Articulated Human Detection with Flexible Mixtures-of-Parts

Yi Yang, *Member, IEEE,* and Deva Ramanan, *Member, IEEE*

Abstract—We describe a method for articulated human detection and human pose estimation in static images based on a new representation of deformable part models. Rather than modeling articulation using a family of warped (rotated and foreshortened) templates, we use a mixture of small, non-oriented parts. We describe a general, flexible mixture model that jointly captures spatial relations between part locations and co-occurrence relations between part mixtures, augmenting standard pictorial structure models that encode just spatial relations. Our models have several notable properties: (1) they efficiently model articulation by sharing computation across similar warps (2) they efficiently model an exponentially-large set of global mixtures through composition of local mixtures and (3) they capture the dependency of global geometry on local appearance (parts look different at different locations). When relations are tree-structured, our models can be efficiently optimized with dynamic programming. We learn all parameters, including local appearances, spatial relations, and co-occurrence relations (which encode local rigidity) with a structured SVM solver. Because our model is efficient enough to be used as a detector that searches over scales and image locations, we introduce novel criteria for evaluating pose estimation and human detection, both separately and jointly. We show that currently-used evaluation criteria may conflate these two issues. Most previous approaches model limbs with rigid and articulated templates that are trained independently of each other, while we present an extensive diagnostic evaluation that suggests that flexible structure and joint training are crucial for strong performance. We present experimental results on standard benchmarks that suggest our approach is the state-of-the-art system for pose estimation, improving past work on the challenging Parse and Buffy datasets, while being orders of magnitude faster.

Index Terms—Pose Estimation, Object Detection, Articulated Shapes, Deformable Part Models

1 Introduction

Articulated pose estimation is a fundamental task in computer vision. A working technology would immediately impact many key vision tasks such as image understanding and activity recognition. An influential approach is the pictorial structure framework [1], [2] which decomposes the appearance of objects into local part templates, together with geometric constraints on pairs of parts, often visualized as springs. When parts are parameterized by pixel location and orientation, the resulting structure can model articulation. This has been the dominant approach for human pose estimation. In contrast, traditional models for object recognition use parts parameterized solely by locations, which simplifies both inference and learning. Such models have been shown to be very successful for object detection [3], [4]. In this work, we introduce a novel, unified representation for both models that produces state-of-the-art results for the tasks of detecting articulated people and estimating their poses.

Representations for articulated pose: Full-body pose estimation is difficult because of the many degrees of freedom to be estimated. Moreover, limbs vary greatly in appearance due to changes in clothing and body shape, as well as changes in viewpoint manifested in in-plane rotations and foreshortening. These difficulties complicate inference as one must typically search images with a large number of warped (rotated and foreshortened) templates. We address these problems by introducing a simple representation for modeling a family of warped templates: a mixture of pictorial structures with small, non-oriented parts (Fig. 1).

Our approach is significantly faster than an articulated model because we exploit dynamic pro-

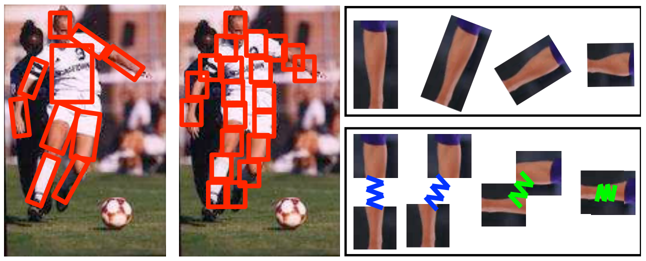


Fig. 1: Our flexible mixture-of-parts model (**middle**) differs from classic approaches (**left**) that model articulation by warping a single template to different orientation and foreshortening states (**top right**). Instead, we approximate small warps by translating patches connected with a spring (**bottom right**). For a large warp, we use a different set of patches and a different spring. Hence, our model captures the dependence of local part appearance on geometry (i.e. elbows in different spatial arrangements look different).

• *Y. Yang and D. Ramanan are with the Department of Computer Science, University of California at Irvine, Irvine, CA 92697. E-mail: {yyang8, dramanan}@ics.uci.edu*

gramming to share computation across similar warps during matching. Our approach can also outperform articulated models because we capture the effect of global geometry on local appearance; an elbow looks different when positioned above the head or beside the torso. One reason for this is that elbows rotate and foreshorten. However, appearance changes also arise from other geometric factors such as partial occlusions and interactions with clothing. Our models capture such often ignored dependencies because local mixtures depend on the spatial arrangement of parts.

Representations for objects: Part models are also common in general object recognition. Because translating parts do not deform too much in practice, one often resorts to global mixture models to capture large appearance changes [4]. Rather, we compose together local part mixtures to model an *exponentially-large* set of global mixtures. Not all such combinations are equally likely; we learn a prior over what local mixtures can co-occur. This allows our model to learn notions of local-rigidity; for example, two parts on the same rigid limb must co-occur with consistent oriented edge structure. An open challenge is that of learning such complex object representations from data. We find that supervision is a key ingredient for learning structured relational models; one can use limb orientation as a supervisory signal to annotate part mixture labels in training data.

Efficiency: For computational reasons, most prior work on pose estimation assumes that people are pre-localized with a detector that provides the rough pixel location and scale of each person. Our model is fast enough to search over all locations and scales, and so we both detect and estimate human poses without any pre-processing. Our model requires roughly 1 second to process a typical benchmark image, allowing for the possibility of real-time performance with further speedups (such as cascaded [5] or parallelized implementations). We have released open-source code [6] which appears to be in use within the community.

Evaluation: The most popular evaluation criteria for pose estimation is the percentage of correctly localized parts (PCP) criteria introduced in [7]. Though this criteria was crucial and influential in spurring quantitative evaluation, it was somewhat ambiguously specified in [7], resulting in possibly conflicting implementations.

One point of confusion is that PCP, as original specified, assume humans are pre-detected on test images. This assumption may be unrealistic because it is hard to build detectors for highly articulated poses (for the same reason it is hard to correctly estimate their configurations). Another point of confusion is that there appears to be two interpretations of the definition of correctly localized parts criteria introduced in [7]. We will give a detailed description of these issues in Section 7.

Unfortunately these subtle confusions lead to significant differences in terms of final performance results. We show that that there may exist a *negative* correlation between body-part detection accuracy and PCP as implemented in the toolkit released by [8]. We then introduce new evaluation criteria for pose estimation and body-part detection that are self-consistent. We evaluate all different types of PCP criteria and our new criteria on two standard benchmark datasets [7], [9].

Overview: An earlier version of this manuscript appeared in [10]. This version includes a slightly refined model, additional diagnostic experiments, and an in-depth discussion of evaluation criteria. After discussing related work, we motivate our approach in Section 3, describe our model in Section 4, describe algorithms for inference in Section 5, and describe methods for learning parameters from training data in Section 6. We then show experimental results and diagnostic experiments on our benchmark datasets in Section 7.

2 Related work

Pose estimation has typically been addressed in the video domain, dating back to classic model-based approaches of O'Rourke and Badler [11], Hogg [12], Rohr [13]. Recent work has examined the problem for static images, assuming that such techniques will be needed to initialize video-based articulated trackers. We refer the reader to the recent survey article [14] for a full review of contemporary approaches.

Spatial structure: One area of research is the encoding of spatial structure, often described through the formalism of probabilistic graphical models. Tree-structured graphical models allow for efficient inference [1], [15], but are plagued by double-counting; given a parent torso, two legs are localized independently and often respond to the same image region. Loopy constraints address this limitation but require approximate inference strategies such as sampling [1], [16], [17], loopy belief propagation [18], or iterative approximations [19]. Recent work has suggested that branch and bound algorithms with tree-based lower bounds can globally solve such problems [20], [21]. Another approach to eliminating double-counting is the use of stronger pose priors [22]. However, such methods may overfit to the statistics of a particular dataset, as warned by [18], [23]. We find that simple tree models, when trained contextually with part models in a discriminative framework, are fairly effective.

Learning: An alternate family of techniques has explored the trade-off between generative and discriminative models. Approaches include conditional random fields [24], margin-based learning [25], and boosted detectors [26], [27], [21]. Most previous approaches train limb detectors independently, in part due to the computational burdens of inference. Our representation is efficient enough to be learned jointly; we show in our experimental results that joint learning is crucial for accurate performance. A small part trained by itself is too weak to provide a strong signal, but a collection of patches trained contextually are rather discriminative.

Image features: An important issue for computer vision tasks is feature description. Past work has explored the use of superpixels [28], contours [26], [29], [30], foreground/background color models [9], [7], edge-based descriptors [31], [32], and gradient descriptors [27], [33]. We use oriented gradient descriptors [34] that allow for fast computation, but our approach could be combined with other descriptors. Recent work has integrated our models with steerable image descriptors for highly efficient pose estimation [35].

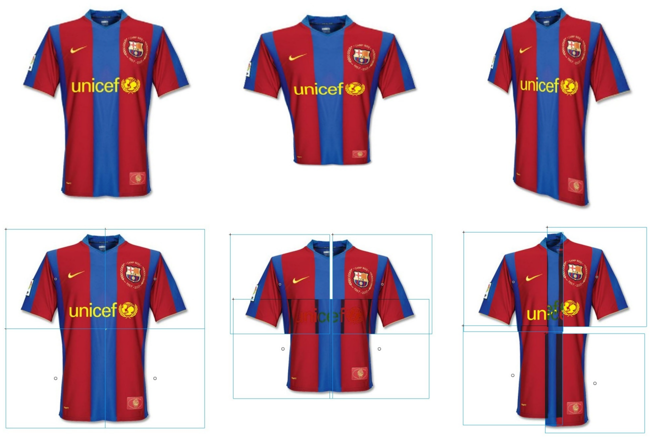
Large vs small parts: In recent history, researchers have begun exploring large-scale, non-articulated parts that span multiple limbs on the body ('Poselets') [3]. Such models were originally developed for human detection, but [36] extends them to pose estimation. Large-scale parts can be integrated into a hierarchical, coarse-to-fine representation [37], [38]. The underlying intuition behind such approaches stems from the observation that it is hard to build accurate limb detectors because they are nondescript in appearance (i.e., limbs are defined by parallel lines that may commonly occur in clutter). This motivates the use of larger parts with more context. We demonstrate that *jointly* training small parts has the same contextual effect.

Object detection: In terms of object detection, our work is most similar to pictorial structure models that reason about mixtures of parts [39], [1], [4], [15]. We show that our model generalizes such representations in Section 4.1. Our local mixture model can also be seen as an AND-OR grammar, where a pose is derived by AND'ing across all parts and OR'ing across all local mixtures [4], [40].

3 Motivation

Our model is an approximation for capturing a continuous family of warps. The classic approach of using a finite set of articulated templates is also an approximation. In this section, we present a straightforward theoretical analysis of both. For simplicity, we restrict ourselves to affine warps, though a similar derivation holds for any smooth warping function, including perspective warps (Fig. 2).

Let us write x for a 2D pixel position in a template, and $w(x) = (I + \Delta A)x + b$ for its new position under a small affine warp $A = I + \Delta A$ and any translation b. We use ΔA to parameterize the deviation of the warp from an identity warp. Define $s(x) = w(x) - x$ to be the shift of position x. The shift of a nearby position



(a) Original (b) Foreshortening (c) Out-of-plane

Fig. 2: We show that 4 small, translating parts can approximate non-affine (e.g., perspective) warps.

$x + \Delta x$ can be written as

$$\begin{aligned} s(x + \Delta x) &= w(x + \Delta x) - (x + \Delta x) \\ &= (I + \Delta A)(x + \Delta x) + b - x - \Delta x \\ &= s(x) + \Delta A \Delta x \end{aligned}$$

Both pixels x and $x + \Delta x$ shift by the same amount (and can be modeled as a single part) if the product $\Delta A \Delta x$ is small, which is true if ΔA has small determinant or Δx has small norm. Classic articulated models use a large family of discretized articulations, where each discrete template only needs to explain a small range of rotations and foreshortening (e.g., a small-determinant ΔA). We take the opposite approach, making Δx small by using small parts. Since we want the norm of Δx to be small, this suggests that circular parts would work best, but we use square parts as a discrete approximation. In the extreme case, one could define a set of single-pixel parts. Such a representation is indeed the most flexible, but becomes difficult to train given our learning formulation described below.

4 Model

Let us write I for an image, $l_i = (x, y)$ for the pixel location of part i and t_i for the mixture component of part i. We write $i \in \{1, \dots K\}$, $l_i \in \{1, \dots L\}$ and $t_i \in \{1, \dots T\}$. We call t_i the "type" of part i. Our motivating examples of types include orientations of a part (e.g., a vertical versus horizontally oriented hand), but types may span out-of-plane rotations (front-view head versus side-view head) or even semantic classes (an open versus closed hand). For notational convenience, we define the lack of subscript to indicate a set spanned by that subscript (e.g., $t = \{t_1, \dots t_K\}$). For simplicity, we define our model at a fixed scale; at test-time we detect people of different sizes by searching over an image pyramid.

Co-occurrence model: To score of a configuration of parts, we first define a compatibility function for part types that factors into a sum of local and pairwise scores:

$$S(t) = \sum_{i \in V} b_i^{t_i} + \sum_{ij \in E} b_{ij}^{t_i,t_j} \tag{1}$$

The parameter $b_i^{t_i}$ favors particular type assignments for part i, while the pairwise parameter $b_{ij}^{t_i,t_j}$ favors particular co-occurrences of part types. For example, if part types correspond to orientations and part i and j are on the same rigid limb, then $b_{ij}^{t_i,t_j}$ would favor consistent orientation assignments. Specifically, $b_{ij}^{t_i,t_j}$ should be a large positive number for consistent orientations t_i and t_j, and a large negative number for inconsistent orientations t_i and t_j.

Rigidity: We write $G = (V, E)$ for a (tree-structured) K-node relational graph whose edges specify which pairs of parts are constrained to have consistent relations. Such a graph can still encode relations between distant parts through transitivity. For example, our model can force a collection of parts to share the same orientation, so long as the parts form a connected *subtree* of $G = (V, E)$. We use this property to model multiple parts on the torso. Since co-occurrence parameters are learned, our model learns which collections of parts should be rigid.

We can now write the full score associated with a configuration of part types and positions:

$$S(I, l, t) = S(t) + \sum_{i \in V} w_i^{t_i} \cdot \phi(I, l_i) + \sum_{ij \in E} w_{ij}^{t_i,t_j} \cdot \psi(l_i - l_j) \tag{2}$$

where $\phi(I, l_i)$ is a feature vector (e.g., HOG descriptor [34]) extracted from pixel location l_i in image I. We write $\psi(l_i - l_j) = \begin{bmatrix} dx & dx^2 & dy & dy^2 \end{bmatrix}^T$, where $dx = x_i - x_j$ and $dy = y_i - y_j$, the relative location of part i with respect to j. Notably, this relative location is defined with respect to the pixel grid and not the orientation of part i (as in classic articulated pictorial structures [1]).

Appearance model: The first sum in (2) is an appearance model that computes the local score of placing a template $w_i^{t_i}$ for part i, tuned for type t_i, at location l_i.

Deformation model: The second term can be interpreted as a "switching" spring model that controls the relative placement of part i and j by switching between a collection of springs. Each spring is tailored for a particular pair of types (t_i, t_j), and is parameterized by its rest location and rigidity, which are encoded by $w_{ij}^{t_i,t_j}$. Our switching spring model encodes the dependence of local appearance on geometry, since different pairs of local mixtures are constrained to use different springs. Together with the co-occurrence term, it specifies an image-independent "prior" over part locations and types.

4.1 Special cases

We now describe various special cases of our model. The first three correspond to special cases that have previously occurred in the literature, while the last refers to a special case we implement in our experiments.

Stretchable human models: [41] describes a human part model that consists of a single part at each joint. This is equivalent to our model with $K = 14$ parts, each with a single mixture $T = 1$. Similar to us, [41] argue that a joint-centric representation efficiently captures foreshortening and articulation effects. However, our local mixture models (for $T > 1$) also capture the dependence of global geometry on local appearance; elbows look different when positioned above the head or beside the torso. We compare to such a model in our diagnostic experiments.

Semantic part models: [39] argues that part appearances should capture semantic classes and not visual classes; this can be done with a type model. Consider a face model with eye and mouth parts. One may want to model different types of eyes (open and closed) and mouths (smiling and frowning). The spatial relationship between the two does not likely depend on their type, but open eyes may tend to co-occur with smiling mouths. This can be obtained as a special case of our model by using a single spring for all types of a particular pair of parts:

$$w_{ij}^{t_i,t_j} = w_{ij} \tag{3}$$

Mixtures of deformable parts: [4] define a mixture of models, where each model is a star-based pictorial structure. This can be achieved by restricting the co-occurrence model to allow for only globally-consistent types:

$$b_{ij}^{t_i,t_j} = \begin{cases} 0 & \text{if} \quad t_i = t_j \\ -\infty & \text{otherwise} \end{cases} \tag{4}$$

Articulation: In our experiments, we explore a simplified version of (2) with a reduced set of springs:

$$w_{ij}^{t_i,t_j} = w_{ij}^{t_i} \tag{5}$$

The above simplification states that the relative location of part with respect to its parent is dependent on part-type, but not parent-type. For example, let i be a hand part, j its parent elbow part, and assume part types capture orientation. The above relational model states that a sideways-oriented hand should tend to lie next to the elbow, while a downward-oriented hand should lie below the elbow, regardless of the orientation of the upper arm.

5 Inference

Inference corresponds to maximizing $S(I, l, t)$ from (2) over l and t. When the relational graph $G = (V, E)$ is a tree, this can be done efficiently with dynamic programming. To illustrate inference, let us re-write (2) by defining $z_i = (l_i, t_i)$ to denote both the discrete pixel location and discrete mixture type of part i:

$$S(I,z) = \sum_{i \in V} \phi_i(I, z_i) + \sum_{ij \in E} \psi_{ij}(z_i, z_j),$$
$$\text{where} \quad \phi_i(I, z_i) = w_i^{t_i} \cdot \phi(I, l_i) + b_i^{t_i}$$
$$\psi_{ij}(z_i, z_j) = w_{ij}^{t_i,t_j} \cdot \psi(l_i - l_j) + b_{ij}^{t_i,t_j}$$

From this perspective, it is clear that our final model is a discrete, pairwise Markov random field (MRF). When $G = (V, E)$ is tree-structured, one can compute $\max_z S(I, z)$ with dynamic programming.

To be precise, we iterate over all parts starting from the leaves and moving "upstream" to the root part. We define kids(i) be the set of children of part i, which is the empty set for leaf parts. We compute the message part i passes to its parent j by the following:

$$\text{score}_i(z_i) = \phi_i(I, z_i) + \sum_{k \in \text{kids}(i)} m_k(z_i) \quad (6)$$

$$m_i(z_j) = \max_{z_i} \left[\text{score}_i(z_i) + \psi_{ij}(z_i, z_j) \right] \quad (7)$$

Eq.(6) computes the local score of part i, at all pixel locations l_i and for all possible types t_i, by collecting messages from the children of i. Eq.(7) computes for every location and possible type of part j, the best scoring location and type of its child part i. Once messages are passed to the root part ($i = 1$), score$_1(z_1)$ represents the best scoring configuration for each root position and type. One can use these root scores to generate multiple detections in image I by thresholding them and applying non-maximum suppression (NMS). By keeping track of the argmax indices, one can backtrack to find the location and type of each part in each maximal configuration. To find multiple detections anchored at the same root, one can use N-best extensions of dynamic programming [42].

Computation: The computationally taxing portion of dynamic programming is (7). We rewrite this step in detail:

$$m_i(t_j, l_j) = \max_{t_i} \Big[b_{ij}^{t_i,t_j} + \max_{l_i} \text{score}_i(t_i, l_i) + w_{ij}^{t_i,t_j} \cdot \psi(l_i - l_j) \Big] \quad (8)$$

One has to loop over $L \times T$ possible parent locations and types, and compute a max over $L \times T$ possible child locations and types, making the computation $O(L^2T^2)$ for each part. When $\psi(l_i - l_j)$ is a quadratic function (as is the case for us), the inner maximization in (8) can be efficiently computed for each combination of t_i and t_j in $O(L)$ with a max-convolution or distance transform [1]. Since one has to perform T^2 distance transforms, message passing reduces to $O(LT^2)$ per part.

Special cases: Model (3) maintains only a single spring per part, so message passing reduces to $O(L)$. Models (4) and (5) maintain only T springs per part, reducing message passing to $O(LT)$. It is worthwhile to note that our articulated model is no more computationally complex than the deformable mixtures of parts in [4], but is considerably more flexible because it searches over an exponential number (T^K) of global mixtures. In practice, the computation time is dominated by computing the local scores of each type-specific appearance models $w_i^{t_i} \cdot \phi(I, l_i)$. Since this score is linear, it can be efficiently computed for all positions l_i by optimized convolution routines.

6 Learning

We assume a supervised learning paradigm. Given labeled positive examples $\{I_n, l_n, t_n\}$ and negative examples $\{I_n\}$, we will define a structured prediction objective function similar to those proposed in [4], [25]. To do so, let us write $z_n = (l_n, t_n)$ and note that the scoring function (2) is linear in model parameters $\beta = (w, b)$, and so can be written as $S(I, z) = \beta \cdot \Phi(I, z)$. We would learn a model of the form:

$$\arg \min_{w, \xi_n \geq 0} \quad \frac{1}{2}\beta \cdot \beta + C \sum_n \xi_n \quad (9)$$
$$\text{s.t.} \quad \forall n \in \text{pos} \quad \beta \cdot \Phi(I_n, z_n) \geq 1 - \xi_n$$
$$\forall n \in \text{neg}, \forall z \quad \beta \cdot \Phi(I_n, z) \leq -1 + \xi_n$$

The above constraint states that positive examples should score better than 1 (the margin), while negative examples, for all configurations of part positions and types, should score less than -1. The objective function penalizes violations of these constraints using slack variables ξ_n.

Detection vs pose estimation: Traditional structured prediction tasks do not require an explicit negative training set, and instead generate negative constraints from positive examples with mis-estimated labels z. This corresponds to training a model that tends to score a ground-truth pose highly and alternate poses poorly. While this translates directly to a pose estimation task, our above formulation also includes a "detection" component: it trains a model that scores highly on ground-truth poses, but generates low scores on images without people. We find the above to work well for *both* pose estimation and person detection.

Optimization: The above optimization is a quadratic program (QP) with an exponential number of constraints, since the space of z is $(LT)^K$. Fortunately, only a small minority of the constraints will be active on typical problems (e.g., the support vectors), making them solvable in practice. This form of learning problem is known as a structural SVM, and there exists many well-tuned solvers such as the cutting plane solver of SVMStruct [43] and the stochastic gradient descent solver (SGD) in [4]. To allow greater flexibility in scheduling model updates and active-set pruning, we implemented our own

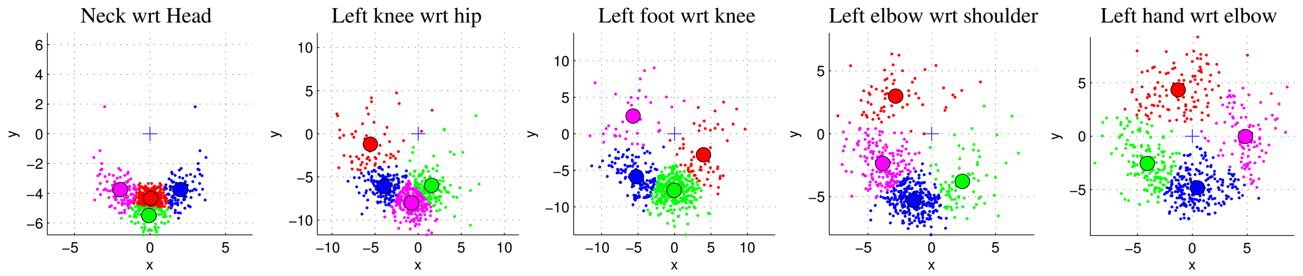


Fig. 3: We take a "data-driven" approach to orientation-modeling by clustering the relative locations of parts with respect to their parents. These clusters are used to generate mixture labels for parts during training. For example, heads tend to be upright, and so the associated mixture models focus on upright orientations. Because hands articulate to a large degree, mixture models for the hand are spread apart to capture a larger variety of relative orientations.

dual coordinate-descent solver, briefly described below.

Dual coordinate descent: The currently fastest solver for linear SVMs appears to be liblinear [44], which is a dual coordinate descent method. A naive implementation of a dual SVM solver would require maintaining a $M \times M$ kernel matrix, where M is the total number of active constraints (support vectors). The innovation of liblinear is the realization that one can implicitly represent the kernel matrix for linear SVMs by maintaining the primal weight vector β, which is typically *much* smaller. In practice, dual coordinate descent methods are efficient enough to reach near-optimal solutions in a single pass through large datasets [45]. Algorithmically, such a pass takes no more computation than SGD, but is guaranteed to always increase the dual objective, while stochastic methods may take wrong steps along the way. We have derived an extension of this insight for structural SVMs, described further in [46]. Briefly put, the main required modification is the ability for linear constraints to share the same slack variable. Specifically, the negative examples from (9) that correspond to a single window I_n with different latent variables z share the same slack ξ_n. This somewhat complicates a dual coordinate step, but the same principle applies; we solve the dual problem coordinate-wise, one-variable at a time, implicitly representing the kernel matrix with β. We also find that we reach optimal solutions in a single pass through our training set.

6.1 Learning in practice

Most human pose datasets include images with labeled joint positions [9], [7], [3]. We define parts to be located at joints, so these provide part position labels l, but not part type labels t. We now describe a procedure for generating type labels for our articulated model (5).

We first manually define the edge structure E by connecting joint positions based on average proximity. Because we wish to model articulation, we can assume that part types should correspond to different relative locations of a part with respect to its parent in E. For example, sideways-oriented hands occur next to elbows, while downward-facing hands occur below elbows. This means we can use relative location as a supervisory cue to help derive type labels that capture orientation.

Deriving part type from position: Assume that our n^{th} training image I_n has labeled joint positions l_n. Let l_i^n be the relative position of part i with respect to its parent in image I_n. For each part i, we cluster its relative position over the training set $\{l_i^n : \forall n\}$ to obtain T clusters. We use K-means with $K = T$. Each cluster corresponds to a collection of part instances with consistent relative locations, and hence, consistent orientations by our arguments above. We define the type labels for parts t_i^n based on cluster membership. We show example results in Fig. 3.

Partial supervision: Because part type is derived heuristically above, one could treat t_i^n as a latent variable that is also optimized during learning. This latent SVM problem can be solved by coordinate descent [4] or the CCP algorithm [47]. We performed some initial experiments with latent updating of part types using the coordinate descent framework of [4], but we found that type labels tend not to change over iterations. We leave such partially-supervised learning as interesting future work.

Problem size: On our training datasets, the number of positive examples varies from 200-1000 and the number of negative images is roughly 1000. We treat each possible placement of the root on a negative image as a unique negative example x_n, meaning we have millions of negative constraints. Furthermore, we consider models with hundreds of thousands of parameters. We found that a careful optimized solver was necessary to manage learning at this scale.

7 Experimental Results

7.1 Datasets

We evaluate results using the Image Parse dataset [9] and the Buffy Stickmen dataset [7], [48]. The Parse set contains 305 pose-annotated images of highly-articulated full-body human poses. The Buffy dataset contains 748 pose-annotated video frames over 5

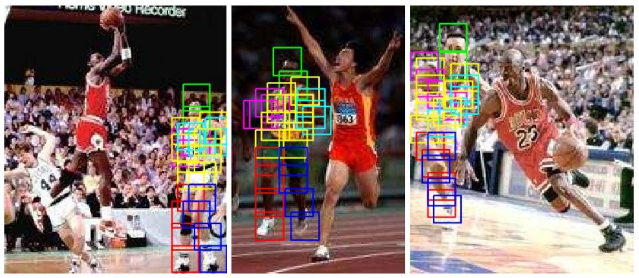


Fig. 4: We show images from the Parse benchmark for which the best-scoring pose of our model lies on a figure in the background, and not the central annotated figure. Previous evaluation criteria either penalize such matches as incorrect or match multiple candidate poses to the ground-truth (inadvertently favoring algorithms that return more candidates). We propose two new evaluation criteria that address these shortcomings.

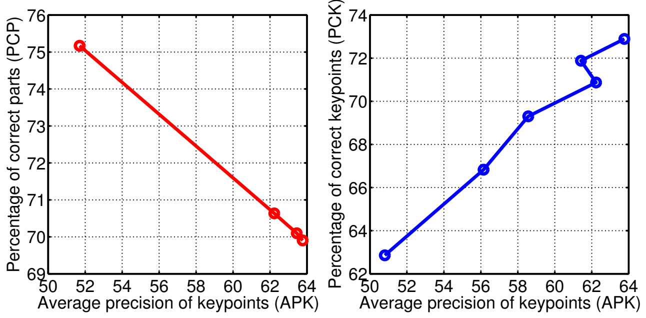


Fig. 5: We compare our "gold standard" evaluation criteria of average precision of keypoints (APK) with PCP and PCK (probability of correct keypoint). Recall that APK treats pose estimation as a body-part detection problem, and computes average precision from a precision-recall detector curve. On the **left**, we plot different PCP and APK values obtained by tweaking non-maximum suppression strategies. By generating more candidates, one produces a low APK but an artificially high PCP (as defined in the Buffy toolkit [8]), suggesting PCP does not correlate well with our gold standard. On the **right**, we show that PCK correlates positively with APK.

episodes of a TV show. Both datasets include a standard train/test split. To train our models, we use the negative training images from the INRIAPerson database [34] as our negative training set. These images tend to be outdoor scenes that do not contain people. Our good performance on other datasets (such as Buffy, which tends to include indoor images) suggests our model generalizes well.

7.2 Evaluation Criteria

In this section, we describe our new proposed evaluation criteria for evaluating pose estimation, and compare it to existing evaluation methods.

PCP: [7] describe a broadly-adopted evaluation protocol based on the probability of a correct pose (PCP), which measures the percentage of correctly localized body parts. A candidate body part is labeled as correct if its segment endpoints lie within 50% of the length of the ground-truth annotated endpoints. This criteria was clearly crucial and influential in spurring quantitative evaluation, thus considerably moving the field forward. However, there are three difficulties associated with using it in practice. Firstly, the Buffy toolkit [8] released with [7] uses a relaxed definition that scores the average of the predicted limb endpoints, and not the limb endpoints themselves. It is not clear which previous published PCP values use the evaluation code versus the original definition. Secondly, PCP is sensitive to the amount of foreshortening of a limb, and so can be too loose a measure in some cases and too strict a measure in others. Finally, PCP requires candidate and ground-truth poses to be placed in correspondence, but does not specify how to obtain this correspondence. Common solutions include evaluating the highest-scoring candidate given (a) an image with a single annotated person or (b) a window returned by a person detector. Option (a) is not satisfactory because the candidate may fire on an un-annotated person in the background (Fig. 4), while option (b) is not satisfactory because this biases the test data to be responses of a (rigid) person detector, as warned by [23]. The Buffy toolkit [8] instead matches multiple candidates to multiple ground-truth poses. Unmatched ground-truth poses (missed detections/false negatives) are penalized as incorrect localizations, but notably, *unmatched candidates (false positives) are not penalized*. This gives an unfair advantage to approaches that predict a large number of candidates, as we will show.

PCK: We propose two measures for pose estimation that address these issues. Our first evaluation explicitly factors out detection by requiring test images to be annotated with tightly-cropped bounding box for each person. Crucially, we do *not* limit ourselves to evaluating a subset of verified bounding boxes found by a detector, as this biases the test windows to be rigid poses (as warned by [23]). Our approach is similar to the protocol used in the PASCAL Person Layout Challenge [49]. Given the bounding box, a pose estimation algorithm must report back keypoint locations for body joints. The Person Layout Challenge measures the overlap between keypoint bounding boxes, which can suffer from quantization artifacts for small bounding boxes. We define a candidate keypoint to be correct if it falls within $\alpha \cdot \max(h, w)$ pixels of the ground-truth keypoint, where h and w are the height and width of the bounding box respectively, and α controls the relative threshold for considering correctness. We use $\alpha = 0.1$ for Parse dataset and $\alpha = 0.2$ for Buffy dataset, due to the fact that Buffy contains half-body people while Parse contains full-body people.

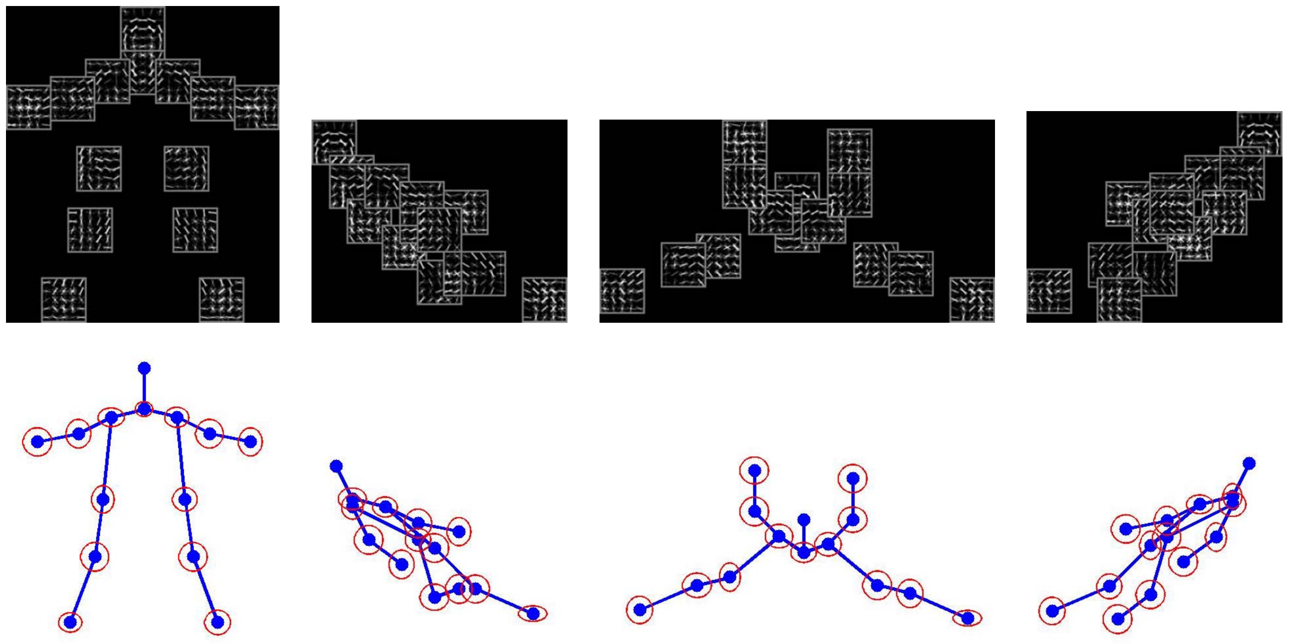


Fig. 6: A visualization of our model for $K = 14$ parts and $T = 4$ local mixtures, trained on the Parse dataset. We show the local templates **above**, and the tree structure **below**, placing parts at their best-scoring location relative to their parent. Though we visualize 4 trees, there exists $T^K \approx 2e7$ global combinations, obtained by composing different part types together with different springs. The score associated with each combination decomposes into a tree, and so is efficient to search over using dynamic programming (1).

Instead of manually annotating bounding boxes as PASCAL Person Layout Challenge does, we generate each of them as the tightest box that covers the set of ground truth keypoints.

APK: In a real system, however, one will not have access to annotated bounding boxes at test time, and so must address the detection problem as well. One can cleanly combine the two problems by thinking of body parts (or rather joints) as objects to be detected, and evaluate object detection accuracy with a precision-recall curve [49]. As above, we deem a candidate to be correct (true positive) if it lies within $\alpha \cdot \max(h, w)$ of the ground-truth. We call this the average precision of keypoints (APK). This evaluation correctly penalizes both missed-detections and false-positives. Note that correspondence between candidates and ground-truth poses are established separately for each keypoint, and so this only provides a "marginal" view of keypoint detection accuracy. But such marginal statistics are useful for understanding which parts are more difficult than others. Finally, APK requires *all* people to be labeled in a test image, unlike PCP and PCK. We have produced such annotations for Parse and Buffy, and will make them public.

PCP vs PCK vs APK. We compare different evaluations for the Parse dataset in Fig. 5, using the implementation of PCP in the Buffy toolkit. Because APK is the most realistic and strictest evaluation, we deem it the "gold standard". By tweaking the non-maximum suppression (NMS) strategy for our detector to return more candidate poses, we do worse at APK but artificially do better at PCP (as implemented in the Buffy toolkit). This behavior makes sense given that false positives are not penalized by PCP, but penalized by APK. We would like to produce a similar curve comparing APK and PCK under different NMS strategies, but recall that PCK is not affected by NMS because ground-truth windows are given. Rather, we select a arbitrary dimension of our model to evaluate (such as the number of mixtures), and show a positive correlation of PCK with APK. Because PCK is easier to interpret and faster to evaluate than APK, we use PCK to perform diagnostic experiments exploring different aspects of our model in the next section.

7.3 Diagnostic experiments

We define a full-body skeleton for the Parse set, and a upper-body skeleton for the Buffy set. To define a fully labeled dataset of part locations and types, we group parts into orientations based on their relative location with respect to their parents (as described in Section 6.1). We show clustering results in Fig. 3. We use the derived type labels to construct a fully supervised dataset, from which we learn flexible mixtures of parts. We show the full-body model learned on the Parse dataset in Fig. 6. We set all parts to be 5×5 HOG cells in size. To visualize the model, we show 4 trees generated by selecting one of the four types of each part, and placing it at its maximum-scoring position. Recall that each part type has its own appearance template and spring encoding its relative location with respect to its parent. This is because we

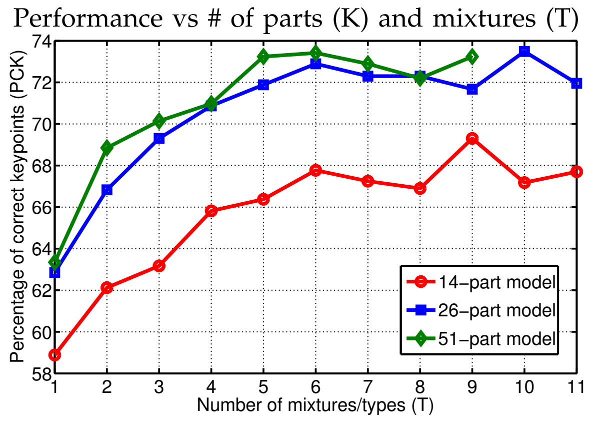


Fig. 7: We show the effect of model structure on pose estimation by evaluating PCK performance on the Parse dataset. Overall, increasing the number of parts from 14 to 26 (by instancing parts at limb midpoints in addition to joints) improves performance. Instancing additional middle parts between limb midpoints and joints (from 26 to 51) yields no clear improvement. In all cases, increasing the number of mixtures improves performance, likely due to the fact that more orientations and foreshortening can be modeled. We find a 26-part model with 6 mixtures provides a good trade-off of performance vs computation.

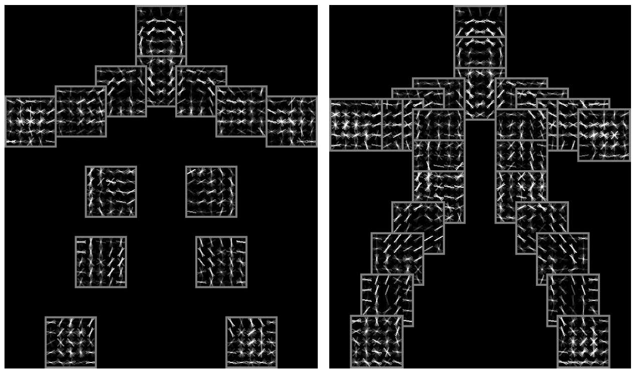


Fig. 8: We visualize our 14 and 26 part model. In Fig. 7, we demonstrate that the additional parts in the 26-part model significantly increase performance.

expect part types to correspond to orientation because of the supervised labeling shown in Fig. 3. Though we visualize 4 trees, we emphasize that there exists an *exponential* number of trees that our model can generate by composing different part types together.

Structure: We consider the effect of varying T (the number of mixtures or types) and K (number of parts) on the accuracy of pose estimation on the Parse dataset in Fig. 7. We experiment with a 14 part model defined at 14 joint positions (shoulder, elbow, hand, etc.) and a 26 part model where midway points between limbs are added (mid-upper arm, mid-lower arm, etc.) to increase coverage. Following the clustering procedure Sec. 6.1, multiple parts on the same limb will have identical mixture type assignments, and so will have consistent orientation states. Performance increases with denser coverage and an increased number of part types, presumably because additional orientations are being captured.

Joint, Independent, and Invariant parts (PCK)

Model	Joint	Indep	Indep+Invar
14 parts	67.8	56.8	48.4
26 parts	72.9	56.8	39.6

TABLE 1: We evaluate various strategies for training parts. We jointly train rotationally-variant part models, but much past work trains rotationally-invariant part detectors. We demonstrate the latter decreases our performance by nearly a factor of 2, suggesting that joint training and rotationally-variant detectors are crucial for high performance.

Diagnostic analysis (PCK)

Joint	No latent	Star	Add rotated images
72.9	73.4	56.7	74.6

TABLE 2: We consider the effect of other aspects of our model (using 26 parts model as an example), including no latent updating, the use of a star structure versus a tree structure, and the addition of rotated training images to increase the size of our training set. We find that the latent updating of mixture labels is not crucial, a star model definitively hurts performance, and adding rotated copies of our training images increases performance by a small but noticeable amount.

Independently-trained parts: In Table 1, we consider different strategies for training parts. Our model jointly trains all parts and their relational constraints with a structured SVM. We also consider a variant of our model where part templates are trained independently with an SVM (the middle column); at test time, we use still dynamic programming to find full-body configurations. We see a significant drop in performance, indicating that joint contextual training is crucial. For example, a forearm part trained independently will be inaccurate because many negative examples will contain parallel lines and be “hard” (e.g., support vectors for an SVM). However, structured SVMs (that jointly train all parts) need collect hard negatives only from backgrounds that trigger a full-body part configuration. This vastly reduces the amount of background clutter that the forearm part must compete against at train-time. We see a larger drop for our 26-part model compared to our 14-part model. Because parts in the larger model tend to overlap more, we posit that they need to be trained jointly to properly calibrate the influence of overlapping regions.

Image Parse Testset

Method	Torso	Head	Upper legs	Lower legs	Upper arms	Lower arms	Total
Ramanan [9]	52.1	37.5	31.0	29.0	17.5	13.6	27.2
Andriluka [27]	81.4	75.6	63.2	55.1	47.6	31.7	55.2
Johnson [33]	77.6	68.8	61.5	54.9	53.2	39.3	56.4
Singhi [21]	91.2	76.6	71.5	64.9	50.0	34.2	60.9
Johnson [50]	85.4	76.1	73.4	65.4	64.7	46.9	66.2
Johnson [51]	87.6	76.8	74.7	67.1	67.4	45.9	67.4
Us [52] single+both (strict)	78.5	81.5	72.2	65.9	55.6	33.7	61.5
Us [6] single+both (strict)	85.9	86.8	74.9	68.3	63.4	42.7	67.1
Us [6] match+avg (buffy tk)	96.1	99.0	85.9	79.0	79.0	53.4	79.0
Us [6] match+both	91.7	94.6	79.3	71.0	66.3	43.9	70.7

TABLE 3: We compare our model to all previous published results on the Parse dataset. Because authors are likely using different definitions/implementations of PCP, previous values may not be comparable to each other. Here we list all possible interpretations of PCP for our model in the last four rows. We also include a preliminary version of model [52]. Single+both is the strictest interpretation of PCP which evaluates only a single detection (given by the maximum scoring detection of an algorithm) for one image and it requires both endpoints of a limb to be correct. Match+avg is the Buffy toolkit [8] implementation, which allows an algorithm to report multiple detections per image, and performs an explicit correspondence matching with the ground truth without penalizing false positives. It also requires only the average of the endpoints of a limb to be correct, rather than both endpoints. Match+both matches multiple candidates without penalizing false positive, but requires both endpoints to be correct. Even under the most strictest criteria, our current model (67.1%) still outperforms all the others and works almost same as Johnson[51] which uses a separate dataset of 10000 images. We argue that none of these interpretations are satisfactory, and propose new evaluation criteria (PCK and APK) described at length in the text.

Rotationally-invariant parts: We also consider the effect of rotationally-invariant parts in the third column of Table 1. We train independent, rotationally-invariant parts (for say, the elbow) as follows: for each discrete rotation, we warp all elbow training patches to that rotation and train an SVM. This means each oriented elbow part is trained with the entire training set, while our mixture model uses only a subset of data belonging to that mixture. We see a large drop in performance, suggesting that elbows (and other parts) look different even when rotated to an appropriate coordinate system. We posit this is due to geometric interactions with other parts, such as partial occlusions and effects from clothing. Our local mixtures capture this geometric dependency. Most previous approaches to pose estimation use independently-trained, invariant parts. We find that joint training of orientation-variant parts increases performance by nearly a factor of 2, from 39% to 72% PCK.

Other aspects: We consider the effect of other aspects of our model in Table 2, including no latent updating, the use of a star structure versus a tree structure, and the addition of rotated training images to increase the size of our training set. We find that latent updating of mixture labels is not helpful, a star model definitively hurts performance, and adding small copies of our training data rotated by $\pm 15^\circ$ increases performance by a small but noticeable amount. The latter probably holds true because the training set on PARSE is rather small (100 images), so artificially augmenting the training set helps somewhat. Our final system used in the benchmark results below makes use of the augmented training set.

7.4 Benchmark results

Parse: We give quantitative results for PCP in Table 3, PCK and APK in Fig. 9, and show example images in Fig. 12. It is difficult to directly compare PCP performance due to the ambiguities in the definition and implementation that were discussed earlier. We refer the reader to the captions for a detailed analysis, but our method appears to be at or above the state-of-the-art. We suspect that previous authors either report a single candidate pose per image, or multiple poses that are matched using the code of [7]. Our analysis suggests both of these reports are unsatisfactory, since the former unfairly penalizes an algorithm for finding a person in the background (Fig. 4), while the latter unfairly favors algorithms that report many candidate detections (Fig. 5). We report our performance for all possible interpretations of PCP. Under all variants, our algorithm still outperforms all prior work that makes use of the given benchmark training set, while being orders of magnitude faster.

Our diagnostic analysis suggests our high performance is due to the fact that our mixtures of parts are learned jointly in a discriminative framework, and the fact that our model is efficient enough to search over scales and locations. In contrast, articulated models are often learned in stages (using pre-trained, orientation-invariant part detectors), and are often applied at a fixed scale and location due to the computational burden of inference.

(a) PCK on Parse

Avg	Hea	Sho	Elb	Wri	Hip	Kne	Ank
72.9	90.2	85.4	68.0	47.1	77.1	75.4	67.1

(b) APK on Parse

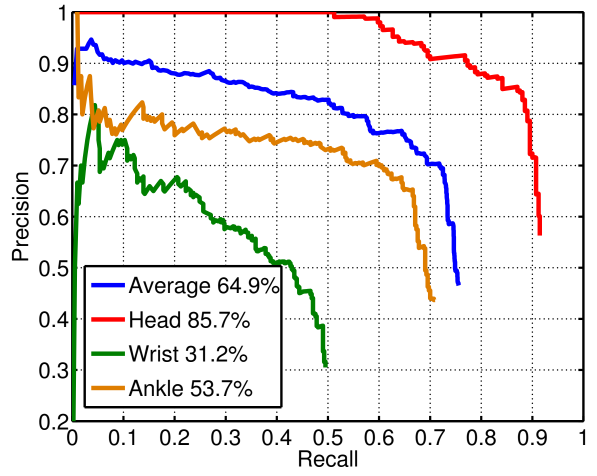


Fig. 9: We present our new evaluations of probability of correct keypoint (PCK) and average precision of keypoints (APK) on the Parse testset, using a full-body model of $K = 26$ parts and $T = 6$ mixtures. PCK is the fraction of times a predicted keypoint was close to a ground-truth keypoint *given* a human bounding box on a test image. APK evaluates the average precision of keypoint detections obtained without access to *any* annotations on the test image; this requires a pose estimation algorithm to perform non-maximum suppression (NMS).

Buffy: We give quantitative results for PCP in Table 4, PCK and APK in Fig. 10, and show example images in Fig. 13. To compare to previous results, we evaluate pose estimation on a subset of windows returned by upper-body detector (provided in the evaluation kit). Notably, all previous approaches use articulated parts. Our algorithm is several orders of magnitude faster than the next-best approaches of [29], [26]. As pointed out by [23], this subset contains little pose variation because it is biased to be responses of a rigid template. The distributed evaluation code [7] also allows one to compute performance on the full test videos by multiplying PCP values with the overall detection rate, but as we argue, this unfairly favors methods that report back many candidate poses (because false positives are not penalized). Indeed, the original performance we reported in [10] appears to be inflated due to this effect. Rather, we evaluate the full test videos using our new criteria for PCK and APK. Our PCK score outperforms our PCP score, likely due to foreshortened arms in the data that are scored too stringently with PCP. Finally, we compare the publically-available code of [48] with our new APK criteria, and show that our method does significantly better.

Subset of Buffy Testset

Buffy	Torso	Head	U.arms	L.arms	Total
Tran [23]					62.3
Andr. [27]	90.7	95.5	79.3	41.2	73.5
Eich. [53]	98.7	97.9	82.8	59.8	80.1
Sapp [29]	100	100	91.1	65.7	85.9
Sapp [26]	100	96.2	95.3	63.0	85.5
Us [52] buffy tk	98.8	99.2	97.8	68.6	88.5
Us [52] strict	98.4	98.8	94.3	57.5	83.5

TABLE 4: The Buffy testset is distributed with a subset of windows detected by a rigid HOG upper-body detector. We compare our results to all published work on this set. We obtain the best overall PCP while being orders of magnitude faster than the next-best approaches. These results have the caveat that authors maybe using different definitions / implementations of PCP, making them incomparable. Our total pipeline requires 1 second to process an image, while [29], [26] take 5 minutes. We outperform or (nearly) tie all previous results on a per-part basis. As pointed out by [23], this subset contains little pose variation because it is biased to be responses of a rigid template. We present results on the full testset using our novel criteria of PCK and APK in Fig. 10.

(a) PCK on Buffy

Avg	Head	Sho	Elb	Wri	Hip
91.6	98.6	98.2	95.3	73.9	92.0

(b) APK on Buffy

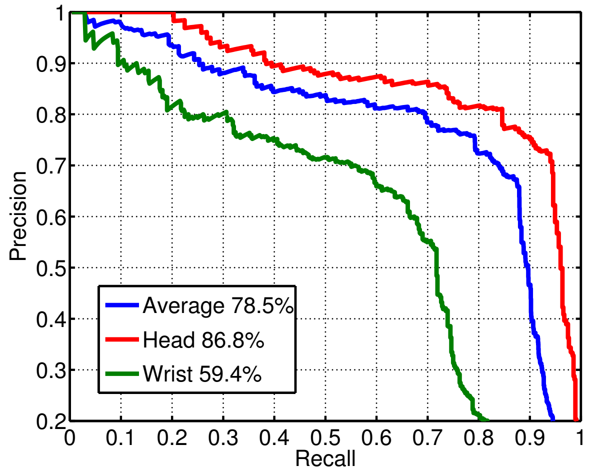


Fig. 10: We present our new evaluations of probability of correct keypoint (PCK) and average precision of keypoints (APK) on the Buffy testset, using a model of $K = 18$ parts and $T = 6$ mixtures. Note that the overall average APK (across all keypoints) is 78.5%, indicating this testset is easier than the Parse image benchmark.

Buffy Testset

Buffy	Head	Sho	Elb	Wrist	Hip	Total
Eich. [48]	84.5	79.9	70.5	43.8		69.7
Us	**86.8**	**87.0**	**81.2**	**59.4**	**77.9**	**78.5**

TABLE 5: We score the publically-available code of [48] using our new APK criteria. Our method performs considerable better for all keypoints.

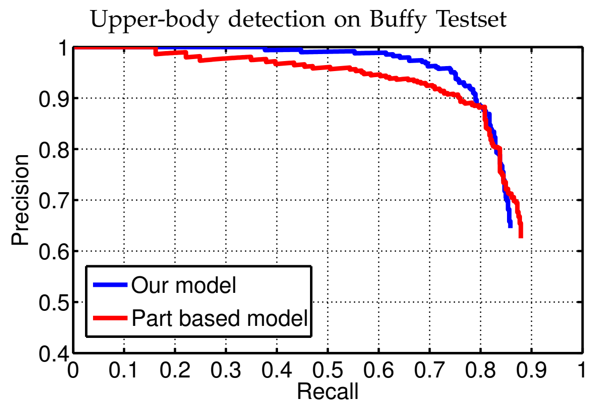


Fig. 11: We plot upper-body detection on the Buffy dataset, comparing our articulated pose detector with the state-of-the-art deformable part model of [4] trained on the same data as our model.

Detection accuracy: We can use our model as an upper body detector on the Buffy dataset shown in Table 11. We compare to the popular DPM model [4], trained on the same training set as our model (but without supervised part annotations). We see that we obtain higher precision for nearly all recall values. These results indicate the potential of our flexible representation and supervised learning framework for general object detection.

8 Conclusion

We have described a simple, but flexible extension of part models to include local mixtures of parts. We use local mixtures to capture the appearance changes of parts due to articulation. We augment part models, which reason about spatial relations between part locations, to also reason about co-occurrence relations between part mixtures. Our models capture the dependence of local appearance on spatial geometry, outperforming classic articulated models in both speed and accuracy. Our local part mixtures can be composed to generate an exponential number of global mixtures, greatly increasing their representational power without sacrificing computational efficiency. Finally, we introduce new evaluation criteria for pose estimation and articulated human detection which address limitations of previous scoring methods. We demonstrate impressive results for the challenging task of human pose estimation.

References

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

[2] M. Fischler and R. Elschlager, "The representation and matching of pictorial structures," *IEEE Transactions on Computers*, vol. 100, no. 1, pp. 67–92, 1973.

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

[4] P. Felzenszwalb, R. Girshick, D. McAllester, and R. D., "Object detection with discriminatively trained part-based models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 9, pp. 1627–1645, 2010.

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

[6] Y. Yang and D. Ramanan, "Flexible mixtures of parts for articulated pose detection, release 1.3," http://phoenix.ics.uci.edu/software/pose/.

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

[8] V. Ferrari, M. Eichner, M. Marin-Jimenez, and A. Zisserman, "Buffy stickmen v3.01: Annotated data and evaluation routines for 2d human pose estimation," http://www.robots.ox.ac.uk/ vgg/data/stickmen/.

[9] D. Ramanan, "Learning to parse images of articulated bodies," in *Advances in Neural Information Processing System*, 2007.

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

[11] J. O'Rourke and N. Badler, "Model-based image analysis of human motion using constraint propagation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 2, no. 6, pp. 522–536, 1980.

[12] D. Hogg, "Model-based vision: a program to see a walking person," *Image and Vision Computing*, vol. 1, no. 1, pp. 5–20, 1983.

[13] K. Rohr, "Towards model-based recognition of human movements in image sequences," *CVGIP-Image Understanding*, vol. 59, no. 1, pp. 94–115, 1994.

[14] D. Ramanan, "Part-based models for finding people and estimating their pose," pp. 199–223, 2011.

[15] S. Ioffe and D. Forsyth, "Human tracking with mixtures of trees," in *IEEE International Conference on Computer Vision*, 2001.

[16] M. Lee and I. Cohen, "Proposal maps driven mcmc for estimating human body pose in static images," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2004.

[17] S. Ioffe and D. Forsyth, "Probabilistic methods for finding people," *International Journal of Computer Vision*, vol. 43, no. 1, pp. 45–68, 2001.

[18] L. Sigal and M. Black, "Measure locally, reason globally: Occlusion-sensitive articulated pose estimation," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2006.

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

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

[21] V. Singh, R. Nevatia, and C. Huang, "Efficient inference with multiple heterogenous part detectors for human pose estimation," in *European Conference on Computer Vision*, 2010.

[22] X. Lan and D. Huttenlocher, "Beyond trees: Common-factor models for 2d human pose recovery," in *IEEE International Conference on Computer Vision*, 2005.

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

[24] D. Ramanan and C. Sminchisescu, "Training deformable models for localization," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2006.

[25] M. Kumar, A. Zisserman, and P. Torr, "Efficient discriminative learning of parts-based models," in *IEEE International Conference on Computer Vision*, 2009.

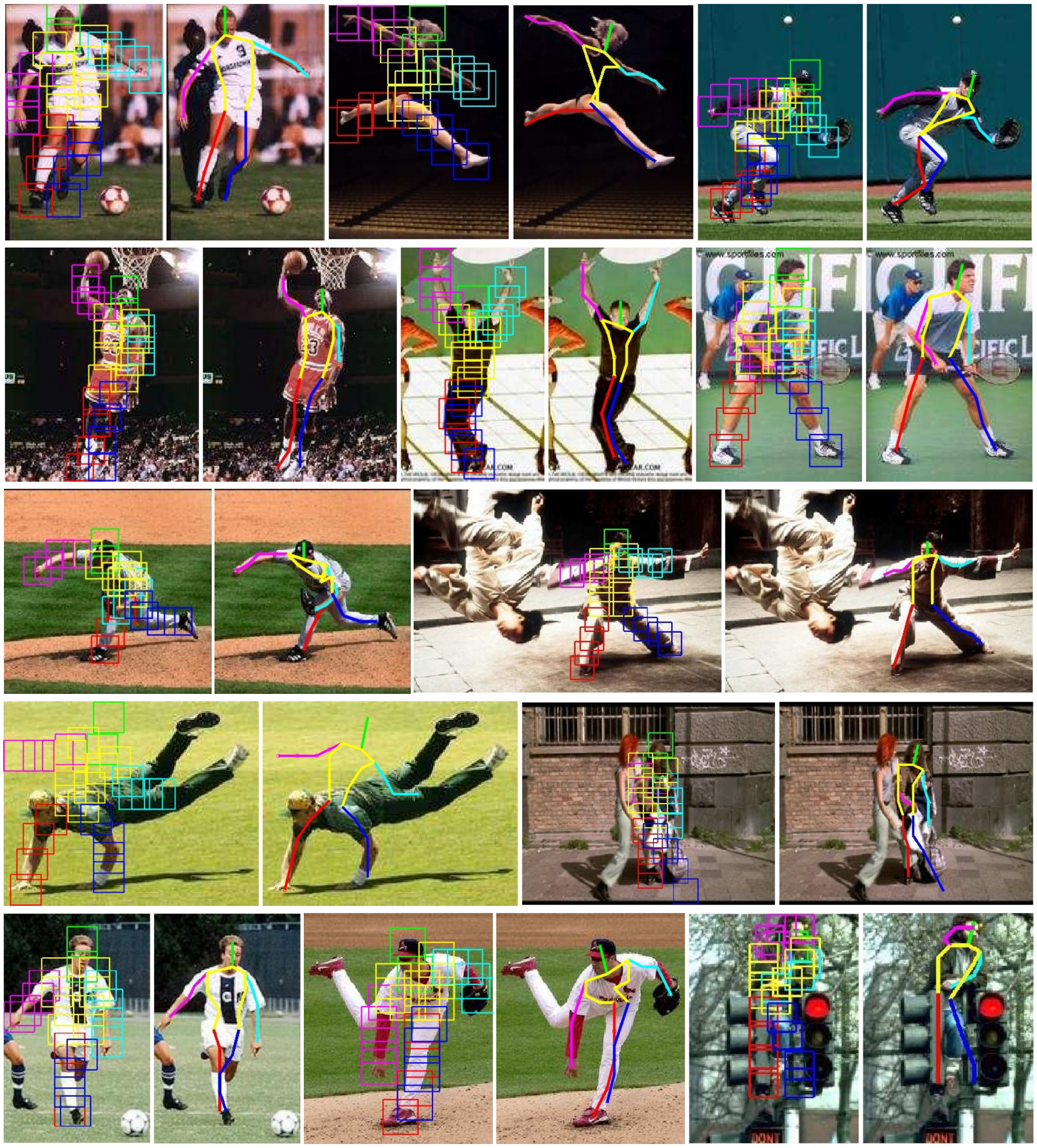


Fig. 12: Results on the Parse dataset. We show 26 part bounding boxes reported by our algorithm along with skeletons computed from the bounding boxes for each image. The **top 3** rows show successful examples, while the **bottom 2** rows show failure cases. Examining failure cases from top left to bottom right, we find our model is not flexible enough to model horizontal people, is confused by overlapping people, suffers from double-counting phenomena common to tree models (both the left and right legs fire on the same image region), and is confused when objects partially occlude people.

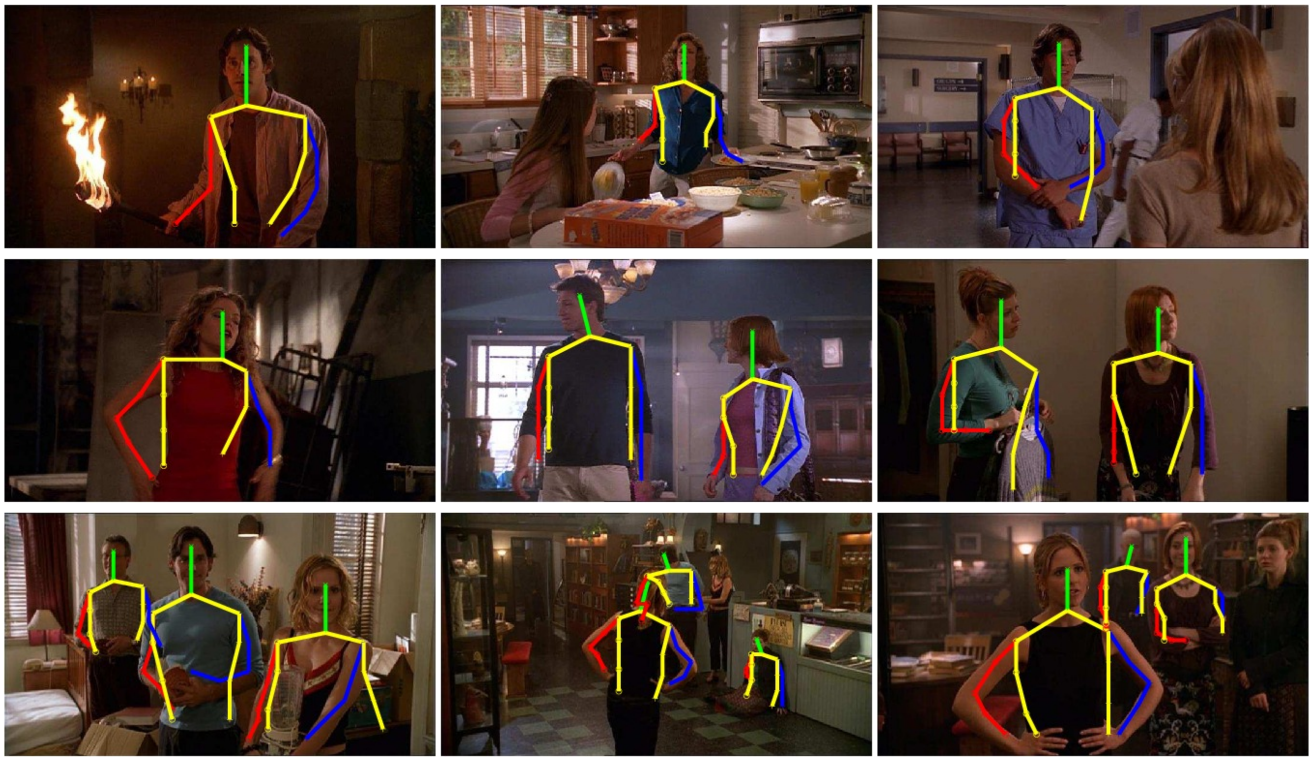


Fig. 13: Results on the Buffy dataset. We visualize *all* skeletons (instead of boxes) reported by our algorithm for a given image, after non-maximum suppression (NMS). Our model hence serves as both an articulated detector and pose estimation algorithm, as evidenced by our average-precision of keypoints (APK) measure.

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

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

[28] G. Mori, X. Ren, A. Efros, and J. Malik, "Recovering human body configurations: Combining segmentation and recognition," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2004.

[29] B. Sapp, C. Jordan, and B. Taskar, "Adaptive pose priors for pictorial structures," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

[30] P. Srinivasan and J. Shi, "Bottom-up recognition and parsing of the human body," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

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

[32] J. Sullivan and S. Carlsson, "Recognizing and tracking human action," in *European Conference on Computer Vision*, 2002.

[33] S. Johnson and M. Everingham, "Combining discriminative appearance and segmentation cues for articulated human pose estimation," in *IEEE International Conference on Computer Vision Workshops*, 2009.

[34] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2005.

[35] P. H. and R. D., "Steerable part models," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[36] W. Yang, Y. Wang, and G. Mori, "Recognizing human actions from still images with latent poses," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

[37] M. Sun and S. Savarese, "Articulated part-based model for joint object detection and pose estimation," in *IEEE International Conference on Computer Vision Workshops*, 2011.

[38] Y. Wang, D. Tran, and D. Liao, Z. Forsyth, "Discriminative hierarchical part-based models for human parsing and action recognition," in *Journal of Machine Learning Research*, 2012.

[39] B. Epshtein and S. Ullman, "Semantic hierarchies for recognizing objects and parts," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[40] L. Zhu, Y. Chen, C. Lin, and A. Yuille, "Max margin learning of hierarchical configural deformable templates (hcdts) for efficient object parsing and pose estimation," *International Journal of Computer Vision*, vol. 93, no. 1, pp. 1–21, 2011.

[41] B. Sapp, D. Weiss, and B. Taskar, "Parsing human motion with stretchable models," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

[42] D. Park and D. Ramanan, "N-best maximal decoders for part models," in *IEEE International Conference on Computer Vision*, 2011.

[43] I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun, "Support vector machine learning for interdependent and structured output spaces," in *International Conference on Machine Learning*, 2004.

[44] R. Fan, K. Chang, C. Hsieh, X. Wang, and C. Lin, "Liblinear: A library for large linear classification," *Journal of Machine Learning Research*, vol. 9, pp. 1871–1874, 2008.

[45] A. Bordes, L. Bottou, P. Gallinari, and J. Weston, "Solving multiclass support vector machines with larank," in *International Conference on Machine Learning*, 2007.

[46] D. Ramanan, "Dual coordinate descent solvers for large structured prediction problems," UC Irvine, Tech. Rep., 2012, to appear.

[47] A. Yuille and A. Rangarajan, "The concave-convex procedure," *Neural Computation*, vol. 15, no. 4, pp. 915–936, 2003.

[48] M. Eichner, M. Marin-Jimenez, A. Zisserman, and V. Ferrari, "2d articulated human pose estimation and retrieval in (almost) unconstrained still images," *International Journal of Computer Vision*, vol. 99, no. 2, pp. 190–214, 2012.

[49] M. Everingham, L. Van Gool, C. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge,"

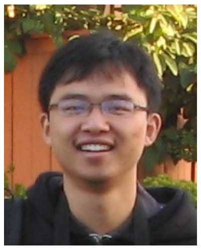
International Journal of Computer Vision, vol. 88, no. 2, pp. 303–338, 2010.

[50] S. Johnson and M. Everingham, "Clustered pose and nonlinear appearance models for human pose estimation," in *British Machine Vision Conference*, 2010.

[51] ——, "Learning effective human pose estimation from inaccurate annotation," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

[52] Y. Yang and D. Ramanan, "Flexible mixtures of parts for articulated pose detection, release 1.2," http://phoenix.ics.uci.edu/software/pose/.

[53] M. Eichner and V. Ferrari, "Better appearance models for pictorial structures," in *British Machine Vision Conference*, 2009.



Yi Yang Yi Yang received a BS with honors from Tsinghua University in 2006 and received a Master of Philosophy degree in Industrial Engineering in Hong Kong University of Science and Technology in 2008. He is currently a PhD student in the Department of Computer Science at the University of California, Irvine. His research interests are in artificial intelligence, machine learning and computer vision.

Deva Ramanan is an associate professor of Computer Science at the University of California at Irvine. He received his Ph.D. in Electrical Engineering and Computer Science from UC Berkeley in 2005. His research interests span computer vision, machine learning, and computer graphics, with a focus on the application of understanding people through images and video.