs, "Learning to parse pictures of people," in *ECCV*, 2002, p. IV: 700 ff.

[30] K. Mikolajczyk, C. Schmid, and A. Zisserman, "Human detection based on a probabilistic assembly of robust part detectors," in *ECCV*, 2004, pp. Vol I: 69–82.

[31] A. Micilotta, E. Ong, and R. Bowden, "Detection and tracking of humans by probabilistic body part assembly," in *British Machine Vision Conference*, vol. 1, 2005, pp. 429–438.

[32] B. Leibe, E. Seemann, and B. Schiele, "Pedestrian detection in crowded scenes," in *CVPR*, 2005, pp. I: 878–885.

[33] B. Taskar, "Learning structured prediction models: A large margin approach," Ph.D. dissertation, Stanford University, 2004.

[34] B. Taskar, S. Lacoste-Julien, and M. Jordan, "Structured prediction via the extragradient method," in *Neural Information Processing Systems Conference*, 2005.

[35] N. Ratliff, J. A. Bagnell, and M. Zinkevich, "Subgradient methods for maximum margin structured learning," in *ICML 2006 Workshop on Learning in Structured Output Spaces*, 2006.

[36] N. Shor, *Minimization Methods for Non-Differentiable Functions and Applications*. Springer, 1985.

[37] C.-C. Chang and C.-J. Lin, *LIBSVM: a library for support vector machines*, 2001, software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.

[38] P. Sabzmeydani and G. Mori, "Detecting pedestrians by learning shapelet features," in *CVPR*, 2007.

[39] M. Andriluka, S. Roth, and B. Schiele, "Pictorial structures revisited: People detection and articulated pose estimation," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2009.

[40] X. Ren, A. Berg, and J. Malik, "Recovering human body configurations using pairwise constraints between parts," in *IEEE International Conference on Computer Vision*, vol. 1, 2005, pp. 824–831.

[41] G. Mori and J. Malik, "Estimating human body configurations using shape context matching," in *European Conf. Computer Vision*, 2002.

[42] Y. Song, X. Feng, and P. Perona, "Towards detection of human motion," in *CVPR*, 2000, pp. 810–17.

[43] D. Ramanan, D. Forsyth, and K. Barnard, "Building models of animals from video," *PAMI*, vol. 28, no. 8, pp. 1319 – 1334, 2006.

[44] R. Ronfard, C. Schmid, and B. Triggs, "Learning to parse pictures of people," in *ECCV02*, 2002, p. IV: 700 ff.

[45] S. Johnson and M. Everingham, "Combining discriminative appearance and segmentation cues for articulated human pose estimation," in *MLVMA09*, 2009.

[46] S. Ioffe and D. Forsyth, "Human tracking with mixtures of trees," in *ICCV*, 2001. [Online]. Available: citeseer.nj.nec.com/ioffe01human.html pp. 690–695.

[47] H. Jiang and R. Martin, "Global pose estimation using non-tree models," in *CVPR*, 2008.

[48] H. Jiang, "Human pose estimation using consistent max-covering," in *ICCV*, 2009.

[49] D. Tran and D. Forsyth, "Configuration estimates improve pedestrian finding," in *Proc. NIPS*, 2007.

[50] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun, "Large margin methods for structured and interdependent output variables," *Journal of Machine Learning Research (JMLR)*, vol. 6, pp. 1453–1484, 2005.

[51] P. F. Felzenszwalb, D. A. McAllester, and D. Ramanan, "A discriminatively trained, multiscale, deformable part model," in *CVPR*, 2008.

[52] E. Shechtman and M. Irani, "Matching local self-similarities across images and videos," in *CVPR*, 2007.

[53] J. Platt, "Probabilities for sv machines," in *Proc. NIPS*, 1999.

[54] A. Farhadi, I. Endres, D. Hoiem, and D. Forsyth, "Learning to describe objects," in *CVPR*, 2009.

[55] M. Varma and A. Zisserman, "A statistical approach to texture classification from single images," *Int. J. Computer Vision*, vol. 62, no. 1-2, pp. 61–81, 2005.

[56] Y. Yang and D. Ramanan, "Articulated pose estimation using flexible mixtures of parts," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2011.

[57] Y. Wang, D. Tran, and Z. Liao, "Learning hierarchical poselets for human parsing," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2011.

[58] L. Bourdev and J. Malik, "Poselets: Body part detectors training using 3d human pose annotations," in *IEEE International Conference on Computer Vision*, 2009.

[59] S. X. Ju, M. J. Black, and Y. Yaccob, "Cardboard people: A parameterized model of articulated image motion," in *International Conference on Automatic Face and Gesture Recognition*, 1996, pp. 38–44.

[60] H. Jiang and D. R. Martin, "Globel pose estimation using non-tree models," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2008.

[61] T.-P. Tian and S. Sclaroff, "Fast globally optimal 2d human detection with loopy graph models," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2010.

[62] Y. Wang and G. Mori, "Multiple tree models for occlusion and spatial constraints in human pose estimation," in *European Conference on Computer Vision*, 2008.

[63] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2001.

[64] D. Weiss and B. Taskar, "Structured prediction cascades," in *Proc. AISTATS*, 2010.

[65] L. Bourdev, S. Maji, T. Brox, and J. Malik, "Detecting people using mutually consistent poselet activations," in *European Conference on Computer Vision*, 2010.