UCLA
Department of Statistics Papers

Title

Unsupervised Learning of Probabilistic Object Models (POMs) for Object Classification, Segmentation and Recognition using Knowledge Propagation

Permalink

https://escholarship.org/uc/item/1979d96q

Authors

Chen, Yuanhao
Zhu, Long Leo
Yuille, Alan
et al.

Publication Date

2009-08-24

Peer reviewed

eScholarship.org Powered by the California Digital Library
University of California

Unsupervised Structure Learning: Hierarchical Recursive Composition, Suspicious Coincidence and Competitive Exclusion

Long (Leo) Zhu[1], Chenxi Lin[2], Haoda Huang[2], Yuanhao Chen[3], and Alan Yuille[1,4]

[1] Department of Statistics, University of California, Los Angeles
{lzhu,yuille}@stat.ucla.edu
[2] Microsoft Research Asia
{chenxi.lin,hahuang}@microsoft.com
[3] University of Science and Technology of China
yhchen4@ustc.edu
[4] Department of Psychology and Computer Science, UCLA

Abstract. We describe a new method for unsupervised structure learning of a *hierarchical compositional* model (HCM) for deformable objects. The learning is unsupervised in the sense that we are given a training dataset of images containing the object in cluttered backgrounds but we do not know the position or boundary of the object. The structure learning is performed by a bottom-up and top-down process. The bottom-up process is a novel form of hierarchical clustering which recursively composes proposals for simple structures to generate proposals for more complex structures. We combine standard clustering with the *suspicious coincidence principle* and the *competitive exclusion principle* to prune the number of proposals to a practical number and avoid an exponential explosion of possible structures. The hierarchical clustering stops automatically, when it fails to generate new proposals, and outputs a proposal for the object model. The top-down process validates the proposals and fills in missing elements. We tested our approach by using it to learn a hierarchical compositional model for parsing and segmenting horses on Weizmann dataset. We show that the resulting model is comparable with (or better than) alternative methods. The versatility of our approach is demonstrated by learning models for other objects (e.g., faces, pianos, butterflies, monitors, etc.). It is worth noting that the low-levels of the object hierarchies automatically learn generic image features while the higher levels learn object specific features.

1 Introduction

The goal of this paper is to learn a hierarchical compositional model (HCM) for deformable objects. The learning is unsupervised in the sense that we are given a training dataset of images containing the object in cluttered backgrounds but we do not know the position or boundary of the object. Unsupervised learning is

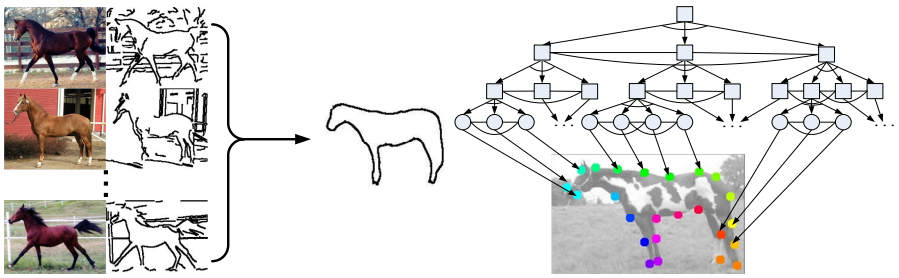

Fig. 1. Left:The learning is unsupervised in the sense that we are given a training dataset of images containing the object in cluttered backgrounds but we do not know the position or boundary of the object. Right: The hierarchical representation of the object. The boxes represent non-leaf nodes. The circles denote leaf nodes that directly relate to properties of the input image. The topology of the structure is not given, but learnt in an unsupervised way.

desirable since it avoids the need for time consuming hand labeling and prevents implicit biases. We apply the model to tasks such as object segmentation and parsing (matching of object parts).

Learning a hierarchical compositional model is very challenging since it requires us to learn the structure of the model (e.g. the relationships between the variables, and the existence of hidden variables) in addition to the parameters of the model. The difficulties are visually illustrated in figure 1: (i) the objects are deformable so there is ambiguity in the structure. (ii) there is cluttered background noise. (iii) parts of the object may be missing, (iv) the input is simple oriented edge features, which is a simple and highly ambiguous representation.

We now discuss some critical computational issues for unsupervised learning which motivate our hierarchical approach and contrast it to other unsupervised approaches for learning object models (e.g. [1], [2]). An abstraction of the structure learning problem is that we have a dataset of images each of which contain approximately M object features and N total features. In this paper, we are considering the case of $M = 100$ and $N = 5000$. Learning an object model requires solving a complicated correspondence problem to determine which features are object, which are background, and the spatial relationships between the object features. There are several strategies for addressing this correspondence problem. The first naive strategy is brute force enumeration which involves testing all N^M possibilities. This is only practical if M and N are small and the appearances of the features are sufficiently distinctive to enable many possible correspondences to be rejected. The second strategy is to learn the model in a greedy (non-hierarchical) way by sequentially growing subparts one by one [1, 2]. This is practical in cases where brute force fails, but it still makes two assumptions: 1) sparseness assumption which requires M and N to be fairly small, e.g., $M = 6$ and $N = 100$ in [1] and 2) small ambiguity assumption which requires the appearances of the features to be somewhat distinctive. Neither of these two strategies are applicable if the features are edgelets, because M and N are both large and all edgelets have the same appearance which leads to big ambiguity. (One strategy is to use more powerful features, but we argue that

this merely postpones the complexity problem). The purpose of this paper is to develop a more general unsupervised learning method without making strong assumptions of sparseness and low ambiguity. In other words, we try to provide a unified learning scheme for both generic features and object structures. Hence, we are driven to a third strategy, named as Recursive Composition, that creates a model by combining elementary structures to build a hierarchy.

Our strategy is based on a novel form of bottom-up hierarchical clustering. We assume that the object can be expressed in terms of recursive compositions of elementary structures (see the right panel of figure 1). We learn the structure by repeatedly combining proposals for substructures to make proposals for more complex structures. This process stops when we cannot generate any more proposals. A huge number of proposals are examined and stored at every level to avoid ambiguity since small substructure are not necessarily distinct enough between object and background. However, combining proposals in this way risks a combinatorial explosion (imagine that the number of combinations will grow exponentially as we go up to the upper levels). We avoid this by the use of two principles: (i) suspicious coincidences, and (ii) competitive exclusion. Suspicious coincidences eliminates proposals which occur infrequently in the image dataset (in less than 90 percent of the images). The competitive exclusion principle is adapted from ecology community where it prevents two animals from sharing the same environmental niche. In this paper, competitive exclusion eliminates proposals which seek to explain overlapping parts of the image.

The bottom-up clustering is followed by a top-down stage which refines and fills in gaps in the hierarchy. These gaps can occur for two reasons. Firstly, the competitive exclusion principle sometimes eliminates subparts of the hierarchy because of small overlaps. Secondly, gaps may occur at low-levels of the hierarchy, for example at the neck of a horse, because there are considerable local shape variations and so suspicious coincidences are not found. The top-down process can remove these gaps automatically by relaxing the two principles. For example, at higher levels the system discovers more regular relationships between the head and torso of the horse which provides context enabling the algorithm to fill in the gaps.

In summary, hierarchical bottom-up and top-down procedure allows us to grow the structure exponentially (the height of a hierarchy is $\log(M)$) and end up with a nearly complete structure (the resulting representation is dense and the size M could be big). We tested our approach by using it to learn a hierarchical compositional model for parsing and segmenting horses on Weizmann dataset [3]. We show that the resulting model is comparable with (or better than) alternative methods. The versatility of our approach is demonstrated by learning models for other objects (e.g., faces, pianos, butterflies, monitors, etc.).

2 Background: Hierarchical Structure Learning

Hierarchical design dates back to Fukushima's Neocognitron [4]. Recently, there have been many new developments including new representations and learning

algorithms. For example, Geman et al. [5] propose a hierarchical object model designed by a compositional principle using an AND/OR graph. Inference algorithms have been invented for this type of model [6], which we will adapt and use to perform inference in this paper (Note [6] does not address any learning algorithm). Deep networks [7], another type of multi-layer system, has recently been proposed by Hinton et al. [7]. Ullman et al. [8] learns a hierarchical feature representation in a supervised manner. Poggio and his colleagues'[9] build a hierarchical structure for rapid object recognition motivated by mimicing the architecture of the visual cortex of the human brain. Ahuja and Todorovic's hierarchical representation [10] is based on segmented image regions. Fleuret and Geman [11] provide a coarse-to-fine strategy which starts from edgelets. Fidler and Leonardis[12]'s unsupervised learning approach is most related to our work (the main difference is that they treat rigid objects, have a less dense representation, and do not have a top-down stage). In summary, in several cases these models[5, 6, 9] are not learnt but are specified by the user. Some unsupervised methods [8] assume that background is clean and the object is roughly aligned. Unsupervised learning (in the sense of this paper) has been performed [1, 2] (without a hierarchy) but for object models with limited number of object features (as discussed earlier). Most models [1, 9, 8, 5, 2] focus on recognition or categorization, but not parsing.

3 The Hierarchical Compositional Model (HCM) and the Inference algorithm

3.1 The Hierarchical Compositional Model (HCM)

The graph structure for the hierarchical model is depicted in the right panel of figure 1 and defined as follows. We let V_r be the set of nodes of the graph with root node r. Each node $\nu \in V_r$ has a set of child nodes T_ν (a node is constrained to have a single parent), V_r^{LEAF} are the leaf nodes (the only ones which interact with the image). For any node $\nu \in V_r$, we can define a graph V_ν with root node ν containing the descendants of ν. The edges for the graph are of three types: (i) vertical *data edges* which relate the leaf nodes to the image, (ii) horizontal edges relating the children of each node to each other (described below), and (iii) vertical edges relating parents to children (specified by $\{T_\nu\}$). In this paper, all vertical edges are directed. The horizontal edges only connect the child nodes with the same parent. Moreover, we restrict the number of child nodes to be less than six, i.e., $|T_\nu| \leq 6$. This ensures that the size of the biggest clique of the hierarchy is small. We use the notation Ω_r to represent the graph (i.e. the set of nodes V_r and the set of edges).

A configuration of the graph is an assignment of state variables $z = \{z_\nu\}$ to all $\nu \in V_r$. In this paper, we set $z_\nu = (x_\nu, y_\nu, \theta_\nu, s_\nu)$, where (x, y), θ and s denote image position, orientation, and scale respectively (the scale is the area of the image that the node represents). We use the notation $Z_\nu = \{z_\mu : \mu \in V_\nu\}$ to denote the state of node ν and all its descendent nodes.

We define a Gibbs distribution $P(z, d; \omega, \Omega)$ for the probability of the graph as follows:

$$P(z, d; \omega, \Omega) = \frac{1}{Z(\omega, \Omega)} \exp\{-E(z, d; \omega, \Omega)\}. \tag{1}$$

where $Z(\omega, \Omega)$ is the partition function, d is the image, ω denotes the parameters of the distribution and Ω denotes the graph structure (i.e. the nodes and the edges). The energy function $E(z, d; \omega, \Omega)$ is the sum of three terms corresponding to the three types of edges in the graph:

$$E_d(d, Z_r; \omega, \Omega) + E_h(Z_r; \omega, \Omega) + E_v(Z_r; \omega, \Omega), \tag{2}$$

where E_d, E_h, E_v are energy terms defined at the data, horizontal, and vertical edges. We now describe the specific choices used in this paper.

The *data term* E_d is given by:

$$E_d(d, Z_r; \omega, \Omega) = \sum_{\nu \in V_r^{LEAF}} f(d_\nu, z_\nu), \tag{3}$$

where V_r^{LEAF} is the set of the leaf nodes and $f(.,.)$ is the (negative) logarithm of a Gaussian defined over the grey-scale intensity gradient (i.e. $d_\nu = \boldsymbol{\nabla} I_\nu$). It biases the leaf nodes to be located at image locations where the image gradient is large, and for their orientations to be perpendicular to the image gradient. This term is fixed and not learnt.

The *horizontal term* E_h is given by:

$$E_h(Z_r; \omega, \Omega) = \sum_{\nu \in V_r / V_r^{LEAF}} \sum_{(\mu,\rho,\tau):\mu,\rho,\tau \in T_\nu} g(z_\mu, z_\rho, z_\tau; \omega), \tag{4}$$

where $g(z_\mu, z_\rho, z_\tau; \omega)$ is the (negative) logarithm of Gaussian distribution defined on the *invariant triplet vector* (ITV) $l(z_\mu, z_\rho, z_\tau)$ constructed from (z_μ, z_ρ, z_τ) [2]. (The ITV is invariant to the translation, rotation, and scaling of the triple, which ensures that the full probability distribution is also invariant to these transformations). The summation is over all triples formed by the child nodes of each parent, see the right panel of figure (1). The parameters of the Gaussian are indicated by $\omega = (\mu, \sigma)$.

The *vertical term* E_v is used to hold the structure together by relating the state of the parent nodes to the state of their children. It is defined by:

$$E_v(Z_r; \omega, \Omega) = \sum_{\nu \in V_r / V_r^{LEAF}} h(z_\nu; \{z_\mu \text{ s.t.} \mu \in T_\nu\}), \tag{5}$$

where $h(.,.) = 0$ provided the average orientations and positions of the child nodes are equal to the orientation and position of the parent node – formally $(x_\nu, y_\nu, \theta_\nu) = (1/|T_\nu|) \sum_{\mu \in T_\nu} (x_\mu, y_\mu, \theta_\mu)$. If this equality does not hold, then $h(.,.) = \kappa$, where κ is a large positive number. The scale variable $s_\nu = \sum_{\mu \in T_\nu} s_\mu$, hence the parent node represents the sum of the regions in the image represented by its children.

Observe that the nodes of the graph have the same variables at all levels (i.e. (x, y, θ, s)). This makes use of hierarchical independence assumption–the higher level interactions depend only on the summarization of all child nodes at the lower levels. More precisely, the child nodes only propagate a limited summary (center position, total size and orientation) of their state information to their parents. The choice of the vertical term E_v means that this information is simply the average of their node variables.

We stress that we can use equations (2,3,4,5) to calculate the energy for any subgraph by specifying the root node. This gives an effective way for computing the full energy, by combining the energy of the subgraphs, and is exploited during inference and learning. We can also exploit this hierarchy in order to compute other important properties, such as the partition function $Z(\omega, \Omega)$. Note, in figure 1, the vertical edges are directed and horizontal edges only connect the child nodes (less than six) with the same parent. Therefore, the partition function can be factorized into several components (corresponding to the cliques) whose sizes are small.

3.2 The Inference Algorithm for parsing when HCM is known

We now specify an efficient inference algorithm for the HCM. This algorithm will be used both when we are learning the model, see section (4), and when we are applying the model to new images to perform detection, segmentation, and parsing. We adapt a compositional algorithm that was designed for inference on AND/OR graphs [6] (Note their work does not present any learning algorithm which is the major topic of this paper). The algorithm has a bottom-up and a top-down stage. We only take the bottom-up stage which makes proposals for the states of the node variables Z_r. For brevity we omit the details. See [6] for more details.

4 Unsupervised Structure Learning

We now address the critical issue of how to learn the structure of the hierarchical model from a set of images which contain an example of the object with varying background. Formally, this requires us to estimate the structure parameters Ω and the distribution parameters ω. The task of the unsupervised learning is defined as :

$$(\omega^*, \Omega^*) = \arg\max P(\omega, \Omega|d) \tag{6}$$

$$= \arg\max P(d|\omega, \Omega)P(\omega, \Omega) \tag{7}$$

$$= \arg\max \sum_z P(d|z, \omega)P(z|\omega, \Omega)P(\omega, \Omega) \tag{8}$$

where $P(d|z, \omega)P(z|\omega, \Omega) = P(d, z|\omega, \Omega)$ is defined in equation (1). $P(\omega, \Omega)$ accounts for the prior distribution of the structure which plays a similar role of

structure regularization. $P(\omega, \Omega)$ is factorized into $P(\omega)$ and $P(\Omega)$. The parameters indicated by ω are μ and σ in the gaussian functions. The prior on σ is of the form $\mathcal{N}(0, \beta I)$. The prior on μ^l at certain level l puts hard constraints on the range of the distance allowed between any two substructures. The prior of Ω is uniform distribution. The summation $\sum_z P(d|z, \omega)P(z|\omega, \Omega)P(\omega, \Omega)$ is used as a *score* function to measure the goodness of the fit of the structure. Intuitively, the score function tells us how frequently a structure encoded by (Ω, ω) appears in the training set. Our approach is based on a bottom-up process to propose a set of structures (Ω, ω) followed by a top-down process which refines the result by adjusting (Ω, ω).

Our algorithm makes several approximations to simplify the learning problem. Firstly, we will exploit the hierarchical nature of the model to estimate the parameters ω locally. In principle, we could use these local estimates to initialize an EM algorithm to determine the parameters globally. Secondly, while our algorithm proposes a structure Ω we cannot guarantee that it is the globally optimal structure. But our experimental results, see section (5), provide empirical evidence that these approximations are reasonable.

4.1 The Bottom-Up Process

The bottom-up process constructs hierarchical object models by composing them from more elementary components. We use two principles to prevent a combinatorial explosion of compositions and to ensure that our compositions result in desirable object models. The two principles are: (i) *suspicious coincidences* where we keep compositions which occur frequently in the images and reject compositions which do not, and (ii) *competitive exclusion* where we remove compositions which match to regions of the image which overlap with other compositions. The relative importance of these different principles is shown empirically in the results section, see table 2 and figure 6. For example, observe how competitive exclusion play a small role when the compositions are small but is of major importance as the compositions get large.

Our strategy proceeds by creating vocabularies of concepts at different levels in the hierarchy, where each concept is a hierarchical compositional model. See figure 2 for visual (symbolic) illustration. The concepts are generated by composing concepts at lower levels. The basic ideas are to detect all instances of the concepts at level l in the images (using the inference algorithm for each concept). We *form compositions* of these concepts by identifying sets of these concepts that appear in sub-regions of the images. We *cluster* these compositions based on the spatial relationships between their elements (the instances of the concepts at level $l-1$) to get a set of concepts at level l whose distribution parameters ω are estimated from the clusters. We run the inference algorithm on the images to find the instances of all the concepts. Then we use the *suspicious coincidence principle* to remove concepts which occur infrequently (i.e. have few instances). Next we use the *competitive exclusion principle* to remove concepts whose instances overlap with those of other concepts (with better scores). The remaining concepts form the vocabulary of concepts at level $l+1$. The process

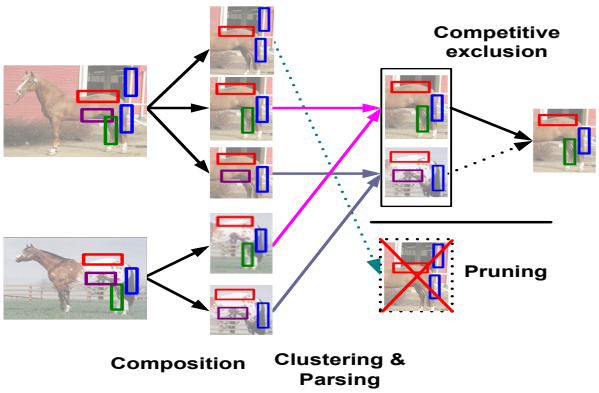

Fig. 2. This figure illustrates the procedure of the unsupervised learning. The first column shows the responses of the features from the "vocabulary" at level l. the second column shows the compositions at step 1). the third column shows the clustering at step 2. The noise(non-regular) pattern bounded by dotted line is pruned out by step 4. The last column plots the results after step 5. At step 5, the compositions, which are constructed by different components, but parse the same areas in the image, are grouped together (the maximum is kept

repeats until no new concepts are composed. For our applications (e.g. images containing a large object with variable background), it will terminate automatically in a small number of proposals when the hierarchy has reached a maximum size (i.e. there is no larger structure to be found).

The full procedure is described more formally in the next few paragraphs and is summarized by the pseudo-code in figure (3) and illustrated visually in figure 2. First we introduce some notation. A *concept* at level l is notated by $P^l_{\omega,\nu}$. It is a HCM with root node ν and ω represents the parameters of this model. A *vocabulary of concepts* at level l is the set $\{P^l_{\omega,\nu}\}$. We can parse the training dataset using the vocabulary of concepts at level l to get a set of *instances* of each concept $\{MP^l_{\nu,a}\}$, where $a = 1, ..., N^l_\nu$ indexes the set of instances. Each instance is represented by its state variables Z_ν.

We define the bottom-up process as follows. At each level $l-1$ we have a vocabulary of concepts $\{P^{l-1}_{\omega,\nu}\}$ and a set of instances of the concepts in the images $\{MP^{l-1}_{\nu,a}\}$. We then repeat the following steps.

1. Composition. We search through the images to find instances of triplets of concepts $MP^{l-1}_{\mu,\omega}, MP^{l-1}_{\rho,\omega}, MP^{l-1}_{\tau,\omega}$ within subregions of size S_{l-1} (for all μ, ρ, τ in the vocabulary). From each triplet instance we construct an instance P^l_ν where node ν has children $\mu, \rho, \tau \in T_\nu$.

2. Clustering. We cluster the instances P^l_ν by the second order moments of the shape descriptor (triplet vector) of the positions, scales and orientation $(x_\rho, y_\rho, \theta_\rho, s_\rho)$ of the children of ν (this clustering is done separately for all triples in the concept vocabulary at level $l-1$). The clustering outputs a set of *concepts* at level l notated by $\{P^l_{\omega,\nu}\}$. Some of the parameters $\omega = (\mu, \sigma)$ of each concept are estimated from the second order moments (see above) and the remainder are inherited from the underlying concepts at level $l-1$.

Input: $\{MP^1\}$. Output: $\{MP_\omega{}^L\}$. $\oplus$ denotes the operation of combining two proposals.

- Bottom-Up-Structure-Construction(MP^1)
  Loop : $l = 1$ to L
  1. Composition: $\{P_\nu^l\} = \oplus(MP_{\mu,a}^{l-1}, MP_{\rho,b}^{l-1}, MP_{\tau,c}^{l-1})$ and $T_\nu^l = \{\mu^{l-1}, \rho^{l-1}, \tau^{l-1}\}$
  2. Clustering: $\{P_{\omega\nu}{}^l\} = Clustering(\{P_\nu^l\})$
  3. Parsing: for each $P_{\omega\nu}{}^l$, $\{MP_{\nu,a}^l\} = Parser(P_{\omega\nu}{}^l, \{MP_\mu^{l-1}, MP_\rho^{l-1}, MP_\tau^{l-1}\})$
  4. Suspicious Coincidence (Pruning) : $\{P_{\omega\nu}{}^l\}\{P\omega_\nu^l | Score(P\omega_\nu^l) > T_{pruning}\}$
  5. Competitive Exclusion: $\{(MP_{\omega\nu}{}^l, CL_{\omega\nu}{}^l)\} = CompetitiveExclusion(\{P_{\omega\nu}{}^l\}, \epsilon_{region})$ where ϵ_{region} is the size of the window W_{image} defined in regions which are covered by a set of images.
- $\tilde{P_\omega} = \arg\max_\nu Score(MP_\omega{}^L_\nu)$
- Top-Down-Structure-Extension
  Loop: $l = L$ to 2, for each node ν at level l within $\tilde{P_\omega}$
  - repeat
    1. $P_{\omega\rho}^* = \arg\max_{P_{\omega\rho}{}^{l-1}} Score(\tilde{P_\omega} \oplus P_\omega{}^{l-1}_\rho)$
    2. $\Delta = Score(\tilde{P_\omega} \oplus P_{\omega\rho}^*) - Score(\tilde{P_\omega})$
    3. $\tilde{P_\omega} = \tilde{P_\omega} \oplus P_{\omega\rho}^*$; $T_\nu^l = T_\nu^l \cup \rho^{l-1}$

    until $\Delta < T_{extension}$

Fig. 3. Bottom-up and Top-down Learning.

3. Parsing. We parse the image dataset using the inference algorithm (see section (3.2)) for the set of concepts output by the clustering $\{P_{\omega,\nu}^l\}$. This gives a set of instances of the concepts in the dataset $\{MP_{\nu,a}^l\}$.

4. Pruning by Suspicious Coincidences. We remove concepts from $\{P_{\omega,\nu}^l\}$ if they do not have sufficient number of instances in the dataset (i.e. if there are instances of the concept in less than ninety percent of the images).

5. Competitive exclusion. We remove concepts from $\{P_{\omega,\nu}^l\}$ if their instances have significant overlap with instances of other concepts (and the other concepts have better scores).

The remaining concepts form the concept vocabulary $\{P_\omega^l\}$ at level l. Their instances in the dataset $\{MP^l\}$ have already been calculated (in the parsing step above). The process repeats itself until no new concepts are generated. In practice, this happens within 4-5 levels (see experimental section). The process is initialized for the leaf nodes. The vocabulary of concepts at level 1 $\{P_{\omega,\nu}^1 : \nu = 1, ..., 4\}$ consists of four types of edgelets characterized by their orientations $\theta = m\pi/4$ for $m = 0, 1, 2, 3$. The size s is set to a default value s_1. We parse the training set of images to detect the instances of all the concepts (more precisely, we compute the set of points $(x, y) : E_d(\nabla I(x, y), (x, y, m\pi/4, s_1) < T_1$, where T_1 is a threshold, for $m = 0, 1, 2, 3)$. This gives the set of instances of each concept $\{P_{\omega,\nu}^1\}$.

4.2 The Top-Down Process

The top-down process fills in the missing parts of the hierarchy and also adds a dense representation at the lowest level (to enable segmentation of the image). Missing parts of the hierarchy can occur for two reasons: (i) competitive exclusion may be too strict at eliminating proposals which only slightly overlap, (ii) shape variations at certain parts of the object may be big at small scales (hence rejected

by suspicious coincidences) but are removed at larger scales. For example, there are big variations locally where the legs of a horse join the torso, which make it hard to detect small scale regularities by clustering. But there are large scale regularities between the legs and the torso which are found at higher levels in the bottom-up process. The top-down process can fill in the gaps at lower level by using the higher levels connections as "context" which allow the pruning criteria to be relaxed.

A greedy strategy is used to examine every node in the hierarchy. For each node ν^l at level l, we seek to add a substructure from the dictionary (the set of proposals obtained in the bottom-up processing) at the lower level $l-1$ as the new child of ν^l so that the extended structure fits the data best (locally). This extension operation for the node ν^l is repeated until the gain is less than some threshold. The extensions correspond to adding more energy terms defined in equation (4) and (5). In figure (5), one can observe that substructure are added into the hierarchy to capture the details of the head and hind leg.

5 Experimental Results

Performance Comparisons for segmentation and parsing. We applied our approach to learn hierarchical composition models for a number of different objects. We will first concentrate on the hierarchical model for the horse, learnt from data from the Weizmann database [3] which is divided to training (12 images without labeling) and testing (316 images with groundtruth) sets. These images cover many poses (standing, running, drinking, etc.), changes in viewing angles, different scales, textured body patterns and cluttered background. Our strategy to obtain segmentation, which is inspired by Grab-Cut [13] , is to obtain the parse by the inference algorithm on the HCMs and then segment object by graph-cut using the feature statistics inside the boundary as initializations (note that, unlike us, Grab-Cut requires initialization by a human). The comparisons using segmentation accuracy are shown in table 1. The methods in [14, 15] are based on supervised learning. Only two methods [16, 15] reported higher accuracy than ours: but (i) Obj Cut[16] is tested on only five images while our method is tested on more than 300 images; (ii) [15] assumes the position of object is given for both learning and testing (Note our method does not make this assumption). [6] manually designed the model (no learning) whose performance, i.e., about 91%, is similar to [14] and inferior to ours. [17] is not listed because they manually select the best among top results (all other methods output single result). In conclusion, our method achieves the comparable (if not the best) performance even though we use far less information (we only know that the object is present somewhere in the image, while the supervised methods know the precise location of the boundaries). In addition, our model simultaneously performs other tasks such as detection and parsing (labeling different parts of the horse). See figure 4 for the typical segmentation and parsing results (we also cropped the results from [16] for comparison). Note that the color points indicate identities of object parts. Our method obtains more complete and stable boundary than [16] which learns

Method	Train	Test	Segmentation	Speed
Our method	12	316	93.3	16.9s
Ren [14]	172	172	91.0	–
Borenstein [18]	64	328	93.0	–
LOCUS [19]	20	200	93.1	–
OBJ CUT [16]	N/A	5	96.0	–
Levin [15]	N/A	N/A	95.0	–

Table 1. Results on horse dataset. Columns 2 and 3 give the size of images taken for training and testing. Column 4 quantifies the segmentation accuracy. The last column shows the average time taken for one image. Note that [16] is tested on only 5 images and [15] assumes the position of the object is roughly given.

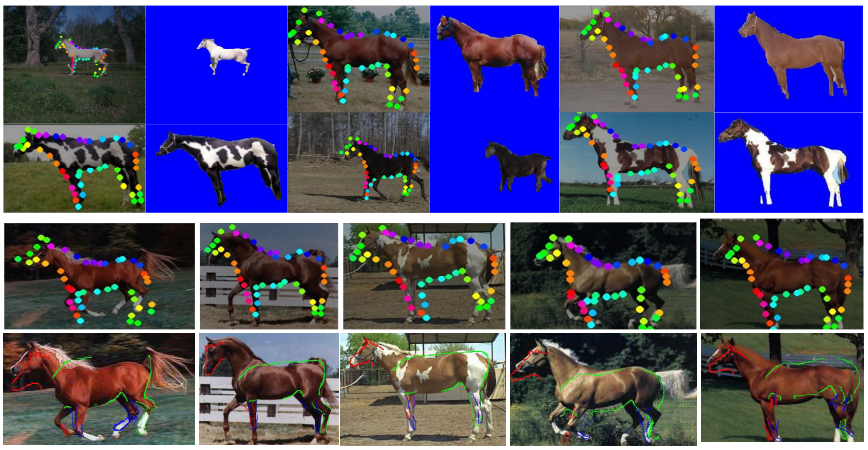

Fig. 4. In the top two rows, Columns 1 ,3 and 5 show parse results of our method followed by segmentation results. The parse result is illustrated by the colored dots which correspond to the leaf nodes of the object. The correspondences are consistent for different images. Rows 3 and 4 show the parse results of our method and OBJ CUT (cropped from [16]) on 5 images used in [16] respectively. Note our method obtains more complete boundary.

object model from video where extra motion cues are used to make learning easier. The average time of our inference algorithm including both parsing and segmentation is 17 seconds.

Study the Hierarchy. The final hierarchy for horse is depicted in the left panel of figure 5. The mean position and orientation of edgelets of leaf nodes are depicted to sketch the model learnt by the unsupervised method. The colored rectangles in the dotted box highlight the parts filled in at the top-down stage. Note that the final skeleton of horse is nearly complete while the sketches obtained by the bottom-up processing miss several parts (head, leg and back). The responses of subparts of the hierarchy is depicted in the right panel of figure 5. One can see that the numbers of proposals decrease from low level to high level.

Complexity Analysis. Our experiments also show the relative effectiveness of our two principles as we form compositions at different levels of the hierar-

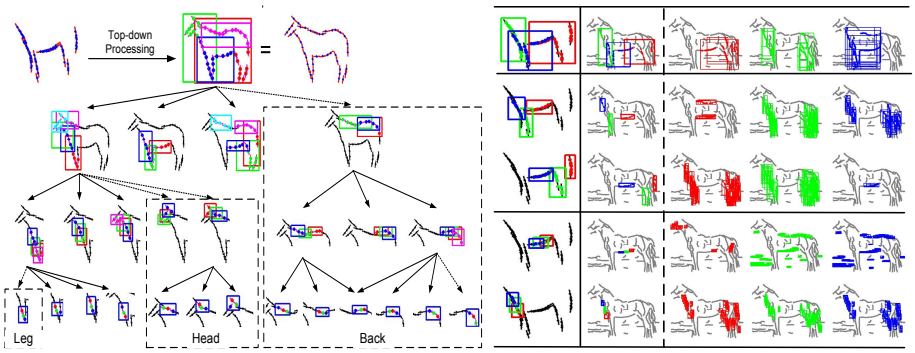

Fig. 5. Left: The hierarchy shows the learnt hierarchical model. The colored rectangles highlight the identities of the structures. All parts are obtained at the bottom-up state except that the rectangles in the dotted boxes show the parts of the model that were learnt in the top-down stage. Right: The rows illustrate structures at levels 4,3,3,2,2 (i.e. top row is level 4, next row is level 3,...). The first column gives the structure (with three children colour coded). The second column shows the structure detected on a specific image. The third, fourth, and fifth columns show the proposals for the sub-structure – colour coded.

L	Composition	Clusters	Suspicious Coincidence	Competitive Exclusion	Time
0				4	1s
1	167431	14684	262	48	117s
2	2034851	741662	995	116	254s
3	2135467	1012777	305	53	99s
4	236955	72620	30	2	9s

Table 2. Columns from 2 to 5 show the numbers of proposals after composition (step 1), clusters after clustering (step 2), after pruning (step 4) and after competitive exclusion (step 5) respectively. The next column shows the time (seconds) taken for each level. Level 0 shows the results of the grouping of the edge points(360 degrees are divided into 4 angle ranges). all numbers are calculated over 12 real images in the training dataset.

chy, see table (2). The reduction in compositions by *clustering* is fairly large (a reduction factor of 10) at level 1 but becomes negligible at higher levels as the compositions get more distinct. *Suspicious coincidences* causes major reductions in compositions with reduction factors ranging from 60 to 2000. This is because the vast number of compositions occur infrequently. *Competitive exclusion* has little effect at level 1 (a reduction factor of 5) but increases rapidly at higher levels to a reduction factor of 15 at level 4. This is because larger compositions are more likely to overlap and compete for the same niches in the images. Competitive exclusion principle is the main factor that causes the bottom-up process to stop at a specific level.

Analyze the Hierarchical Dictionary. It seems plausible that our algorithm will learn generic features (e.g., oriented straight lines, single curves, double edges, Y-junctions, T-junctions, etc.) at low-level of the hierarchy and more specific object features at the higher-levels. This is confirmed by observing the vocabularies that are extracted at different levels, see the left panel of figure (6). Recall that all these features are encoded by the same hierarchical

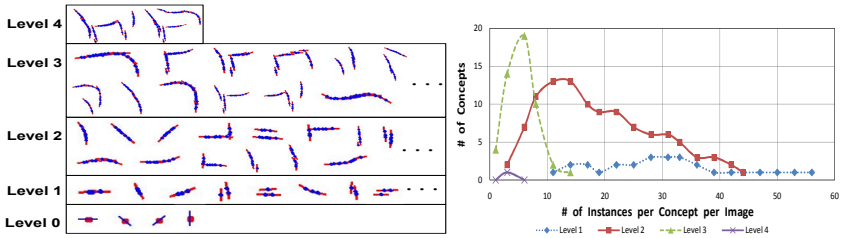

Fig. 6. Left: This figure shows elements of the vocabulary for horses at different levels (mean values only). Observe how the vocabulary contains "generic" shapes at low levels, but finds horse specific parts at the higher levels. Right: This figure shows the histogram of concepts at the different levels of the dictionary. A point in a curve quantifies the number of concepts which have a certain number of instances.

Fig. 7. This figure illustrates the generality of our approach. We show some typical training images which contain cluttered background, different shapes and deformations. The red-sketch images show the learnt models.

composition principle. In the right panel of figure (6), we also plot the distribution of the concepts at the different levels of the dictionary. The peaks of levels 4, 3, 2 and 1 appear from right to left. Note that the concepts at level 1 have only one instance per image.

More objects. To demonstrate the generality of our approach, we apply it to learn models for a range of other objects collected from Caltech 101 [20], MIT LabelMe [21] and internet, see figure (7). These images cover different types of shapes (man-made structure and animal), cluttered background (monitor, face and deer) and rotation (violin). These models were learnt using a small amount of training data (12 images per object). They can be applied to parse images using the inference algorithm. This experiment shows the versatility of our approach while modeling different types of shapes of objects.

6 Conclusion

We described a new method for unsupervised structure learning of a hierarchical compositional model for deformable objects. The structure learning was performed by a bottom-up and top-down process. We tested our approach by using it to learn hierarchical models for horses and other objects (e.g. watches, purses, faces, grand pianos, violins, revolvers, butterflies). We evaluated the resulting models by comparing their performance to alternative methods.

Acknowledgements

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

References

1. Fergus, R., Perona, P., Zisserman, A.: Object class recognition by unsupervised scale-invariant learning. In: CVPR (2). (2003) 264–271
2. Zhu, L., Chen, Y., Yuille, A.L.: Unsupervised learning of a probabilistic grammar for object detection and parsing. In: NIPS. (2006) 1617–1624
3. Borenstein, E., Ullman, S.: Class-specific, top-down segmentation. In: ECCV (2). (2002) 109–124
4. Fukushima, K.: Neocognitron: A hierarchical neural network capable of visual pattern recognition. Neural Networks **1** (1988) 119–130
5. Jin, Y., Geman, S.: Context and hierarchy in a probabilistic image model. In: CVPR (2). (2006) 2145–2152
6. Chen, Y., Zhu, L., Lin, C., Yuille, A.L., Zhang, H.: Rapid inference on a novel and/or graph for object detection, segmentation and parsing. In: NIPS. (2007)
7. Hinton, G.E., Osindero, S., Teh, Y.W.: A fast learning algorithm for deep belief nets. Neural Computation **18** (2006) 1527–1554
8. Epshtein, B., Ullman, S.: Feature hierarchies for object classification. In: ICCV. (2005) 220–227
9. Serre, T., Wolf, L., Poggio, T.: Object recognition with features inspired by visual cortex. In: CVPR (2). (2005) 994–1000
10. Ahuja, N., Todorovic, S.: Learning the taxonomy and models of categories present in arbitrary image. In: ICCV. (2007)
11. Fleuret, F., Geman, D.: Coarse-to-fine face detection. In: IJCV. (2001)
12. Fidler, S., Leonardis, A.: Towards scalable representations of object categories: Learning a hierarchy of parts. In: CVPR. (2007)
13. Rother, C., Kolmogorov, V., Blake, A.: "grabcut": interactive foreground extraction using iterated graph cuts. ACM Trans. Graph. **23** (2004) 309–314
14. Ren, X., Fowlkes, C., Malik, J.: Cue integration for figure/ground labeling. In: NIPS. (2005)
15. Levin, A., Weiss, Y.: Learning to combine bottom-up and top-down segmentation. In: ECCV (4). (2006) 581–594
16. Kumar, M.P., Torr, P.H.S., Zisserman, A.: Obj cut. In: CVPR (1). (2005) 18–25
17. Cour, T., Shi, J.: Recognizing objects by piecing together the segmentation puzzle. In: CVPR. (2007)
18. Borenstein, E., Malik, J.: Shape guided object segmentation. In: CVPR (1). (2006) 969–976
19. Winn, J.M., Jojic, N.: Locus: Learning object classes with unsupervised segmentation. In: ICCV. (2005) 756–763
20. Fei-Fei, L., Fergus, R., Perona, P.: Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. Comput. Vis. Image Underst. **106** (2007) 59–70
21. Russell, B., Torralba, A., Murphy, K., Freeman, W.: Labelme: a database and web-based tool for image annotation. Technical Report (2005)