Noname manuscript No.
(will be inserted by the editor)

Max Margin Learning of Hierarchical Configural Deformable Templates (HCDTs) for Efficient Object Parsing and Pose Estimation

Long (Leo) Zhu · Yuanhao Chen · Chenxi Lin · Alan Yuille

the date of receipt and acceptance should be inserted later

Abstract In this paper we formulate a hierarchical configurable deformable template (HCDT) to model articulated visual objects – such as horses and baseball players – for tasks such as parsing, segmentation, and pose estimation. HCDTs represent an object by an AND/OR graph where the OR nodes act as switches which enables the graph topology to vary adaptively. This hierarchical representation is compositional and the node variables represent positions and properties of subparts of the object. The graph and the node variables are required to obey the *summarization principle* which enables an efficient *compositional inference* algorithm to rapidly estimate the state of the HCDT. We specify the structure of the AND/OR graph of the HCDT by hand and learn the model parameters discriminatively by extending Max-Margin learning to AND/OR graphs. We illustrate the three main aspects of HCDTs – representation, inference, and learning – on the tasks of segmenting, parsing, and pose (configuration) estimation for horses and humans. We demonstrate that the inference algorithm is fast and that max-margin learning is effective. We show that HCDTs gives state of the art results for segmentation and pose estimation when compared to other methods on benchmarked datasets.

Keywords
Hierarchy, Shape Representation, Object Parsing, Segmentation, Structure Learning, Max Margin.

Long (Leo) Zhu
Department of Statistics
University of California at Los Angeles
Los Angeles, CA 90095
E-mail: lzhu@stat.ucla.edu

Yuanhao Chen
University of Science and Technology of China
Hefei, Anhui 230026 P.R.China
E-mail: yhchen4@ustc.edu

Chenxi Lin
Alibaba Group R&D
E-mail: chenxi.lin@alibaba-inc.com

Alan Yuille
Department of Statistics,Psychology and Computer Science
University of California at Los Angeles
Los Angeles, CA 90095
E-mail: yuille@stat.ucla.edu

1 Introduction

Vision is a pre-eminent machine intelligence problem which is extremely challenging due to the complexity and ambiguity of natural images. Its importance and difficulty can be appreciated by realizing that the human brain devotes roughly half of the cortex to visual processing.

In recent years there has been encouraging progress in addressing challenging machine intelligence problems by formulating them in terms of probabilistic inference using probability models defined on structured knowledge representations. These structured representations, which can be formalized by attributed graphs, encode knowledge about the structure of the data. Important examples include stochastic context free grammars [24] and AND/OR graphs [15, 6, 17]. The probability distributions deal with the stochastic variability in the data and, when training data is available, enable the models to be learnt rather than being specified by hand. But although this approach is conceptually very attractive for vision [52] it remains impractical unless we can specify efficient algorithms for performing inference and learning. For example, the success of stochastic context free grammars [24] occurs because the nature of the structured representation for these types of grammars enables efficient inference by dynamic programming. The classic applications of machine learning to vision concentrated on applications where there was no need to model the structure (i.e. there was no benefit to having unobserved hidden states) such as support vector machines [28] and AdaBoost

applied to face [43] and text detection [5]. More recent work – such as latent SVM [16] and boosting [44] – does involve some hidden variables but these models still contain far less structure than the probabilistic grammars, such as AND/OR graphs, which seem necessary to deal with the full complexity of vision.

In this paper we address the problem of parsing, segmenting and estimating the pose/configuration of articulated objects, such as horses and humans, in cluttered backgrounds. By parsing, we mean estimating the positions and properties of all sub-parts of the object. Pose estimation for articulated objects, and in particular humans, has recently received a lot of attention. Such problems arise in many applications including human action analysis, human body tracking, and video analysis. But the major difficulties of parsing the human body, which arise from the *large appearance variations* (e.g. different clothes) and *enormous number of poses*, have not been fully solved. From our machine learning perspective, we must address three issues. *Firstly*, what object representation is capable of modeling the large variation of both shape and appearance? *Secondly*, how can we learn a probabilistic model defined on this representation? *Thirdly*, if we have a probabilistic model, how can we perform inference efficiently (i.e. rapidly search over all the possible configurations of the object in order to estimate the positions and properties of the object subparts for novel images). These three issues are clearly related to each other. Intuitively, the greater the representational power, the bigger the computational complexity of learning and inference. Most works in the literature, e.g. [38,27,7], focus on only one or two aspects, and not on all of them (see section (2.2) for a review of the literature). In particular, the representations used have been comparatively simple. Moreover, those attempts which do use complex representations tend to specify their parameters by hand and do not learn them from training data.

In this paper, we *represent* the different configurations/poses of humans and horses by the hierarchical AND/OR graphs proposed by Chen et al. for modeling deformable articulated objects [7], see figure (1). The nodes of this graph specify the position, orientation and scale of sub-parts of the object (together with an index variable which specifies which subparts of the object are present). This gives a representation where coarse information about the object – such as its position and size – are specified at the top level node of the graph and more precise details are encoded at the lower levels. We call this the *summarization* principle because, during inference and learning, the lower level nodes in the graph only pass on summary statistics of their states to the higher level nodes – alternatively, the higher level nodes give an "executive summary" of the lower level nodes. In this paper we restrict the graph to be a tree which simplifies the inference and learning problems.

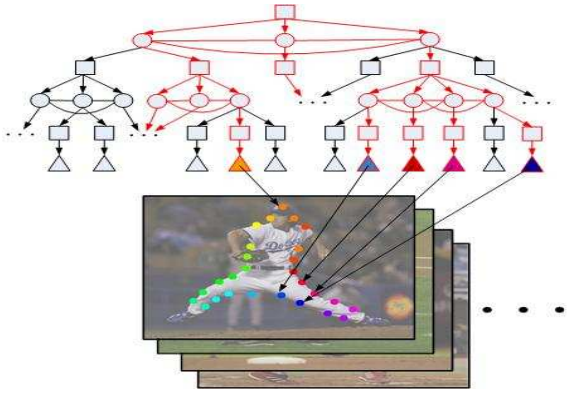

Fig. 1 The HCDT uses an AND/OR representation which enables us to represent many different configurations/poses of the object by exploiting reusable subparts. The AND nodes (squares) represent compositions of subparts represented by their children. The OR nodes (circles) are "switch nodes" where the node can select one of its child nodes, hence enabling the graph to dynamically change topology. The LEAF nodes (triangles) relate directly to the image. The compositional inference algorithm estimates a *parse tree* which is an instantiation of the AND/OR graph. This involves determining which subparts are present – i.e. the graph topology specified by the OR nodes – and estimating the positions and properties of the subparts. For the baseball player dataset [26], the topology of the parse tree specifies the pose/configuration of the baseball player. The nodes and edges in red indicate one parse tree. In this paper, the HCDT for baseball players represents 98 possible poses – i.e. parse tree topologies.

The probability distribution defined on this representation is built using local potentials, hence obeying the Markov condition, and is designed to be invariant to the position, pose, and size of the object. The AND nodes encodes spatial relationships between different sub-parts of the graph while the OR nodes act as switches to change the graph topology. The AND/OR graph represents a mixture of distributions where the state of the OR nodes determines the mixture component. But the AND/OR graph is compact because different topologies can contain the same sub-parts (see reusable parts [17]). Hence the AND/OR graph (see figure (1)) can represent a large number of different topologies (98 for humans and 40 for horses) while using only a small total number of nodes (see analysis in section (3.3).

We perform *inference* on the HCDT to parse the object – determine which subparts are present in the image and estimate their positions and properties. Inference is made efficient by exploiting the hierarchical tree structure, the summarization principle, and the reusable parts. These imply that we can perform exact inference by dynamic programming which is of polynomial complexity in terms of the dimensions of the state space (e.g. the number of positions and orientations in the image). But these dimensions are so high that we perform pruning to restrict the number of state configurations that we evaluate (pruning is often done in practical implementations of dynamic programming [9]). This leads to a very intuitive algorithm where proposals for

the states of the higher level nodes are constructed by composing proposals for the lower level nodes and keeping the number of proposals small by pruning. Pruning is done in two ways: (i) removing proposals whose goodness of fit (i.e. energy) is poor, and (ii) performing *surround suppression* to remove proposals that are very similar (e.g. in space and orientation). We show that this pruned dynamic programming algorithm performs rapidly on benchmarked datasets and yields high performance (see results section).

Finally, we *learn* the model parameters, which specify both the geometry and appearance variability. We design the graph structure by hand (which takes a few days) but our more recent work [50] suggests how the graph structure without OR nodes could be learnt in an unsupervised manner (i.e. performing "structure induction" in machine learning terminology). To learn the parameters we extend the max-margin structure learning algorithm [1,39,40] to HCDTs – AND/OR graphs – (these papers extended max-margin learning from classification to estimating the parameters of structured models such as stochastic context free grammars). Max-margin learning is a supervised approach where the groundtruth of parse tree is given for training. Max-margin learning is global in the sense that we learn all the parameters simultaneously (by an algorithm that is guaranteed to find the global minimum) rather than learning local subsets of the parameters independently. The discriminative nature of max-margin learning means that it is often more effective than standard maximum likelihood estimation when the overall goal is classification (e.g. into different poses). It also has some technical advantages such as: (i) avoiding the computation of the partition function of the distribution, and (ii) the use of the kernel trick to extend the class of features. We perform some pre-processing to estimate potentials which are then fed into the max-margin procedure. We note that max margin learning requires an efficient inference algorithm – and is only possible for HCDTs graphs because we have an efficient compositional inference algorithm. The shorter vision of this learning has appeared in our previous work [51].

In summary, our paper makes contributions to both machine learning and computer vision. The contribution to machine learning is the AND/OR graph representation (with the summarization principle), the compositional inference algorithm, and the extension of max-margin learning to AND/OR graphs. The contribution to computer vision is HCDT model for deformable objects with multiple configurations and its application to model objects such as horses and humans. Our experimental results, see section (6), show that we can perform rapid inference and achieve state of the art results on difficult benchmarked datasets.

2 Background

To describe the computer vision background we first need to emphasize that an HCDT is able to perform several different visual tasks which are often studied independently in the literature. Recall that the compositional inference will output a parse tree which includes a specification of which subparts of the object are present (i.e. the topology), and the positions and properties of these subparts. An HCDT can perform: (i) the *pose/configuration estimation task* by outputting the estimated topology, (ii) the *segmentation task* by outputting the states of the leaf nodes (and some postprocessing to fill in the gaps), (iii) the *parsing task* by detecting the positions and properties of the sub-parts (by the nodes above the leaves), and (iv) the *weak-detection task* of detecting where the object is located in an image (but knowing that the object is present). We use "weak-detection" to distinguish this from the much harder "strong-detection" task which requires estimating whether or not an object is present in the image.

2.1 Object Representation

Detection, segmentation and parsing are all challenging problems. Most computer vision systems only address one of these tasks. There has been influential work on weak-detection [11] and on the related problem of registration [8,2]. Work on segmentation includes [18,22,4,12,23,45], and [33].

Much of this work is formulated, or can be reformulated, in terms of probabilistic inference. But the representations are fixed graph structures defined at a single scale. This restricted choice of representation enables the use of standard inference algorithms (e.g., the hungarian algorithm, belief propagation) but it puts limitations on the types of tasks that can be addressed (e.g. it makes parsing impossible), the number of different object configurations that can be addressed, and on the overall performance of the systems.

In the broader context of machine learning, there has been a growing use of probabilistic models defined over variable graph structures. Important examples include stochastic grammars which are particularly effective for natural language processing [24].

In particular, vision researchers have advocated the use of probability models defined over AND/OR graphs [6],[17] where the OR nodes enable the graph to have multiple topological structures. Zhu and Mumford [52] have made a strong case for the use of AND/OR graphs in vision and given practical examples. Similar AND/OR graphs have been used in other machine learning problems [15].

But the representational power of AND/OR graphs comes at the price of increased computational demands for performing inference (and learning). For one dimensional problems, such as natural language processing, this can be han-

dled by dynamic programming. But computation becomes considerably harder for vision problems and it is not clear how to efficiently search over the large number of configurations of an AND/OR graph. The inference problem simplifies significantly if the OR nodes are restricted to lie at certain levels of the graph (e.g. [25], [47]), but these simplifications are not suited to the problem we are addressing. These models have OR nodes only at the top level which means they are like conventional mixture distributions and do not have re-usable parts.

2.2 Human Body Parsing

There has been considerable recent interest in human body parsing. Sigal and Black [36] address the occlusion problem by enhancing appearance models. Triggs and his colleagues [34] learn more complex models for individual parts by SVM and combine them by an extra classifier. Mori [27] use super-pixels to reduce the search space and thus speed up the inference. Ren et al. [32] present a framework to integrate multiple pairwise constraints between parts, but their models of body parts are independently trained. Ramanan [31] proposes a tree structured CRF to learn a model for parsing human body. Lee and Cohen [21] and Zhang et al. [46] use MCMC for inference. In summary, these methods involve representations of limited complexity (i.e. with less varieties of pose than AND/OR graphs). If learning is involved, it is local but not global (i.e. the parameters are not learnt simulateously) [32,36,34,27]. Moreover, the performance evaluation is performed by the bullseye criterion: outputting a list of poses and taking credit if the groundtruth result is in this list [27,46,38].

The most related work is by Srinivasan and Shi [38] who introduced a grammar for dealing with the large number of different poses. Their model was manually defined, but they also introduced some learning in a more recent paper [37]. Their results are the state of the art, so we make comparisons to them in section (6).

By contrast, our model uses the AND/OR graph in the form of Chen et al. [7] which combines a grammatical component (for generating multiple poses) with a markov random field (MRF) component which represents spatial relationships between components of the model (see [17,6,52] for different types of AND/OR graph models). We perform global learning of the model parameters (both geometric and appearance) by max-margin learning. Finally, our inference algorithm outputs a single pose estimate which, as we show in section (6), is better than any of the results in the list output by Srinivasan and Shi [38] (and their output list is better than that provided by other algorithms [27]).

2.3 Max Margin Structure Learning

The first example of max-margin structure learning was proposed by Altun et al. [1] to learn Hidden Markov Models (HMMs) discriminatively. This extended the max margin criterion, used in binary classification [42] and multi-class classification [13], to learning structures where the output can be a sequence of binary vectors (hence an extension of multi-class classification to cases where the number of classes is 2^n, where n is the length of the sequence). We note that there have been highly successful examples in computer vision of max-margin applied to binary classification, see SVM-based face detection [29].

Taskar et al. [39] generalized max margin structure learning to general markov random fields (MRF's), referred to a max margin markov network (M^3). Taskar et al. [40] also extended this approach to probabilistic context-free grammar (PCFG) for language parsing. But max-margin learning has not, until now, been extended to learning AND/OR graph models which can be thought of as combinations of PCFG's with MRF's.

This literature on max-margin structure learning shows that it is highly competitive with conventional maximum likelihood learning methods as used, for example, to learn conditional random fields (CRF's) [19]. In particular, max-margin structure learning avoids the need to estimate the partition function of the probability distribution (which is major technical difficulty of maximum likelihood estimation). Max-margin structure learning essentially learns the parameters of the model so that the groundtruth states are those with least energy (or highest probability) and states which are close to groundtruth also have low energy (or high probability). See section (5) for details.

3 The Hierarchical Configural Deformable Template

We represent an object by a hierarchical graph defined by parent-child relationships, illustrated in figures (2) and (3). The state variables defined at the top node of the hierarchy represents the position and properties (position, orientation, and scale) of the center of the object. The state variables at the leaf nodes and the intermediate nodes represent the position and properties of points on the object boundary and of subparts of the object respectively. Nodes are classified into three types depending on their type of parent-child relationships: (i) "AND" nodes have fixed parent-child relationships and their states are compositions of their children's states, (ii) "OR" nodes are switchable [52] which enables the parent node to choose one of its children thereby affecting the topology of the graph, (iii) the "LEAF" nodes are connected directly to the image. The children of AND nodes are related by sideways connections which enforce spatial relationships on the states of the child nodes.

The graph structure is specified so that AND and OR nodes appear in neighboring levels with the top node being an AND node and the LEAF nodes being children of AND nodes. There are 98 possible configurations/parse trees for the human body, some of which are shown in figure (2). The subparts include the torso, the left leg, and the right leg of the body (see circular nodes in the second row). Each of these subparts is built out of sub-subparts, shown by the AND nodes (rectangles in the third row), which are selected by the OR nodes.These are, in turn, composed by AND-ing more elementary subparts (see fourth and fifth row).

The graph structure was hand-specified as follows. First we decompose the object into natural parts – e.g. the horse is decomposed into the head, upper body, front legs and back legs, see second row of figure (3). The horse is an 'AND' of these four parts. Next we consider all possible 'heads' that occur in the dataset and group them into a small number of 'types' (between 3 and 6). The 'head' is an OR of these different types. Heads within each type are required to be similar and to share similar subparts. For example, the third node from the left in the third row of figure (3), shows two heads of the same type which share two subparts, see fourth row. We repeat the process by considering the exemplars of all these subparts and dividing them up into types. Alternatively we can think of this as a bottom-up process which combines subparts (fifth row) by AND-ing and OR-ing to form larger parts (third row) which are combined by ADD-ing and OR-ing to form the object (first row). Hence we can combine the most elementary parts – at the leaf nodes – by composition (AND-ing) and selection (OR-ing) to build objects with varying poses and viewpoint. A similar strategy is used to obtain a graph structure for the baseball player, see figure (2). We decompose the baseball player into torso and head, left leg, and right leg – i.e. the baseball player is an 'AND' of these three parts – and then we proceed as for horses. Note that the arms were not labeled in the baseball dataset [27] so we do not model them in our graph. It took us about three days to determine the graph structure for humans and horses. After determining the structure it takes three minutes to hand-label an image.

The graph structures are specified as follows. For the baseball player, see figure (2), there are seven levels starting at the root node. The nodes at the following levels are given by: (i) 94 LEAF nodes at level 1, (ii) 94 OR nodes at level 2, (iii) 24 AND nodes at level 3, (iv) 14 OR nodes at level 4, (v) 10 AND nodes at level 5, (vi) 3 OR nodes at level 6, and (vii) 1 AND node at level 7. The horse model in figure (3) also has seven levels. The nodes are specified by: (i) 68 LEAF nodes at level level 1, (ii) 68 OR nodes at level 2, (iii) 27 AND nodes at level 3, (iv) 24 OR nodes at level 4, (v) 8 AND nodes at level 5, (vi) 4 OR nodes at level 6, and (vii) one AND node at level 8.

We define a probability distribution over the state variables of the graph which is specified by potentials defined over the cliques. These cliques are specified by the parent-child relations for the OR and AND nodes and between the children of the AND nodes. In addition, there are potentials for the appearance terms linking the LEAF nodes to the image. This probability distribution specifies the configuration of the object including its topology, which will correspond to the pose of the object in the experimental section. The probability distribution over the AND/OR graph is a mixture model where the number of mixture components is determined by the number of different topologies of the graph. But the AND/OR graph is much more compact than traditional mixture distributions because it uses re-usable parts [17] which are shared between mixtures. This compactness enables a large number of mixture distributions to be represented by a graph with a limited number of nodes, see subsection (3.3). As we will describe later in section (4) this leads to efficient inference and learning.

3.1 The Structure of the AND/OR Graph

Notation	Meaning
V	the set of nodes
E	the set of edges
$V^{AND}, V^{OR}, V^{LEAF}$	sets of "AND", "OR" and "LEAF" nodes
μ, ν, ρ, τ	the index of nodes
Ch_ν	the children of node ν
z_ν	the state of node ν
t_ν	the switch variable for node ν
$V(t)$	the set of "active" nodes
z_{t_ν}	the state of the "active" child node
$(\mathbf{x}_\nu, \theta_\nu, s_\nu)$	position, orientation and size
z_{Ch_ν}	the states of child nodes of ν
$\mathbf{I}$	input image
$y = (z, t)$	the parse tree
m	the max num. of children of AND nodes
n	the max num. of children of OR nodes
h	the num. of layers containing OR nodes
$\alpha_\mu^{AND}, \alpha_\mu^{OR}, \alpha_\mu^D$	the parameters
$\phi^{AND}, \phi^{OR}, \phi^D$	the potentials

Table 1 The terminology used in the HCDT model.

We use the notations listed in table (1) to define the HCDT. Formally, an HCDT is a graph $G = (V, E)$ where V is the set of nodes (vertices) and E is the set of edges (i.e., nodes $\mu, \nu \in V$ are connected if $(\mu, \nu) \in E$)). The edges are defined by the parent-child structure where Ch_ν denotes the children of node ν. The graph is required to be tree-like so that each node has only one parent – e.g., $Ch_\nu \bigcap Ch_\mu = \emptyset$ for all $\nu, \mu \in V$.

There are three types of nodes,"OR","AND" and "LEAF" nodes which specify different parent-child relationships. These are depicted in figure (1) by circles, rectangles and triangles

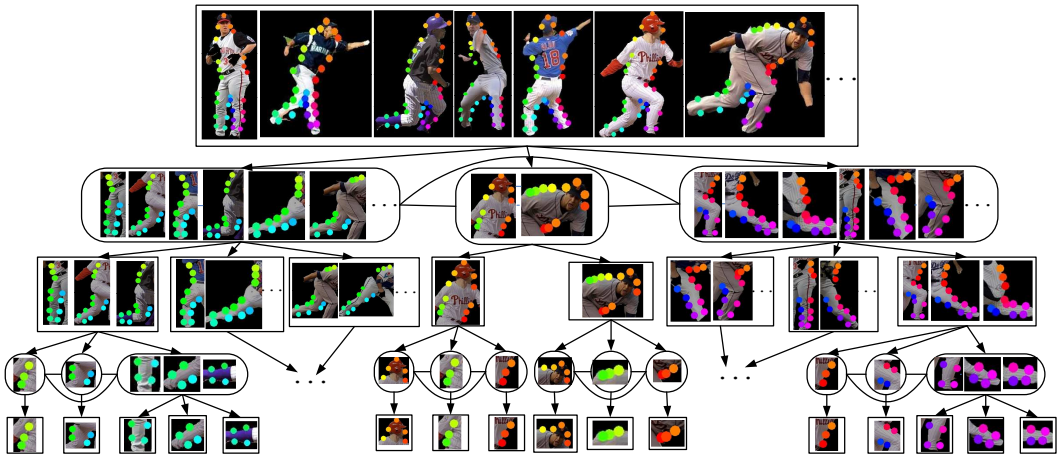

Fig. 2 The HCDT uses an AND/OR graph as an efficient way to represent different configurations of an object by selecting different subparts. The graph in this figure, and the next, were both hand designed (taking a few days). The leaf nodes of the graph indicates points on the boundary of the object (i.e. the baseball player). The higher levels nodes represent the positions and properties of sub-parts of the object, and whether the sub-parts are used. The HCDT for baseball players uses eight levels, but we only show five for reasons of space. Color is used to distinguish different body parts. The arms are not modeled in this paper (or in related work in the literature due to the nature of the database).

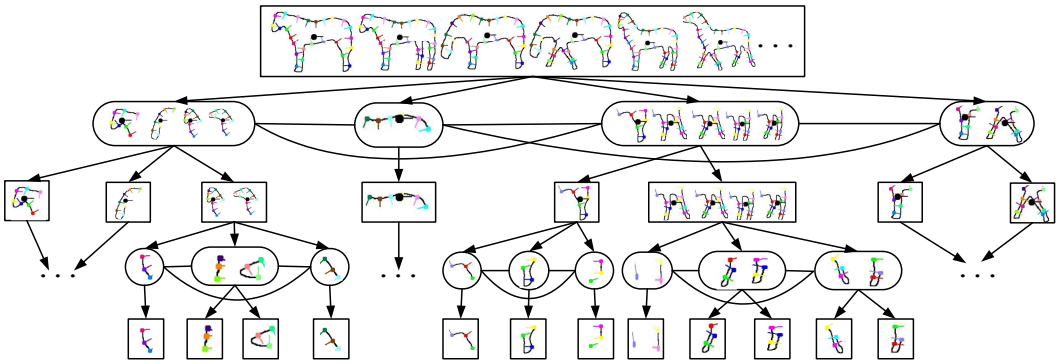

Fig. 3 The HCDT AND/OR graph for horses which has 40 different topologies. The first pose (far left of top row) is the most common and is used for the fixed hierarchy model (whose performance is compared to HCDT).

respectively. LEAF nodes have no children, so $Ch_\nu = \emptyset$ if $\nu \in V^{LEAF}$. The children Ch_ν of AND nodes $\nu \in V^{AND}$ are connected by horizontal edges defined over all triplets (μ, ρ, τ) (i.e. we have horizontal edges for all triplets $\mu, \rho, \tau \in Ch_\nu$ if $\nu \in V^{AND}$). These connections between the child nodes means that the graph does contain some closed loops, but the restriction that each child node has a single parent means that we can convert the graph into a tree by merging the states of all child nodes into a single node as done in the junction tree algorithm [20] (this will enable our inference algorithm). The children Ch_ν of OR nodes $\nu \in V^{OR}$ are not connected. We let V^{LEAF} denote the leaf nodes. See the examples in figures (2) and (3).

3.2 The State Variables

The state variables of the graph specify the graph topology and the positions and properties of the subparts of the object. The graph topology is specified by a variable t which indicates the set of nodes $V(t)$ which are *active* (i.e. which component of the mixture model). The topologies are determined by the switches at the OR nodes. Hence, starting from the top level, an active OR node $\nu \in V^{OR}(t)$ must select a child $t_\nu \in Ch_\nu$, where t_ν is a switch variable, see figure (2). The topology is changed by an active OR node "switching" to a different child. The active nodes $\nu \in V(t)$ also have states $z_\nu = (\mathbf{x}_\nu, \theta_\nu, s_\nu)$ which specify the position $\mathbf{x}_\nu$, orientation θ_ν, and size s_ν of the subpart of the object. The states of the active AND and LEAF nodes $\nu \in V^{AND}(t) \bigcup V^{LEAF}(t)$ are specified by z_ν, and the states of the active OR nodes are specified by (z_ν, t_ν). In summary, we specify the state of the tree by the states $y = \{(z_\nu, t_\nu) : \nu \in V^{OR}(t)\} \bigcup \{z_\nu : \nu \in V^{AND}(t) \bigcup V^{LEAF}(t)\}$ where the active nodes $V(t)$ are determined from the $\{t_\nu : \nu \in V(t)\}$ by recursively computing from the top node, see figure (2). We let $z_{Ch_\nu} = \{z_\mu : \mu \in Ch_\nu^{AND}\}$ denote the states of all the child nodes of an AND node $\nu \in V^{AND}$. We let z_{t_ν} denote the state of the selected child node of an OR node $\nu \in V^{OR}$.

Observe that the node variables z_ν take the same form at all levels of the hierarchy. This relates to the *summarization principle*, see section (3.4), and means that the state of parent nodes are simple deterministic function of the state variables of the children, see section (3.5). The use of the summarization principle is exploited in a later section (4) to design an efficient inference algorithm.

The task of parsing an object from an input image - i.e. determining the parse tree – requires estimating the state variables. This includes determining the graph topology, which corresponds to the object configuration (e.g. the pose of a human), and the positions and properties of the active nodes. Different vision tasks are "solved" by different state variables. The segmentation task is solved by the state variables $\{z_\nu \in V^{LEAF}(t)\}$ of the active LEAF nodes (followed by some post-processing). The parsing task is to determine the positions and properties of specific subparts of the object. In the experimental section we evaluate it by determining that states of the LEAF nodes $\{z_\nu \in V^{LEAF}(t)\}$ (i.e. the smallest sub-parts), but we could also include the higher level nodes $\{z_\nu \in V^{AND}(t) \bigcup V^{OR}(t)\}$. The *configuration estimation* task is determined by estimating the topology t of the graph and is used, for example, to determine the pose of humans playing baseball.

3.3 The Compactness of the AND/OR Graph Representation

The AND/OR graph used by HCDTs is a very powerful, but compact, representation of objects since it allows many different topologies/parse trees which can be used for describing different object configurations (e.g. different poses of baseball players). This efficiency occurs because different configurations share subparts. This enables an enormous number of topologies to be represented with only a limited number of nodes and with a limited number of cliques. This compactness is of great help for inference and learning. It enables rapid inference since we can search over the different topologies very efficiently by exploiting their shared subparts. Similarly the sharing of subparts, and hence the limited number of parameters required to specify all the clique potentials, enables us to achieve good learning performance while requiring comparatively little training data.

The following argument illustrates the compactness of the HCDT by giving an upper bound for the number of different topologies/parse trees and showing that the number of topologies becomes exponentially larger than the number of nodes. We calculate the number of topologies and the number of nodes assuming that all OR and AND nodes have a fixed number of children n and m respectively. These calculations become upper bounds for HCDTs where the number of child nodes are variable provided we set $n = \max_{\nu \in V^{OR}} |Ch_\nu|$ and $m = \max_{\nu \in V^{AND}} |Ch_\nu|$ (where $|Ch_\nu|$ is the size of the set Ch_ν of children of ν). We can convert these into lower bounds by replacing $\max_\nu$ with $\min_\nu$.

With these assumptions, we calculate (or bound) the number of topologies by $n^{(m^h)}n^{(m^{h-1})}...n^m$ which is of order n^{m^h}. The calculation proceeds by induction recalling that each topology (parse tree) corresponds to a specification of the switch variables $\{t_\nu\}$ which select the children of the OR nodes. Suppose g_h is the number of topologies for an HCDT with h OR levels. Then adding a new level of OR and AND nodes gives the recursive formula $g_{h+1} = \{ng_h\}^m$, with $g_0 = 1$, and the result follows.

Similarly, we can calculate (or bound) the number of nodes by the series $1+m+mn+m^2n+...(mn)^h$ which can be summed to obtain $\{(1+n)m^{h+1}n^h - (1+m)\}/\{mn-1\}$, which is of order $O(\{nm\}^h)$. *Hence the ratio of the number of topologies to the number of nodes is of order $n^{m^h}/\{nm\}^h$ which becomes enormous very rapidly as h increases.*

By comparison, mixture models which do not use part-sharing will require far more nodes to represent an equivalent number of mixtures (equating each mixture with a topology). The ratio of the number of nodes used (by all components) to the number of mixture components is simply the average number of nodes for each mixture component and will be finite and large. Hence, unless the mixture components are very simple, standard mixture models will usually have a much larger ratio of nodes to mixtures/topologies than HCDTs.

We stress that a small node-to-topology ratio not only gives a compact representation but also enables efficient learning and inference. The number of parameters to be learnt depends on the total number of nodes and the size of the cliques. Hence keeping the node-to-topology ratio small will reduce the number of parameters to be learnt (assuming that the number of topologies/object configurations is fixed) and hence simplify the learning and reduce the amount of training data required. Moreover, the complexity of the inference algorithm depends on the number of nodes and hence is also reduced by keeping the node-to-topology ratio as small as possible.

For the baseball player HCDT in the results section we have $m = 4, n = 3, h = 4$, $|V^{LEAF}| = 94$, $|V^{OR}| = 112$, $|V^{AND}| = 129$. The number of the active LEAF nodes is always 27. The total number of different toplogies/parse trees is 98. Note that for this HCDT the bounds on the number of nodes is not tight – the bounds give a total of $(144)^2$ nodes which the total number of nodes is 335. So, in practice, the number of nodes does not need to grow exponentially with the number of layers.

3.4 The Summarization Principle and HCDT

We now describe the summarization principle in more detail. This principle is critical to make the HCDT computationally tractable – i.e. to make inference and learning efficient. It was not used in our early work on hierarchical modeling [48] or in alternative applications of AND/OR graphs to vision [6], [17],[52].

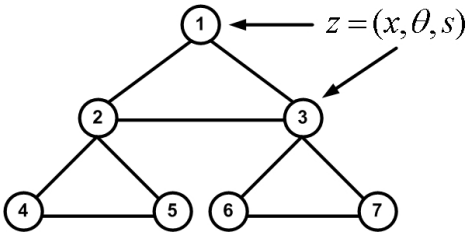

Fig. 4 The Summarization Principle for a simple model. All nodes shown are active and nodes $1, 2, 3$ are AND nodes. See text for details.

In figure (4), we show a simple parse graph keeping only the nodes which have been selected by the topology variable t, hence all the nodes shown are active. Each node has a state variable $z = (x, \theta, s)$. The summarization principle has four aspects:

(i) The state variable of node ν is the summary of the state variables of its children – e.g., the state of node 2 is a summary of the states of nodes 4 and 5. In this paper, the state variables of the parent are deterministic functions of the state variables of its active children (see next subsection for details).

(ii) The clique potentials defined for the clique containing nodes $1, 2, 3$ do not depend on the states of their child nodes $4, 5, 6, 7$ (e.g., the Markov property).

(iii) The representational complexity of the node is the same at all levels of the tree. More specifically, the state variable at each node has the same low-dimensional representation which does not grow (or decrease) with the level of the node or the size of the subpart that the node represents.

(iv) The potentials defined over the cliques are simple since they are defined over low-dimensional spaces and use simple statistics (see next subsection).

As we will describe in section (4), the use of the summarization principle enables polynomial time inference using dynamic programming (with a junction tree representation). The complexity of inference is partially determined by the clique size which has been designed to be small (but due to the size of the state space we need to perform pruning). The learning is practical because of: (1) the small clique size, (2) the potentials depend on simple statistics of the state variables (which are the same at all levels of the hierarchy), and (3) the efficiency of the inference algorithm, which is required during learning to compare the performance of the model, with its learnt values of the parameters, to the groundtruth.

3.5 The Probability Distributions: the potential functions

We now define a conditional probability distribution for the state variables y conditioned on the input image $\mathbf{I}$. This distribution is specified in terms of potentials/statistics defined over the cliques of the AND/OR graph.

The conditional distribution on the states and the data is given by a Gibbs distribution:

$$P(y|\mathbf{I};\boldsymbol{\alpha}) = \frac{1}{Z(\mathbf{I};\boldsymbol{\alpha})} \exp\{-\boldsymbol{\alpha} \cdot \boldsymbol{\Phi}(\mathbf{I}, \mathbf{y})\}. \tag{1}$$

where $\mathbf{I}$ is the input image, y is the parse tree, $\boldsymbol{\alpha}$ denotes model parameters (which will be learnt), $\Phi(\mathbf{I}, \mathbf{y})$ are potentials and $Z(\mathbf{I}, \boldsymbol{\alpha})$ is the partition function.

Let $E(y|\mathbf{I})$ denote the energy, i.e. $E(y|\mathbf{I}) = \boldsymbol{\alpha} \cdot \boldsymbol{\Phi}(\mathbf{I}, \mathbf{y})$. The energy $E(y|\mathbf{I})$ is specified by potential defined over cliques of the graph and data terms. This gives three types of potential terms: (i) cliques with AND node parents, (ii) cliques with OR node parents, and (iii) LEAF terms (i.e. data terms). Hence we can decompose the energy into three terms:

$$\begin{aligned} E(y|\mathbf{I}) = & \sum_{\mu \in V^{AND}(t)} \boldsymbol{\alpha}_{\mu}^{AND} \cdot \boldsymbol{\phi}^{AND}(z_{\mu}, z_{Ch_{\mu}}) \\ & + \sum_{\mu \in V^{OR}(t)} \boldsymbol{\alpha}_{\mu}^{OR} \cdot \boldsymbol{\phi}^{OR}(z_{\mu}, t_{\mu}, z_{t_{\mu}}) \\ & + \sum_{\mu \in V^{LEAF}(t)} \boldsymbol{\alpha}_{\mu}^{D} \cdot \boldsymbol{\phi}^{D}(\mathbf{I}, z_{\mu}). \end{aligned} \tag{2}$$

The first two terms – the clique terms – are independent of the data and can be considered as priors on the spatial geometry of the HCDT. These clique terms are defined as follows.

The cliques with AND node parents can be expressed by *horizontal* and *vertical* terms. The *horizontal terms* specify spatial relationships between triplets of the child node which are invariant to scale and rotation. For each triplet (ν, ρ, τ) such that $\nu, \rho, \tau \in Ch_{\mu}$ we specify the invariant shape vector ITV [49] $l(z_{\nu}, z_{\rho}, z_{\tau})$, see figure (5), and define a potential $\phi_H^{AND}(z_{\nu}, z_{\rho}, z_{\tau})$ to be Gaussian (i.e. the first and second order statistics of $l(z_{\nu}, z_{\rho}, z_{\tau})$). Recall that $l(z_{\nu}, z_{\rho}, z_{\tau})$ is independent of the scale and rotation of the points $z_{\nu}, z_{\rho}, z_{\tau}$ and so the HCDT is independent of scale and rotation. Denote $Tri(\mu)$ to be the set of triples of children of node μ – i.e. $Tri(\mu) = \{(\nu, \rho, \tau) : \nu, \rho, \tau \in Ch_{\mu}\}$. This gives *horizontal potential terms*:

$$\sum_{(\nu,\rho,\tau) \in Tri(\mu)} \alpha_{\mu}^{AND}(\nu, \rho, \tau) \phi_H^{AND}(l(z_{\nu}, z_{\rho}, z_{\tau})), \tag{3}$$

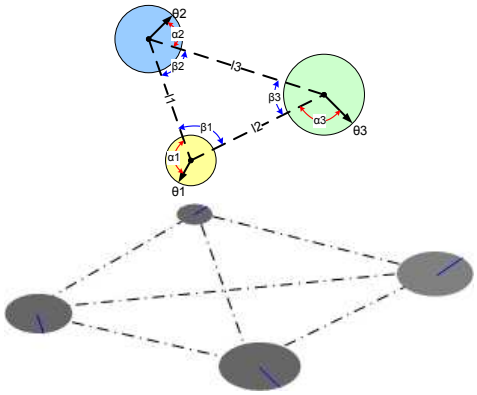

Fig. 5 The horizontal terms for the AND clique potentials. The first panel demonstrates the invariant shape vector (ITV) constructed from the positions and properties $z = (\mathbf{x}, \theta, s)$ of a triplet of nodes (which are children of AND nodes). The ITV $\mathbf{1}(z_1, z_2, z_3)$ depends only on properties of the triplet – such as the internal angles, the ratios of the distances between the nodes, the angles between the orientations at the nodes (i.e. the θ's) and the orientations of the segments connecting the nodes – which are invariant to the translation, rotation, and scaling of the triplet. The second panel show how the triplets are constructed from all child nodes of an AND node. In this example, there are four child node and so four triplets are constructed. Each circle corresponds to a child node with property $(\mathbf{x}, \theta, s)$. The potentials for each triplet are of Gaussian form defined over the ITV's of the triplets – the ϕ^{AND} corresponds to the sufficient statistics of the Gaussian and the parameters α^{AND} will be learnt.

where there are coefficients $\alpha_{\mu}^{AND}(\nu, \rho, \tau)$ for each triplet.

The *vertical terms* relate the state variable z_{μ} of the parent node to the state variables of its children $z_{Ch_{\mu}} = \{z_{\nu} : \nu \in Ch_{\mu}\}$ by a deterministic function. This is defined by $f(z_{Ch_{\mu}}) = (\mathbf{x}_{Ch_{\mu}}, \theta_{Ch_{\mu}}, s_{Ch_{\mu}})$, where $\mathbf{x}_{Ch_{\mu}}, \theta_{Ch_{\mu}}$ are the averages of the positions and orientations of the states $\mathbf{x}_{\nu}, \theta_{\nu}$ of the children $\nu \in Ch_{\mu}$, and $s_{Ch_{\mu}}$ is the size of the region determined by the $z_{Ch_{\mu}}$.

The vertical potential terms are defined to be $\phi_V^{AND}(z_{\mu}, z_{Ch_{\mu}}) = \delta(z_{\mu}, f(z_{Ch_{\mu}}))$ (where $\delta(z_{\mu}, f(z_{Ch_{\mu}})) = 0$ if $z_{\mu} = f(z_{Ch_{\mu}})$, and $\delta(.,.)$ takes an arbitrarily large value otherwise). This eliminates state configurations where the deterministic relation between the parents and children are not enforced.

The clique with OR node parents are expressed by a vertical term which puts a probability on the different switches that can be performed, and constrains the state (i.e. position and properties) to be those of the selected child. This gives a potential $\phi^{OR}(z_{\mu}, t_{\mu}, z_{t_{\mu}}) = \alpha_{t_{\mu}} + \delta(z_{\mu}, f(z_{t_{\mu}}))$, where $\alpha_{t_{\mu}}$ species the probabilities of the switches.

The *data terms* $\phi^D(\mathbf{I}, z_{\mu})$ are specified only at LEAF nodes $\mu \in V^{LEAF}(t)$ and are defined in terms of a dictionary of potentials computed from image features $F(\mathbf{I}, D(z_{\mu}))$ computed from regions in the image $\mathbf{I}$ specified by $D(z_{\mu})$ (i.e. local "evidence" for a LEAF state z_{μ} is obtained by computing image features from the domain $D(z_{\mu})$ which surrounds z_{μ}). More precisely, the potentials are of form

$$\phi^D(\mathbf{I}, z_{\mu}) = \log \frac{P(F(\mathbf{I}, D(z_{\mu}))|object)}{P(F(\mathbf{I}, D(z_{\mu}))|background)}. \quad (4)$$

The distributions $P(F(\mathbf{I}, D(z_{\mu}))|object)$ and $P(F(\mathbf{I}, D(z_{\mu}))|background)$ are the distributions of the feature response $F(.)$ conditioned on whether z_{μ} is on the object or in the background. They are univariate Gaussian distributions. There are 37 features in total including: (i) the intensity (1 feature), (ii) the intensity gradient represented both by the (x, y) components and by the magnitude and orientation at four scales (16 features), (iii) the Canny edge detector at four scales $(1, 2, 4, 8)$ (4 features), (iv) the Difference of Gaussian at four scales (7*7, 9*9,19*19 and 25*25) (4 features), (v) the Difference of Offset Gaussian (DOOG) at two scales (13*13 and 22*22) and six orientations $(0, \frac{1}{6}\pi, \frac{2}{6}\pi, ...)$ (12 features).

4 The Inference/Parsing Algorithm

We now describe our inference algorithm which is designed to exploit the hierarchical structure of the HCDT. Its goal is to obtain the best state y^* by estimating $y^* = \arg\max P(y|\mathbf{I}; \boldsymbol{\alpha})$ which can be re-expressed in terms of minimizing the energy function:

$$y^* = \arg\min_{y} \{\boldsymbol{\alpha} \cdot \boldsymbol{\Phi}(\mathbf{I}, \mathbf{y})\}, \quad (5)$$

where the energy $\boldsymbol{\alpha} \cdot \boldsymbol{\Phi}(\mathbf{I}, \mathbf{y})$ is given by the three terms in equation (2).

To perform inference, we observe that the hierarchical structure of the HCDT, and the lack of shared parents (i.e., the independence of different parts of the tree), means that we can express the energy function recursively and hence find the optimum y using dynamic programming (DP). But although DP is guaranteed to be polynomial in the relevant quantities (number of layers and graph nodes, the size of state space of the node variables z and t) full DP is too slow because of the large size of the state space – every sub-part of the object can occur in any position of the image, at any orientation, and any scale (recall that $z_{\nu} = (\mathbf{x}_{\nu}, \theta_{\nu}, s_{\nu})$) so the states of the nodes depend on the image size, the number of allowable orientations, and the allowable range of sizes of the sub-parts). Hence, as in other applications of DP or BP to vision [10,11] we must perform pruning to reduce the set of possible states and ensure that the algorithms converges rapidly.

We call the algorithm *compositional inference* [48,7], see table (6). It has a bottom-up pass which starts by estimating possible states, or *proposals*, for the leaf nodes and proceeding to estimate states/proposals for the nodes higher up the tree, hence determining the lowest energy states of the tree (subject to pruning). The top-down pass (not shown

here) selects the optimal states of the nodes from the pruned states (the bottom-up and top-down passes are analogous to the forward and backward passes in standard DP). We have experimented with modifying the top-down pass to deal with errors which might arise from the pruning but, in our experience, such errors occurred rarely so we do not report it here (the modification involved keeping additional clusters of proposals which were not propagated upward in the bottom-up pass but which could be accessed in the top-down pass – see [48]).

The pruning is used in the bottom-up pass to restrict the allowable states of a node μ to a set of *proposals* (borrowing terminology from the MCMC literature) which are represented including their energies. These proposals are selected by two mechanisms: (i) *energy pruning* - to remove proposals corresponding to large energy, and (ii) *surround suppression* - to suppress proposals that occur within a surrounding window in z (similar to non-maximal suppression). Intuitively, the first pruning mechanism rejects high energy configurations (which is a standard pruning technique for DP – see [10,11]) while the second mechanism rejects states which are too similar. The second pruning mechanism is less commonly used but the errors it may cause are fairly benign – e.g. it may make a small error in the position/orientation/size of a sub-part but would not fail to detect the subpart.

The pruning threshold, and the window for surround suppression, must be chosen to that there are very few false negatives (i.e. the object is always detected as, at worst, a small variant of one of the proposals). In practice, the window is $(5, 5, 0.2, \pi/6)$ (i.e., 5 pixels in the x and y directions, up to a factor of 0.2 in scale, and $\pi/6$ in orientation – same for all experiments). Rapid inference is achieved by keeping the number of proposals small. We performed experiments to balance the trade-off between performance and computational speed. Our experiments show good scaling, see experimental section (6).

Compositional inference has an intuitive interpretation since it starts by detecting states/proposals for the low-level sub-parts of the object and proceeds to combine these together to give proposals for the higher level states. It is reminiscent of constraint satisfaction since the proposals for nodes at low levels in the tree are obtained only by using the energy potentials for the corresponding subpart of the tree (i.e. the subtree whose root node is the node for which we are making the proposals), but the proposals for nodes at higher levels involves imposes more energy potentials (ensuring that the higher level subparts have plausible spatial relationships).

We now specify compositional inference precisely by first specifying how to recursively compute the energy function – which enables dynamic programming – and then describe the algorithm including the approximations (energy pruning and surround suppression) made to speed up the algorithm. Pseudocode for compositional inference is specified in figure (6).

Input: $\{p^1_{\nu^1}\}$. Output:$\{p^L_{\nu^L}\}$

- Bottom-Up(p^1)

  Loop : $l = 1$ to L, for each node ν at level l

  – If ν is an OR node:

  1. Union $\{p^l_{\mu,b}\} = \bigcup_{\rho \in Ch_\mu} p^{l-1}_{\rho,a}$

  – If ν is an AND node:

  1. Composition: $\{p^l_{\nu,b}\} = \oplus_{\rho \in Ch_\nu, a} p^{l-1}_{\rho,a}$
  2. Pruning by Energy: $\{p^l_{\nu,a}\} = \{p^l_{\nu,a} | \boldsymbol{\alpha} \cdot \Phi_\nu(\mathbf{I}, p^l_{\nu,a}) > Th_l\}$
  3. Surround Suppression: $\{(p^l_{\nu,a}\} = LocalMaximum(\{p^l_{\nu,a}\}, \epsilon_W)$ where ϵ_W is the size of the window W^l_ν defined in space, orientation, and scale.

Fig. 6 The inference algorithm. $\oplus$ denotes the operation of combining proposals from child nodes to make proposals for parent nodes.

Recursive Formulation of the Energy The HCDT is specified by a Gibbs distribution, see equations (1,2). We exploit the tree-structure to express this energy function recursively by defining an energy function $E_\nu(y_{des(\nu)}|\mathbf{I})$ over the subtree with root node ν in terms of the state variables $y_{des(\nu)}$ of the subtree – where $des(\nu)$ stands for the set of descendent nodes of ν so $z_{des(\nu)} = \{z_\mu : \mu \in V_\nu(t)\}$ (For any node ν, we define V_ν to be the subtree formed by the set of descendent node with ν as the root node).

There are two different cases for this recursion depending on whether the level contains AND or OR nodes. We specify these, respectively, as follows:

$$E_\nu(y_{des(\nu)}|\mathbf{I}) = \sum_{\rho \in Ch_\nu} E_\rho(y_{des(\rho)}|\mathbf{I}) + \boldsymbol{\alpha}^{AND}_\nu \cdot \boldsymbol{\phi}^{AND}(z_\nu, z_{Ch_\nu}). \ \nu \in V^{AND}(t) \quad (6)$$

$$E_\nu(y_{des(\nu)}|\mathbf{I}) = E_{t_\nu}(y_{des(t_\nu)}|\mathbf{I}) + \alpha^{OR}_\nu \cdot \phi^{OR}(z_\nu, t_\nu, z_{t_\nu}) \quad (7)$$

Compositional Inference. *Initialization*: at each leaf node $\nu \in V^{LEAF}$ we calculate the states $\{p_{\nu,b}\}$ (b indexes the proposal) such that $E_\nu(p_{\nu,b}|\mathbf{I}) < Th$ (*energy pruning* with threshold Th) and $E_\nu(p_{\nu,b}|\mathbf{I}) \leq E_\nu(p_\nu|\mathbf{I})$ for all $z_\nu \in W(p_{\nu,b})$ (*surround suppression* where $W(p_{\nu,b})$ is a window centered on $p_{\nu,b}$). (The window $W(p_{\nu,b})$ is $(5, 5, 0.2, \pi/6)$ centered on $p_{\nu,b}$). We refer to the $\{p_{\nu,b}\}$ as proposals for the state z_ν and store them with their energies $E_\nu(p_{\nu,b}|\mathbf{I})$.

Recursion for parent AND nodes: to obtain the proposals for a parent node μ at a higher level of the graph $\mu \in V^{AND}$, we first access the proposals for all its child nodes $\{p_{\mu_i,b_i}\}$ where $\{\mu_i : i = 1, ..., |Ch_\mu|\}$ denotes the set of child nodes of μ and their energies $\{E_{\mu_i}(p_{\mu_i,b_i}|\mathbf{I}) : i = 1, ..., |Ch_\mu|\}$. Then we compute the states $\{p_{\mu,b}\}$ such that $E_\mu(p_{\mu,b}|\mathbf{I}) \leq$

$E_\mu(z_\mu|\mathbf{I})$ for all $z_\mu \in W(p_{\mu,b})$ where

$$E_\mu(p_{\mu,b}|\mathbf{I}) = min_{\{b_i\}}\{\sum_{i=1}^{|Ch_\mu|} E_{\mu_i}(z_{des(\mu_i,b_i)}|\mathbf{I}) + \boldsymbol{\alpha}_\mu \cdot \boldsymbol{\phi}^{AND}(p_{\mu,b}, \{p_{\mu_i,b_i}\})\} \quad (8)$$

In our experiments, the thresholds Th are set adaptively to keep the top K proposals ($K = 300$ in our experiments). In rare situations we may find no proposals for the state of one node of a triplet. In this case, we use the states of the other two nodes together with the horizontal potentials (geometrical relationship) to propose states for the node. A similar technique was used in [47].

Recursion for parent OR nodes: this simply requires enumerating all proposals (without composition) and pruning out proposals which have a weak energy score.

The number q of proposals of each node at different levels is linearly proportional in the size of the image. It is straightforward to conclude that the complexity of our algorithm is bounded above by $Mn^m q^m$. Recall that m is the maximum number of children of AND nodes (in this paper we restrict $m \leq 4$), n denotes the maximum number of possible children of OR nodes and M is the number of AND nodes connected to OR nodes ($M = 35$ for the baseball player). This shows that the algorithm speed is polynomial in n and q (and hence in the image size). The complexity for our experiments is reported in section (6).

5 Max Margin HCDT AND/OR Graph Learning

5.1 Primal and Dual Problems

The task of learning an HCDT is to estimate the parameters α from a set of training samples $(\mathbf{I}_1, y_1)$,...,$(\mathbf{I}_e, y_e)$ $\in \mathcal{I} \times \mathcal{Y}$ drawn from some fixed, but unknown probability distribution. In this paper, the set of examples $\{i : i = 1, ..., N_e\}$ corresponds to N_e images and N_e states of the HCDT. The learning is performed in a supervised manner where the states of parse tree which are labeled by hand are provided for training.

We formulate this learning task by using the max-margin criterion [42]. This is a discriminative criterion whose goal is to learn the parameters which are best for classification and be contrasted with maximum likelihood estimation (MLE). Max-margin is arguably preferable to MLE when the goal is discrimination and when limited amounts of data are available, see [42]. Standard max-margin was developed for binary classification – i.e. when y takes only two values – and so cannot be directly applied to HCDTs. But we build on recent work [51] which has extended max-margin to hidden markov models, markov models, and stochastic context free grammars [1],[39],[40]. A practical advantages of max-margin learning is that it gives a computationally tractable learning algorithm which, unlike MLE, avoids the need for computing the partition function $Z[\mathbf{I}, \alpha]$ of the distribution.

Max-margin learning seeks to find values of the parameters α which ensure that the energies $\boldsymbol{\alpha} \cdot \boldsymbol{\Phi}(\mathbf{I}, \mathbf{y})$ are smallest for the ground-truth states y and for states close to the ground-truth. To formulate the learning for HCDTs we first express the negative of the energy E_{neg} as a function of $\mathbf{I}, y, \boldsymbol{\alpha}$:

$$E_{neg}(\mathbf{I}, y, \boldsymbol{\alpha}) = \boldsymbol{\alpha} \cdot \boldsymbol{\Phi}(\mathbf{I}, y). \quad (9)$$

We define the *margin* γ of the parameter $\boldsymbol{\alpha}$ on example i as the difference between the true parse (groundtruth) y_i and the best parse y^*:

$$\gamma_i = E_{neg}(\mathbf{I}_i, y_i, \boldsymbol{\alpha}) - \max_{y \neq y_i} E_{neg}(\mathbf{I}_i, y, \boldsymbol{\alpha}) \quad (10)$$

$$= \boldsymbol{\alpha} \cdot \{\boldsymbol{\Phi}_{i,y_i} - \boldsymbol{\Phi}_{i,y^*}\} \quad (11)$$

where $\boldsymbol{\Phi}_{i,y_i} = \boldsymbol{\Phi}(\mathbf{I}_i, y_i)$ and $\boldsymbol{\Phi}_{i,y} = \boldsymbol{\Phi}(\mathbf{I}_i, y)$.

Intuitively, the size of the margin quantifies the confidence in rejecting an incorrect parse y. Larger margins [42] leads to better generalization and prevents over-fitting.

The *goal* of max margin learning of an HCDT is to maximize the margin subject to obtain correct results on the training data:

$$\max_{\gamma} \gamma \quad (12)$$

$$\text{s.t.} \quad \alpha \cdot \{\Phi_{i,y_i} - \Phi_{i,y}\} \geq \gamma L_{i,y}, \forall y; \quad \|\alpha\|^2 \leq 1; \quad (13)$$

where $L_{i,y} = L(y_i, y)$ is a loss function which penalizes incorrect estimate of y. This loss function gives partial credit to states which differ from the groundtruth by only small amounts (i.e. it will encourage the energy to be small for states near the groundtruth).

The loss function is defined as follows:

$$L(y_i, y) = \sum_{\nu \in V^{AND}(t^i)} \triangle(z^i_\nu, z_\nu) + \sum_{\nu \in V^{LEAF}(t^i)} \triangle(z^i_\nu, z_\nu) \quad (14)$$

where $\triangle(z^i_\nu, z_\nu) = 1$ if $dist(z^i_\nu, z_\nu) \geq \sigma$ or $\nu \notin V(t)$ (Note that if node $\nu \in V(t^i)$ from the ground truth is not active in $V(t)$, then the cost is set to 1, which penalizes configurations which have the "wrong" topology). Otherwise, we have $\triangle(z^i_\nu, z_\nu) = 0$. $dist(.,.)$ is a measure of the spatial distance between two image points and σ is a threshold. Note that the summations are defined over the active nodes. This loss function, which measures the distance/cost between two parse trees, is calculated by summing over individual subparts. This ensures that the computational complexity of the loss function is linear in the size of the LEAF and AND nodes of the hierarchy.

Max-margin can be reformulated as minimizing the constrained quadratic cost function of the weights:

$$\min_{\boldsymbol{\alpha}} \frac{1}{2}\|\boldsymbol{\alpha}\|^2 + C\sum_i \xi_i \tag{15}$$

$$\text{s.t.} \quad \boldsymbol{\alpha}\cdot\{\boldsymbol{\Phi}_{i,y_i} - \boldsymbol{\Phi}_{i,y}\} \geq L_{i,y} - \xi_i, \forall y; \tag{16}$$

where C is a fixed penalty parameter which balances the trade-off between margin size and outliers (its value is specified in the experimental section). Outliers are training samples which are only correctly classified after using a slack variable ξ_i to "move them" to the correct side of the margin. The constraints are imposed by introducing Lagrange parameters $\lambda_{i,y}$ (one λ for each constraint).

The solution to this minimization can be found by differentiation and expressed in form:

$$\boldsymbol{\alpha}^* = C\sum_{i,y} \lambda^*_{i,y}\left(\boldsymbol{\Phi}_{i,y_i} - \boldsymbol{\Phi}_{i,y}\right), \tag{17}$$

where the λ^* are obtained by maximizing the dual function:

$$\max_{\lambda} \sum_{i,y} \lambda_{i,y} L_{i,y} -$$

$$\frac{1}{2}C\sum_{i,j}\sum_{y,w}\lambda_{i,y}\lambda_{j,w}(\boldsymbol{\Phi}_{i,y_i} - \boldsymbol{\Phi}_{i,y})\cdot(\boldsymbol{\Phi}_{j,y_j} - \boldsymbol{\Phi}_{j,w}) \tag{18}$$

$$\text{s.t.} \quad \sum_y \lambda_{i,y} = 1, \forall i; \quad \lambda_{i,y} \geq 0, \forall i, y; \tag{19}$$

Observe that the solution will only depend on the training samples $(\mathbf{I}_i, y_i)$ for which $\lambda_{i,y} \neq 0$. These are the so-called *support vectors*. They correspond to training samples that either lie directly on the margin or are outliers (that need to use slack variables). The concept of support vectors is important for the optimization algorithm that we will use to estimate the λ^* (see next subsection).

It follows from equations (17,18), that the solution only depends on the data by means of the inner product $\Psi\cdot\Psi'$ of the potentials. This enables us to use the kernel trick [14] which replaces the inner product by a kernel $K(,)$ (interpreted as using features in higher dimensional spaces). In this paper, the kernels $K(,)$ take two forms, the linear kernel, $K(\boldsymbol{\Phi},\boldsymbol{\Phi}') = \boldsymbol{\Phi}\cdot\boldsymbol{\Phi}'$ for the data potentials $\boldsymbol{\Psi}^D$ and the radial basis function (RBF) kernel, $K(\boldsymbol{\Phi},\boldsymbol{\Phi}') = \exp\left(-r\|\boldsymbol{\Phi}-\boldsymbol{\Phi}'\|^2\right)$ for the remaining potentials $\boldsymbol{\Phi}$ – those defined on the AND and OR cliques – where r is the scale parameter of the RBF (specified in the experimental section).

5.2 Optimization of the Dual

The main difficulty with optimizing the dual, see equation (18), is the enormous number of constraints (i.e. the large number of $\{\lambda_{i,y}\}$ to solve for). We risk having to enumerate all the parse trees $y \in \mathcal{Y}$ which is almost impractical for an HCDT. Fortunately, in practice only a small number of support vectors will be needed. More precisely, only a small number of the $\{\lambda_{i,y}\}$ will be non-zero. This motivates the working set algorithm [1,41] to optimize the objective function in equation (18). The algorithm aims at finding a small set of *active constraints* that ensure a sufficiently accurate solution. More precisely, it sequentially creates a nested working set of successively tighter relaxations using a cutting plane method. It is shown [1,41] that the remaining constraints are guaranteed to be violated by no more than ϵ, without needing to explicitly add them to the optimization problem. The pseudocode of the algorithm is given in figure (7). Note that the inference algorithm is performed at the first step of each loop. Therefore, the efficiency of the training algorithm highly depends on the computational complexity of the inference algorithm (recall that we show in section (4) that the complexity of the inference algorithm is polynomial in the size of the AND/OR graph and the size of the input image). Thus, the efficiency of inference makes the learning practical. The second step is to create the working set sequentially and then estimate the parameter λ on the working set. The optimization over the working set is performed by Sequential Minimal Optimization (SMO) [30]. This involves incrementally satisfying the Karush-Kuhn-Tucker (KKT) conditions which are used to enforce the constraints. The pseudo-code of the SMO algorithm is depicted in figure (8). This procedure consists of two steps. The first step selects a pair of data points not satisfying the KKT conditions. The pseudo-code of pair selection is shown in figure (9). Two KKT conditions are defined by:

$$\lambda_{i,y} = 0 \Rightarrow H(\mathbf{I}_i, y) \leq H(\mathbf{I}_i, y^*) + \epsilon; \quad (KKT1)$$
$$\lambda_{i,y} > 0 \Rightarrow H(\mathbf{I}_i, y) \geq H(\mathbf{I}_i, y^*) - \epsilon; \quad (KKT2)$$

where $H(\mathbf{I}_i, y) = \boldsymbol{\alpha}\cdot\boldsymbol{\Phi}_{i,y} + L(y_i, y)$, $y^* = \arg\max_y H(\mathbf{I}_i, y)$ and ϵ is a tolerance parameter.

The second step is a local ascent step which attempts to update the parameters given the selected pair. The updating equations are defined as:

$$\lambda^{new}_{i,y'} = \lambda_{i,y'} + \delta$$
$$\lambda^{new}_{i,y''} = \lambda_{i,y''} - \delta \tag{20}$$

The dual optimization problem in equation (18) reduces to the simple problem of solving for δ:

$$\max_{\delta}[H(\mathbf{I}_i, y') - H(\mathbf{I}_i, y'')]\delta - \frac{1}{2}C\|\boldsymbol{\Phi}_{i,y'} - \boldsymbol{\Phi}_{i,y''}\|^2\delta^2 \tag{21}$$

$$\text{s.t.} \quad \lambda_{i,y'} + \delta \geq 0, \lambda_{i,y''} - \delta \geq 0. \tag{22}$$

Loop over i

1. $y^* = \arg\max_y H(\mathbf{I}_i, y)$ where $H(\mathbf{I}_i, y) = \langle \alpha, \Psi_{i,y} \rangle + L(y_i, y)$.
2. if $H(\mathbf{I}_i, y^*) - \max_{y \in S_i} H(\mathbf{I}_i, y) > \varepsilon$
   $S_i \leftarrow S_i \bigcup y^*$
   $\lambda_s \leftarrow$optimize dual over S, $S = S \bigcup S_i$

Fig. 7 Working Set Optimization

Given a training set S and parameter λ
Repeat

1. select a pair of data points (y', y'') not satisfying the KKT conditions.
2. solve optimization problem on (y', y'')

Until all pairs satisfy the KKT conditions.

Fig. 8 Sequential Minimal Optimization (SMO)

1. $Violation = False$
2. For each $\mathbf{I}^i, y', y'' \in S_i$
   (a) If $H(\mathbf{I}^i, y') > H(\mathbf{I}^i, y'') + \epsilon$ and $\lambda_{i,y'} = 0$ (KKT 1)
   $Violation = TRUE$; Goto step 3.
   (b) If $H(\mathbf{I}^i, y') < H(\mathbf{I}^i, y'') - \epsilon$ and $\lambda_{i,y'} > 0$ (KKT 2)
   $Violation = TRUE$; Goto step 3.
3. Return $y', y'', Violation$

Fig. 9 Pair Selection in SMO

This can be re-expressed as solving:

$$\max_{\delta} \{a\delta - \frac{b}{2}\delta^2\} \tag{23}$$

$$\text{s.t.} \quad c \leq \delta \leq d \tag{24}$$

where $a = H(\mathbf{I}_i, y') - H(\mathbf{I}_i, y'')$, $b = C\|\mathbf{\Phi}_{i,y'} - \mathbf{\Phi}_{i,y''}\|^2$, $c = -\lambda_{i,y'}$, $d = \lambda_{i,y''}$.

Hence, the analytical solution for two data points can be easily obtained by

$$\delta^* = \max(c, \min(d, a/b)). \tag{25}$$

This completes the description for how to get solutions for the update equations in (20). More details can be found in [30] and [39].

6 Experiments

In this section, we study the performance of HCDTs for segmentation, parsing, and weak-detection. We first study HCDT's performance on the horse dataset [3] and analyze the computational complexity of the inference. Next we apply HCDT's to determine the pose of baseball players.

6.1 Datasets and Implementation Details.

Datasets. We performed the experimental evaluations on two datasets: (i) the Weizmann Horse Dataset [3] and (ii) the Human Baseball dataset [27]. Some examples from these datasets are shown in figures (12) and (13) with parsing and segmentations results obtained by our method. Many results have been reported for these datasets which we use for comparison – see [33, 12, 23, 18, 45] for the horse dataset and [38, 26, 27] for the baseball dataset.

The Weizmann horse dataset is designed to evaluate segmentation, so the groundtruth only gives the regions of the object and the background. To supplement this groundtruth, we asked students to manually parse the images by locating the positions of active leaf nodes (about 24 to 36 nodes) of the HCDT in the images. These parse trees are used as groundtruth to evaluate the ability of the HCDT to parse the horses. There are 328 horse images in [3] of which 100 images are used for testing. The remaining 228 images and their parsing groundtruth are used for training. The HCDT model has 40 possible configurations to deal with the range of horse poses.

For the baseball dataset Srinivasan and Shi [38] only used 5 joint nodes (head-torso, torso-left thigh, torso-right thigh, left thigh-left lower leg, right thigh-right lower leg). For HCDTs there are 27 nodes on the boundary of each baseball player which gives more detailed parsing than alternatives [38, 26, 27]. These 27 points correspond to the 94 leaf nodes of the AND/OR graph in figure (2). Each (smallest) part has on average 3.48 (94/27) poses (e.g. orientation). We also asked students to label the parts of the baseball players (i.e. to identify different parts of the humans). For human body parsing, we used 48 human baseball images in Mori's dataset [27] as the testing set. The HCDT for humans is able to model 98 poses. In figure (13), observe that the dataset contains a large variety of human poses and the appearance of clothes changes a lot from image to image. We created a training dataset by collecting 156 human baseball images from the internet and got students to manually label the parse tree for each image for training.

Parameter Settings. The HCDT (after training by max-margin) was used to obtain the parse y (i.e. to locate the body parts). We used max-margin on the training dataset to learn the parameters of the max-margin model. During learning, we set $C = 0.1$ in equation (18), used the radial basis function kernel with $r = 0.1$, set the parameter in the loss function equation (14) to be $\sigma = 12$, and set $\epsilon = 0.01$ in figure (7). Our strategy for segmentation, which is inspired by Grab-Cut [35], is to obtain the parse by the inference algorithm on the HCDT to determine the positions of the leaf nodes on the boundary of the object. Then we performed postprocessing to give a continuous closed boundary by graph-cut (using the positions of the leaf nodes to yield an initial estimate of the feature statistics).

The Parsing Criterion. We used the *average position error* [38] as a criterion to measure the quality of the parsing. The position error is defined to be the distance (in pixels) between the positions of groundtruth and the parsing

result (leaf nodes). The smaller the position error, the better the quality of the parsing. For horses, there are between 24 and 36 leaf nodes which appear on the boundary of a horse. For the baseball player experiments, Srinivasan and Shi [38] only used 5 joint nodes (head-torso, torso-left thigh, torso-right thigh, left thigh-left lower leg, right thigh-right lower leg) per image. For HCDTs, there are 27 nodes along the boundary of human body which gives more detailed parsing.

The Segmentation Criterion Two evaluation criteria are used to measure the performances of segmentation. We use *segmentation accuracy* to quantify the proportion of the correct pixel labels (object or non-object). For the baseball player experiments, we use the segmentation measure, '*overlap score*', which is defined by $\frac{area(P \cap G)}{area(P \cup G)}$, where P is the area which the algorithm outputs as the segmentation and G is the area of ground-truth. The larger the overlap score, the better the segmentation.

The Criterion for Weak-Detection We also evaluated the performance of HCDTs for the weak-detection task (where the object is known to be in the image but its location is unknown). Our criterion judges detection to be successful if the area of the intersection of the detected object region and the true object region is greater than half the area of the union of these two regions.

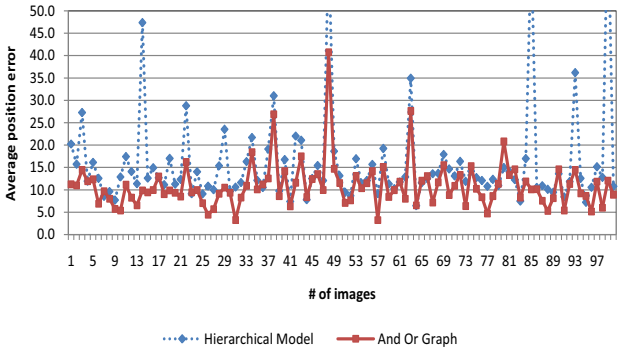

Fig. 10 We compare the parsing performance of hierarchical model and AND/OR graph on 100 test images.

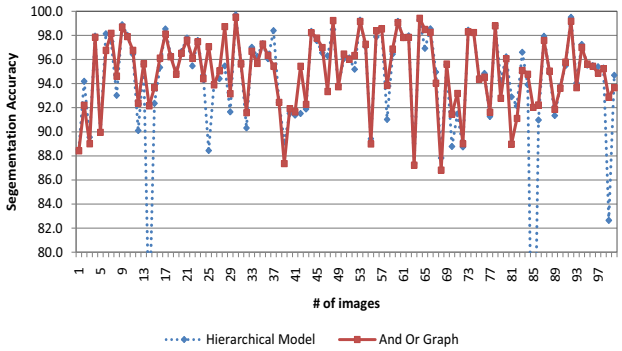

Fig. 11 We compare the segmentation performance of hierarchical model and AND/OR graph on 100 test images.

6.2 Performance of HCDT's on the Horse dataset

Results. In table (2) we compare the performances of an HCDT with 40 configurations/topologies against a simple hierarchical model with a fixed topology (i.e. we fix the states of the OR nodes). The topology of this fixed hierarchical model (the first one in the top node in figure (3) was chosen to be the topology that most frequently occurred (after doing inference with the HCDT with variable topology). The evaluations are performed on the same 100 test images. To quantify the significance of the improvement made by the HCDT (i.e. the AND/OR representation) we plot the 100 paired comparisons in figures (10,11). The p-value for the t-test on parsing is less than 10^{-5} while the p-value for segmentation is 0.03. These tests show that the improvement made by adding OR nodes is statistically significant for parsing (alignment of object parts), but less so for segmentation.

For all experiments we used a computer with 4 GB memory and 2.4 GHz CPU. Learning took 150 minutes for the model with fixed hierarchy and 180 minutes for the full HCDT. The inference time per image was 20 seconds for the fixed hierarchy model and 27 second for the full HCDT.

Our results show that the HCDT outperforms the fixed hierarchy model for all tasks (i.e. parsing, detection and segmentation) with only 30% more computational cost. In figure (12), we compare the parse and segmentation results obtained by the fixed hierarchical model and the HCDT. The states of the leaf nodes of parse tree give the estimated positions of the points along the boundary and are illustrated by colored dots (the same colors for corresponding subparts in different images). Observe that both models (i.e. fixed hierarchy and HCDT) are able to deal with large shape deformation and appearance variations (after training with max margin), see the top four examples, despite cluttered background and varied texture on the horses. But the HCDT is much better at locating subparts, such as the legs and the head, than the fixed hierarchy model which is only able to detect locate the torso reliably. This is illustrated by the last four examples in figure (12) where the legs and heads appear in different poses. The fixed hierarchy model succeeds at performing segmentation reasonably well even though its parsing results are not very good (mainly because of the effectiveness of grab cut). This observation is consistent with the quantitative comparisons in table (2). But the last example shows a case where incorrect parsing – the fixed hierarchy model locates the head in the wrong position – can result in poor segmentation. In summary, the HCDT performs well for both parsing and segmentation in these experiments.

Comparisons. In table (2), we compare the segmentation performance of our approach with other successful methods. Note that the object cut method [18] was reported on only 5 images. Levin and Weiss [23] make the assumption that the position of the object is given (other methods do

Fig. 12 This figure is best viewed in color. Columns (a) to (d) show the parsing and segmentation results obtained by the fixed hierarchy model and the HCDT respectively. The colored dots show the positions of the leaf nodes of the object.

Table 3 Empirical Complexity Analysis. This table shows the numbers of proposals and the time costs at different levels of the hierarchy ("clusters" means the number of proposals accepted after surround suppression). 'Aspects' is the average number of grandchildren nodes for each level, see text for details.

L	Nodes	Aspects	No. Clusters	Proposals	Time
8	1	12	11.1	2058.8	1.206s
6	8	1.5	30.6	268.9	1.338s
4	27	1	285.1	1541.5	1.631s
2	68	1	172.2	1180.7	0.351s

not make this assumption) and do not report how many images they tested their method on. Overall, Cour and Shi's method [12] was the most successful when evaluated on a large subset of the dataset. But their result is obtained by manually selecting the best from the top 10 results (other methods output a single result). By contrast, our approach outputs a single result but yields a higher pixel accuracy of 94.8%. Hence we conclude that our approach outperforms those alternatives which report results on large subsets of this dataset. Note that no other papers report parsing performance on this dataset since most (if not all) of these methods do not estimate the positions of different parts of the horse (and do not even represent them).

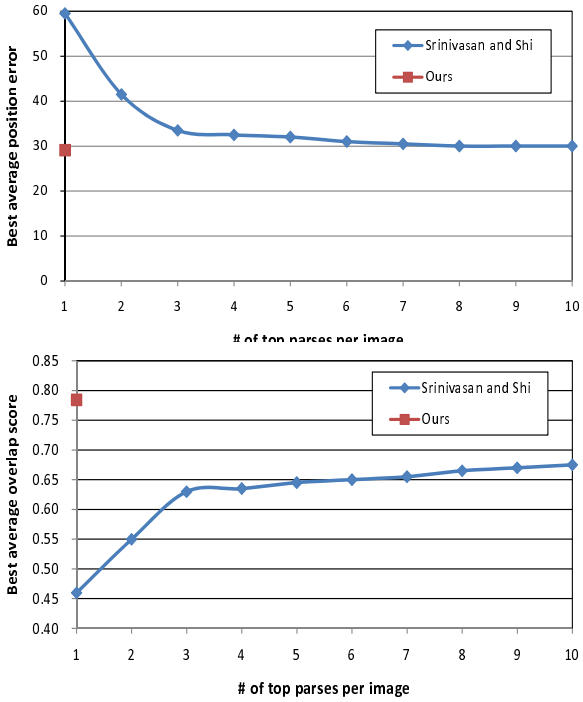

Fig. 14 We compare the results of HCDTs with those of Srinivasan and Shi [38]. The performance for parsing (position error) and segmentation (overlap score) are shown in the top and bottom figures respectively. Note that [38] select the best one (manually) of the top parses.

6.3 Empirical Complexity Analysis

Table (3) shows the empirical complexity properties of the algorithm. We report for AND levels only (recall that the model has a total of 8 levels) because the computation at the OR-nodes is almost instantaneous (it only requires listing the proposals from all its child nodes and doing pruning). Column 2 reports the number of nodes at each level. Column 3 gives the average number of aspects of the AND nodes at each level. Columns 4 and 5 reports the average number of *clusters* and proposals for each node – the number of proposals are the number of compositions of child nodes and the number of clusters is the number of accepted proposals after surround suppression. Column 6 reports the time. Observe that the number of proposals increases by an order of magnitude from level 6 to level 8. This is mostly due to the increase in the number of *aspects* (the more aspects, the more the number of proposals required) [1].Note that surround suppression greatly reduces the number of proposals (compare the numbers of clusters and proposals in table (3).

[1] We define aspects as follows. Let μ denote an "AND" node. Suppose node μ has k child "OR" node. Let $n_1, n_2, ..., n_k$ denote the number of child nodes of the k "OR" nodes. Then the aspect of node a is $n_1 \times n_2 \times ... \times n_k$. The number of proposals for node μ scales linearly with the number of aspects. Note that k and $n_1, ..., n_k$ will usually be small, so the aspects are usually not large.

6.4 Human Body Parsing

Parsing Results. We show our parsing and segmentation results for humans in figure (13). The dotted points give the positions of the leaf nodes of parse tree (which lie along the boundary of human). Similar subparts are indicated by the same color in all images. For example: (i) yellow and red points correspond to the left and right shoulder respectively, and (ii) light blue and dark blue points correspond to the left and right legs respectively. Observe that the variety of poses are extremely large, but the HCDT is able to successful estimate pose/configuration and to segment the body. The time cost of training the HCDT was 20 hours and the inference takes 2 minutes for images with size 640×480.

Performance Comparisons. We compare the performance of HCDTs to that reported by Srinivasan and Shi [38], which are the best results achieved so far on this dataset (e.g. better than Mori et al. 's [27]). Firstly, we compare the average position errors of the parse in figure (14). Observe that the best parse of the HCDT gives a performance slightly better than the best (manually selected) of the top 10 parses output by [38] and significantly better than the best (manually selected) of their top three parses. Secondly, we compare the average overlap scores in figure (14). The difference of performance measured by overlap score is arguably more significant. Observe that our result is significantly better than the best (manually selected) of their top 10 parses.

Table 2 Performance for parsing, segmentation and detection. The table compares the results for the fixed hierarchy model (without OR nodes), the HCDT, and alternative methods. Column 3 gives the parsing accuracy – the average position error of leaf nodes of the HCDT is 10 pixels. Column 4 quantifies the segmentation accuracy. Column 5 quantifies the detection rate. Column 6 lists the training time. The last column shows the average time of inference taken for one image.

Models	Testing Size	Parsing	Segementation Accuracy	Detection	Training Time	Testing Time
Hierarchical Model	100	15.6	94.5%	99%	150 m	20s
AND/OR Graph	100	10.8	94.8%	‘100%	180 m	27s
Ren et. al[33]	172	–	91%	–	–	–
Borenstein [4]	328	–	93.0%	–	–	–
LOCUS [45]	200	–	93.1%	–	–	–
Cour [12]	328	–	94.2%	–	–	–
Levin [23]	N/A	–	95.0%	–	–	–
OBJ CUT [18]	5	–	96.0%	–	–	–

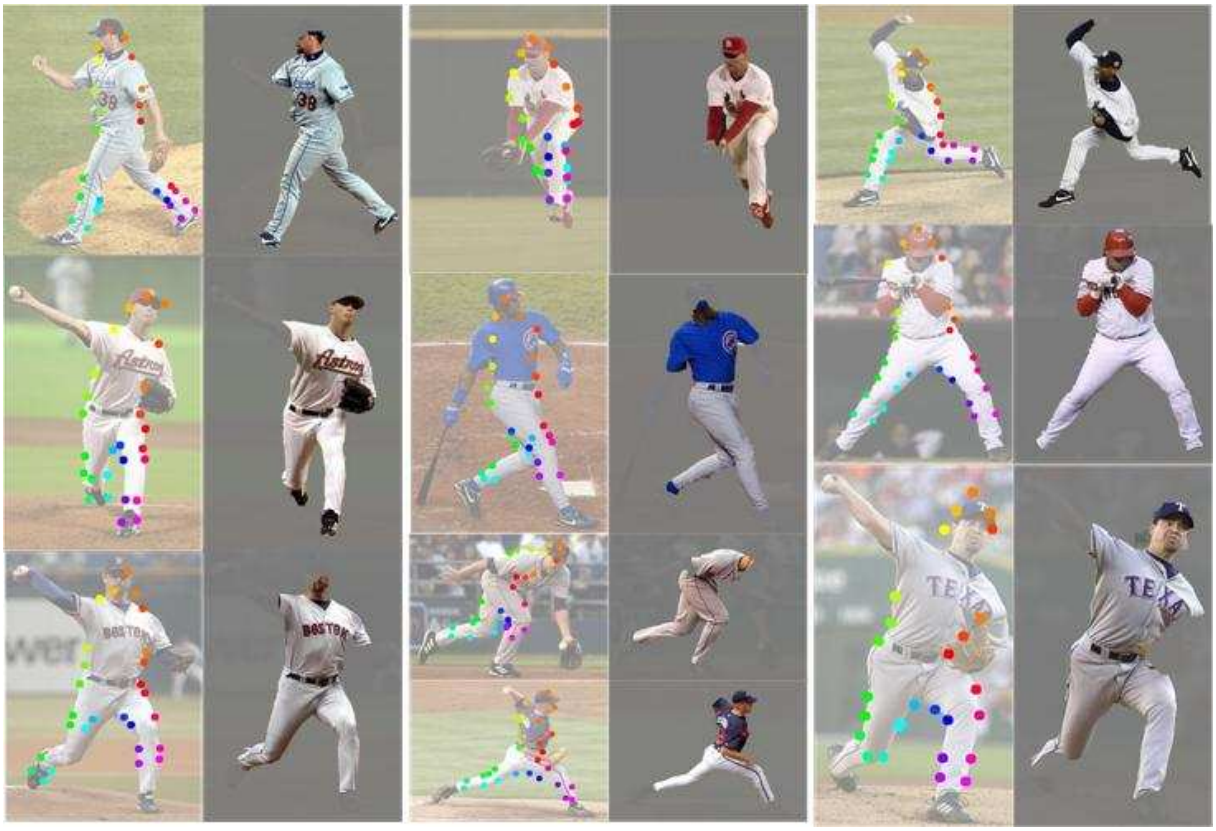

Fig. 13 The first column shows the parse results of the human body. Color points indicate the positions of subparts. The same color is used in all images to indicate the same subparts. The second column shows the segmentations. The remaining four columns give extra examples.

Convergence Analysis. We study the convergence behavior of max-margin AND/OR graph learning in figure (15). The left panel shows the convergence curve in terms of the objective function defined in equation (18). Observe that there is a big jump around iteration 200. The right panel plots the convergence curves of the average position error on the training and testing data. The trends of both plots are similar.

7 Discussion

We formulated a novel hierarchical deformable template (HCDT) for representing objects which has varying topology enabling it to adjust to a large variety of different configurations/poses. The HCDT specifies a probability distribution defined over an AND/OR graph. The node variables indicate the positions and properties of subparts of the object and obey the summarization principle. We report an efficient “compositional inference” which estimate the most probable states of the HCDT. This is a form of dynamic programming where pruning mechanisms are used to ensure that the algorithm

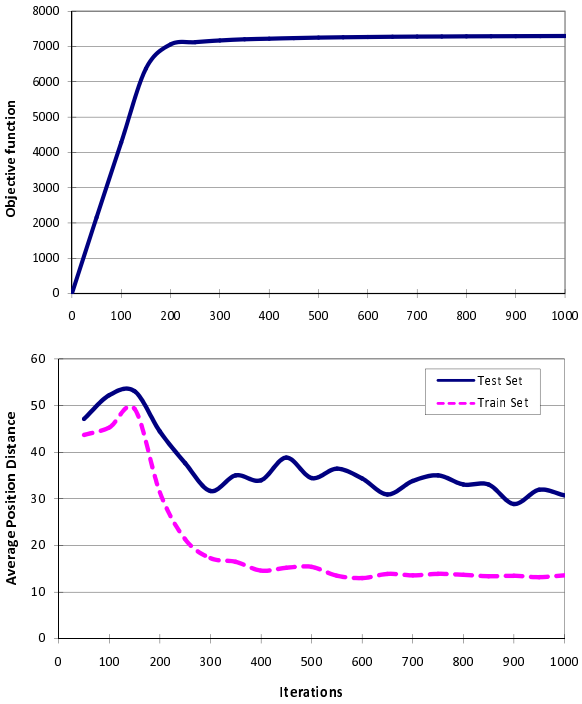

Fig. 15 Convergence Analysis. We study the behavior of max margin training. The first panel shows the convergence curve of the objective function defined in equation (18). The second panel shows the converge curves of the average position error evaluated on training and testing set.

is fast when evaluated on two public datasets. The structure of the AND/OR graph is specified by hand but the parameters are learnt in a globally optimal way by extending max-margin structure learning technique developed in machine learning. Advantages of our approach include (i) the ability to model the enormous number of poses/configurations that occur for articulated objects such as humans and horses, (ii) the discriminative power provided by max-margin learning (by contrast to MLE), and (iii) the use of the kernel trick to make use of high-dimensional features. We gave detailed experiments on the Weizmann horse and human baseball datasets, showing improvements over the state-of-the-art methods. We are currently working on improving the inference speed of our algorithm by using a cascade strategy. We are also extending the model to represent humans in more details.

Acknowledgements

We gratefully acknowledge support from the National Science Foundation with NSF grant number 0413214, IIS-0917141, and from the W.M. Keck Foundation.

References

1. Y. Altun, I. Tsochantaridis, and T. Hofmann, “Hidden markov support vector machines,” in *ICML*, 2003, pp. 3–10.
2. S. Belongie, J. Malik, and J. Puzicha, “Shape matching and object recognition using shape contexts,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 4, pp. 509–522, 2002.
3. E. Borenstein and S. Ullman, “Class-specific, top-down segmentation,” in *ECCV (2)*, 2002, pp. 109–124.
4. E. Borenstein and J. Malik, “Shape guided object segmentation,” in *CVPR (1)*, 2006, pp. 969–976.
5. X. Chen and A. Yuille, “A time-efficient cascade for real-time object detection:. with applications for the visually impaired,” in *CVPR*, 2005.
6. H. Chen, Z. Xu, Z. Liu, and S. C. Zhu, “Composite templates for cloth modeling and sketching,” in *CVPR (1)*, 2006, pp. 943–950.
7. Y. Chen, L. Zhu, C. Lin, A. L. Yuille, and H. Zhang, “Rapid inference on a novel and/or graph for object detection, segmentation and parsing,” in *NIPS*, 2007.
8. H. Chui and A. Rangarajan, “A new algorithm for non-rigid point matching,” in *CVPR*, 2000, pp. 2044–2051.
9. J.M. Coughlan, A. L. Yuille, C. English and D. Snow, “ Efficient Optimization of a Deformable Template Using Dynamic Programming,” in CVPR 1998.
10. J.M. Coughlan, A. L. Yuille, C. English and D. Snow,“ Efficient Deformable Template Detection and Localization without User Initialization,” Computer Vision and Image Understanding 78(3): 303-319. 2000.
11. J. M. Coughlan and S. J. Ferreira, “Finding deformable shapes using loopy belief propagation,” in *ECCV (3)*, 2002, pp. 453–468.
12. T. Cour and J. Shi, “Recognizing objects by piecing together the segmentation puzzle,” in *CVPR*, 2007.
13. K. Crammer and Y. Singer, “On the algorithmic implementation of multiclass kernel-based vector machines,” *Journal of Machine Learning Research*, vol. 2, pp. 265–292, 2001.
14. N. Cristianini and J. Shawe-Taylor, *An introduction to support Vector Machines: and other kernel-based learning methods*. New York, NY, USA: Cambridge University Press, 2000.
15. R. Dechter and R. Mateescu, “And/or search spaces for graphical models,” *Artif. Intell.*, vol. 171, no. 2-3, pp. 73–106, 2007.
16. P. Felzenszwalb and D. McAllester and D. Ramanan,“ A Discriminatively Trained, Multiscale, Deformable Part Model” in Proceedings of the IEEE CVPR 2008.
17. Y. Jin and S. Geman, “Context and hierarchy in a probabilistic image model,” in *CVPR (2)*, 2006, pp. 2145–2152.
18. M. P. Kumar, P. H. S. Torr, and A. Zisserman, “Obj cut,” in *CVPR (1)*, 2005, pp. 18–25.
19. J. D. Lafferty, A. McCallum, and F. C. N. Pereira, “Conditional random fields: Probabilistic models for segmenting and labeling sequence data,” in *ICML*, 2001, pp. 282–289.
20. S.L. Lauritzen, and D.J. Spiegelhalter. ”Local Computations with Probabilities on Graphical Structures and their Application to Expert Systems”. Journal of the Royal Statistical Society. Series B. 50 (2): 157224. 1988.
21. M. W. Lee and I. Cohen, “Proposal maps driven mcmc for estimating human body pose in static images,” in *CVPR (2)*, 2004, pp. 334–341.
22. B. Leibe, A. Leonardis, and B. Schiele, “Combined object categorization and segmentation with an implicit shape model,” in *ECCV’04 Workshop on Statistical Learning in Computer Vision*, Prague, Czech Republic, May 2004, pp. 17–32.
23. A. Levin and Y. Weiss, “Learning to combine bottom-up and top-down segmentation,” in *ECCV (4)*, 2006, pp. 581–594.
24. C. Manning and H. Schuetze, *Foundations of statistical natural language processing*. Cambridge, Mass, USA: MIT Press, 1999.
25. M. Meila and M. I. Jordan, “Learning with mixtures of trees,” *Journal of Machine Learning Research*, vol. 1, pp. 1–48, 2000.

26. G. Mori, X. Ren, A. A. Efros, and J. Malik, "Recovering human body configurations: Combining segmentation and recognition," in *CVPR (2)*, 2004, pp. 326–333.
27. G. Mori, "Guiding model search using segmentation," in *ICCV*, 2005, pp.1417–1423.
28. M. Oren, C. Papageorgiou, P. Sinha , E. Osuna and T. Poggio, "Pedestrian detection using wavelet templates," In Proc. Computer Vision and Pattern Recognition, pages 193–199, Puerto Rico, June 16–20 1997.
29. E. Osuna, R. Freund, and F. Girosi, "Training support vector machines: an application to face detection," in *CVPR*, 1997, pp. 130–136.
30. J. C. Platt, "Using analytic qp and sparseness to speed training of support vector machines," in *NIPS*, 1998, pp. 557–563.
31. D. Ramanan, "Learning to parse images of articulated bodies," in *NIPS*, 2006, pp. 1129–1136.
32. X. Ren, A. C. Berg, and J. Malik, "Recovering human body configurations using pairwise constraints between parts," in *ICCV*, 2005, pp. 824–831.
33. X. Ren, C. Fowlkes, and J. Malik, "Cue integration for figure/ground labeling," in *NIPS*, 2005.
34. R. Ronfard, C. Schmid, and B. Triggs, "Learning to parse pictures of people," in *ECCV (4)*, 2002, pp. 700–714.
35. C. Rother, V. Kolmogorov, and A. Blake, ""grabcut": interactive foreground extraction using iterated graph cuts," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 309–314, 2004.
36. L. Sigal and M. J. Black, "Measure locally, reason globally: Occlusion-sensitive articulated pose estimation," in *CVPR (2)*, 2006, pp. 2041–2048.
37. P. Srinivasan and J. Shi, "Bottom-up recognition and parsing of the human body," in *EMMCVPR*, 2007, pp. 153–168.
38. P. Srinivasan and J. Shi, "Bottom-up recognition and parsing of the human body," in *CVPR*, 2007.
39. B. Taskar, C. Guestrin, and D. Koller, "Max-margin markov networks," in *NIPS*, 2003.
40. B. Taskar, D. Klein, M. Collins, D. Koller, and C. Manning, "Max-margin parsing," in *EMNLP*, 2004.
41. I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun, "Support vector machine learning for interdependent and structured output spaces," in *ICML*, 2004.
42. V. N. Vapnik, *The nature of statistical learning theory*. New York, NY, USA: Springer-Verlag New York, Inc., 1995.
43. P. A. Viola and M. J. Jones, "Robust real-time face detection," *International Journal of Computer Vision*, vol. 57, no. 2, pp. 137–154, 2004.
44. P. Viola, J. C. Platt, C. Zhang, "Multiple Instance Boosting for Object Detection,"in NIPS 2005.
45. J. M. Winn and N. Jojic, "Locus: Learning object classes with unsupervised segmentation," in *ICCV*, 2005, pp. 756–763.
46. J. Zhang, J. Luo, R. T. Collins, and Y. Liu, "Body localization in still images using hierarchical models and hybrid search," in *CVPR (2)*, 2006, pp. 1536–1543.
47. L. Zhu, Y. Chen, and A. L. Yuille, "Unsupervised learning of a probabilistic grammar for object detection and parsing," in *NIPS*, 2006, pp. 1617–1624.
48. L. Zhu and A. L. Yuille, "A Hierarchical Compositional System for Rapid Object Detection," in *NIPS*, 2005.
49. L. Zhu, Y. Chen and A. Yuille, " Unsupervised Learning of Probabilistic Grammar-Markov Models for Object Categories," IEEE Trans. Pattern Anal. Mach. Intell. 2009.
50. L. Zhu, C. Lin, H. Huang, Y. Chen and A. Yuille, " Unsupervised Structure Learning: Hierarchical Recursive Composition, Suspicious Coincidence and Competitive Exclusion," in ECCV 2008.
51. L. Zhu, Y. Chen, Y. Lu, C. Lin and A. Yuille, "Max Margin AND/OR Graph Learning for Parsing the Human Body," in CVPR 2008.
52. S. Zhu and D. Mumford, "A stochastic grammar of images," vol. 2, no. 4, pp. 259–362, 2006.