Unsupervised Learning of Probabilistic Grammar-Markov Models for Object Categories

Long (Leo) Zhu, Yuanhao Chen, and Alan Yuille

Abstract—We introduce a Probabilistic Grammar-Markov Model (PGMM) which couples probabilistic context-free grammars and Markov Random Fields. These PGMMs are generative models defined over attributed features and are used to detect and classify objects in natural images. PGMMs are designed so that they can perform rapid inference, parameter learning, and the more difficult task of structure induction. PGMMs can deal with unknown 2D pose (position, orientation, and scale) in both inference and learning, different appearances, or aspects of the model. The PGMMs can be learned in an unsupervised manner, where the image can contain one of an unknown number of objects of different categories or even be pure background. We first study the weakly supervised case, where each image contains an example of the (single) object of interest, and then generalize to less supervised cases. The goal of this paper is theoretical, but, to provide proof of concept, we demonstrate results from this approach on a subset of the Caltech data set (learning on a training set and evaluating on a testing set). Our results are generally comparable with the current state of the art and our inference is performed in less than five seconds.

Index Terms—Computer vision, structural models, grammars, Markov random fields, object recognition.

1 Introduction

Remarkable progress in the mathematics and computer science of probability is leading to a revolution in the scope of probabilistic models. There are exciting new probability models defined on structured relational systems, such as graphs or grammars [1], [2], [3], [4], [5], [6]. Unlike more traditional models such as Markov Random Fields (MRFs) [7] and Conditional Random Fields (CRFs) [2], these models are not restricted to having fixed graph structures. Their ability to deal with varying graph structure means that they can be applied to model a large range of complicated phenomena, as has been shown by their applications to natural languages [8], machine learning [6], and computer vision [9].

Our long-term goal is to provide a theoretical framework for the unsupervised learning of probabilistic models for generating, and interpreting, natural images [9]. This is somewhat analogous to Klein and Manning's work on unsupervised learning of natural language grammars [3]. In particular, we hope that this paper can help bridge the gap between computer vision and related work on grammars in machine learning [8], [1], [6]. There are, however, major differences between vision and natural language processing. First, images are arguably far more complex than sentences, so learning a probabilistic model to generate natural images is too ambitious to start with. Second, even if we restrict ourselves to the simpler task of generating an image containing a single object, we must deal with 1) the cluttered background (similar to learning a natural language grammar when the input contains random symbols, as well as words), 2) the unknown 2D pose (size, scale, and position) of the object, and 3) different appearances, or *aspects*, of the object (these aspect deal with changes due to different 3D poses of the object, different photometric appearance, different 2D shapes, or combinations of these factors). Third, the input is a set of image intensities and is considerably more complicated than the limited types of speech tags (e.g., nouns, verbs, etc.) used as input in [3].

In this paper, we address an important subproblem. We are given a set of images containing one of an unknown number of objects (with variable 2D pose) of different categories or even pure background. The object is allowed to have several different appearances or aspects. We call this *unsupervised learning* in contrast to *weakly supervised learning*, where each image contains an example of a single object (but the position and boundary of the object are unknown). We represent these images in terms of attributed features (AFs). The task is to learn a probabilistic model for generating the AFs (both those of the object and the background). We require that the probability model must allow 1) rapid inference (i.e., interpret each image), 2) rapid parameter learning, and 3) *structure induction*, where the structure of the model is unknown and must be grown in response to the data.

To address this subproblem, we develop a Probabilistic Grammar Markov Model (PGMM) that is motivated by this goal and its requirements. The PGMM combines elements of MRFs [7] and probabilistic context-free grammars (PCFGs) [8]. The requirement that we can deal with a variable number of AFs (e.g., caused by different aspects of the object) motivates the use of grammars (instead of fixed graph models like MRFs). However, PCFGs, see Fig. 1, are inappropriate because they make independent assumptions on the production rules and hence must be supplemented by MRFs to model the spatial relationships between AFs of the object. The requirement that we deal with a 2D pose (both for learning and inference) motivates the use of

- *L. Zhu and A. Yuille are with the Department of Statistics, University of California, Los Angeles, 8125 Math Science Bldg., Los Angeles, CA 90095. E-mail: {lzhu, yuille}@stat.ucla.edu.*
- *Y. Chen is with the Department of Automation, University of Science and Technology of China, Heifei 230026, China. E-mail: yhchen4@ustc.edu.*

Manuscript received 5 Mar. 2007; revised 11 Oct. 2007; accepted 22 Jan. 2008; published online 11 Mar. 2008.
Recommended for acceptance by P. Torr.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number TPAMI-2007-03-0146.
Digital Object Identifier no. 10.1109/TPAMI.2008.67.

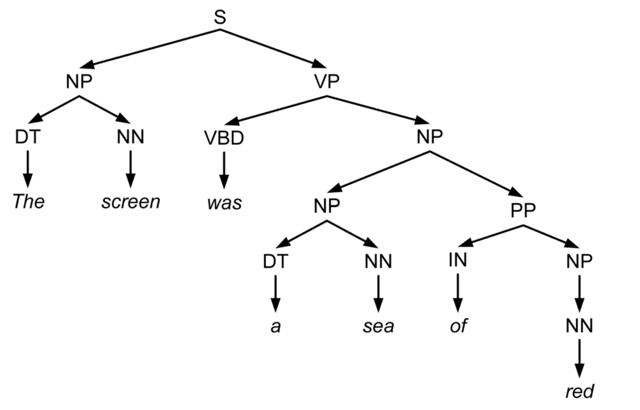

Fig. 1. Probabilistic context-free grammar. The grammar applies production rules with the probability to generate a tree structure. Different random sampling will generate different tree structures. The production rules are applied independently on different branches of the tree. There are no sideways relations between nodes.

oriented triangles of AFs as our basic building blocks for the probabilistic model, see Fig. 2. These oriented triangles are represented by features such as the internal angles of the triangle that are invariant to the 2D pose of the object in the image. The requirement that we can perform rapid inference on new images is achieved by combining the triangle building blocks to enable dynamic programming (DP). The ability to perform rapid inference ensures that parameter estimation and structure learning is practical.

We decompose the learning task into 1) learning the structure of the model and 2) learning the parameters of the model. Structure learning is the more challenging task [8], [1], [6] and we propose a structure induction (or structure pursuit) strategy that proceeds by building an AND-OR graph [4], [5] in an iterative way by adding more triangles or OR nodes (for different aspects) to the model. We use clustering techniques to make proposals for adding triangles/OR nodes and validate or reject these proposals by model selection. The clustering techniques relate to Barlow's idea of suspicious coincidences [10].

We evaluate our approach by testing it on parts of the Caltech 4 (faces, motorbikes, airplanes, and background) [11] and Caltech 101 database [12]. Performance on this database has been much studied [11], [13], [14], [15], [16]. However, we stress that the goal of our paper is to develop a novel theory and test it, rather than simply trying to get better performance on a standard database. Nevertheless, our experiments show three major results. First, we can learn PGMMs for a number of different objects and obtain performance results close to the state of the art. Moreover, we can also obtain good localization results (which is not always possible with other methods). The speed of inference is under five seconds. Second, we demonstrate our ability to do learning and inference independent of the scale and orientation of the object (we do this by artificially scaling and rotating images from Caltech 101 (Fig. 3), lacking a standard database where these variations occur naturally). Third, the approach is able to learn from noisy data (where half of the training data is only background images) and to deal with object classes, which we illustrate by learning a hybrid class consisting of faces, motorbikes, and airplane.

This paper is organized as follows: We first review the background in Section 2. Section 3 describes the features we use to represent the images. In Section 4, we give an overview of PGMMs. Section 5 specifies the probability distributions defined over the PGMM. In Section 6, we describe the algorithms for inference, parameter learning, and structure learning. Section 7 illustrates our approach by learning models for 38 objects, demonstrating invariance to scale and rotation and performing learning for object classes.

2 Background

This section gives a brief review of the background in machine learning and computer vision.

Structured models define a probability distribution on structured relational systems such as graphs or grammars. This includes many standard models of probability distributions defined on graphs—for example, graphs with fixed structure such as MRFs [7], CRFs [2], or PCFGs [8], where the graph structure is variable. Attempts have been made to unify these approaches under a common formulation. For example, Case-Factor Diagrams [1] have recently been proposed as a framework which subsumes both MRFs and PCFGs. In this paper, we will be concerned with models that combine probabilistic grammars with MRFs. The grammars are based on AND-OR graphs [1], [4], [5], which relate to mixtures of trees [17]. This merging of MRFs with probabilistic grammars results in structured models that have the advantages of variable graph structure (e.g., from PCFGs) combined with the rich spatial structure from the MRFs.

There has been considerable interest in inference algorithms for these structured models; for example, McAllester et al. [1] describe how DP algorithms (e.g., Viterbi and inside-outside) can be used to rapidly compute properties of interest

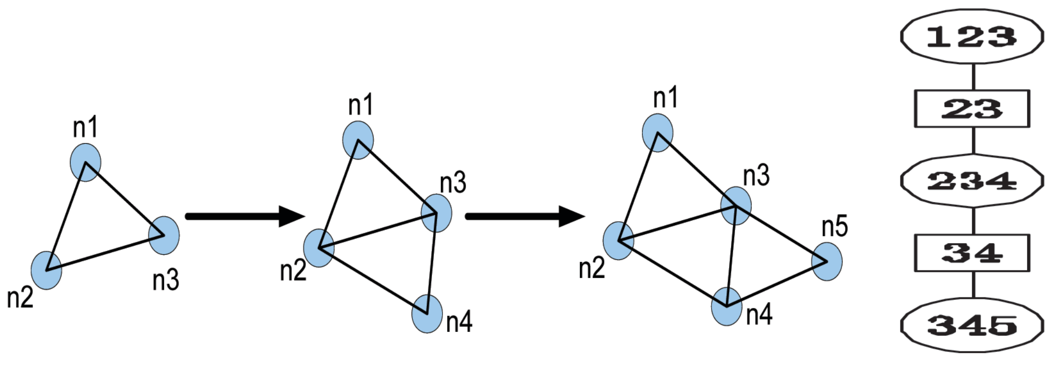

Fig. 2. This paper uses triplets of nodes as building blocks. We can grow the structure by adding new triangles. The junction tree (the far right panel) is used to represent the combination of the triplets to allow efficient inference.

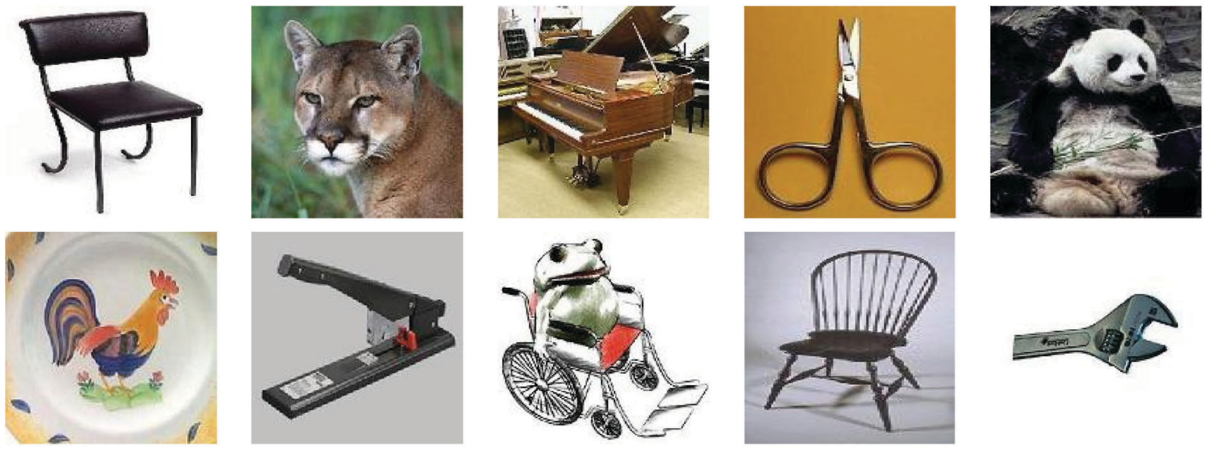

Fig. 3. Ten of the object categories from Caltech 101 that we learn in this paper.

for Case-Factor diagrams. However, inference on arbitrary models combining PCFGs and MRFs remains difficult.

The task of learning and, particularly, structure induction is considerably harder than inference. For MRF models, the number of graph nodes is fixed and structure induction consists of determining the connections between the nodes and the corresponding potentials. For these graphs, an effective strategy is feature induction [18], which is also known as feature pursuit [19]. A similar strategy is also used to learn CRFs [20], where the learning is fully supervised. For Bayesian network, there is work on learning the structure using the EM algorithm [21].

Learning the structure of grammars in an unsupervised way is more difficult. Klein and Manning [3] have developed the unsupervised learning of PCFGs for parsing natural language, but, here, the structure of grammar is specified. Zettlemoyer and Collins [6] perform similar work based on lexical learning with lambda-calculus language.

To our knowledge, there is no unsupervised learning algorithm for structure induction for any PGMM. However, an extremely compressed version of part of our work appeared in [22].

There has been a considerable amount of work on learning MRF models for visual tasks such as object detection. An early attempt was described in [23]. The constellation model [11] is a nice example of a weakly supervised algorithm that represents objects by a fully connected (fixed) graph. Crandall et al. [14], [15] explore different simpler MRF structures, such as k-fans models, which enable rapid inference.

There is also a great deal of literature [11], [13], [14], [15], [16] on computer vision models for performing object recognition, many of which have been evaluated on the Caltech databases [11]. Indeed, there is a whole range of computer vision methods that have been evaluated on the Caltech database [12]. A review of the performance and critiques of the database is given in [24]. A major concern is that the nature of this data set enables overgeneralization, for example, the models can use features that occur in the background of the image and not within the object.

3 The Image Representation: Features and Oriented Triplets

In this paper, we will represent images in terms of isolated AFs, which will be described in Section 3.1. A key ingredient of our approach is to use conjunctions of features and, in particular, triplets of features with associated angles at the vertices which we call *oriented triplets*, see Figs. 4 and 5. The advantages of using conjunctions of basic features is well known in natural language processing and leads to unigram, bigram, and trigram features [8].

There are several reasons for using oriented triplets in this paper. First, they contain geometrical properties that are invariant to the scale and rotation of the triplet. These properties include the angles between the vertices and the relative angles at the vertices, see Figs. 4 and 5. These properties can be used both for learning and inference of a PGMM when the scale and rotation are unknown. Second, they lead to a representation that is well suited to DP, similar to the junction tree algorithm [25], which enables rapid inference, see Figs. 6 and 2. Third, they are well suited to the task of structure pursuit since we can combine two oriented triplets by a common edge to form a more complex model, see Figs. 2 and 6.

3.1 The Image Features

We represent an image by AFs $\{x_i : i = 1,..,N_\tau\}$, where N_τ is the number of features in image $\mathbf{I}_\tau$ with $\tau \in \Lambda$, where Λ is the set of images. Each feature is represented by a triple $x_i = (z_i, \theta_i, A_i)$, where z_i is the location of the feature in the image, θ_i is the orientation of the feature, and A_i is an appearance vector.

These features are computed as follows: We apply the Kadir-Brady [26] operator K_b to select circular regions $\{C^i(\mathbf{I}_\tau) : i = 1, \dots, N_\tau\}$ of the input image $\mathbf{I}_\tau$ such that $K_b(C^i(\mathbf{I}_\tau)) > \mathrm{T}, \forall\, i$, where T is a fixed threshold. We scale

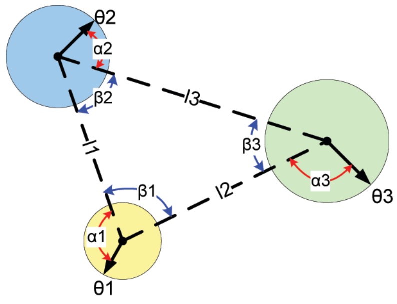

Fig. 4. The oriented triplet is specified by the internal angles β, the orientation of the vertices θ, and the relative angles α between them.

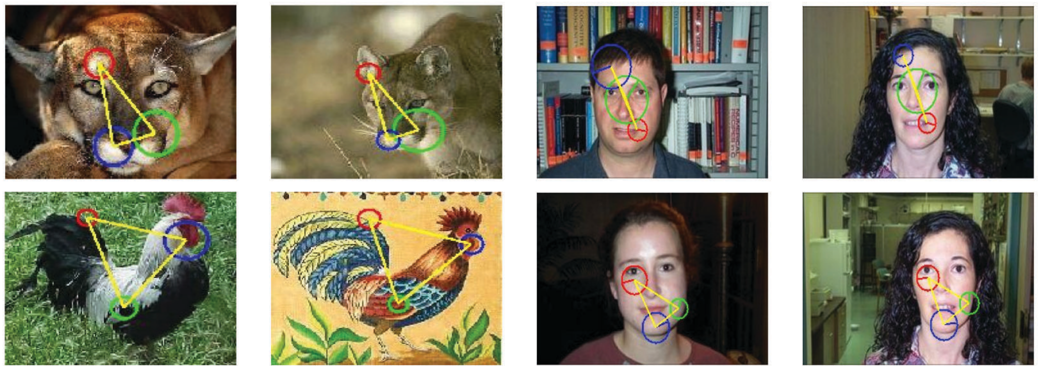
Fig. 5. This figure illustrates the features and triplets without orientation (left two panels) and oriented triplets (next two panels).

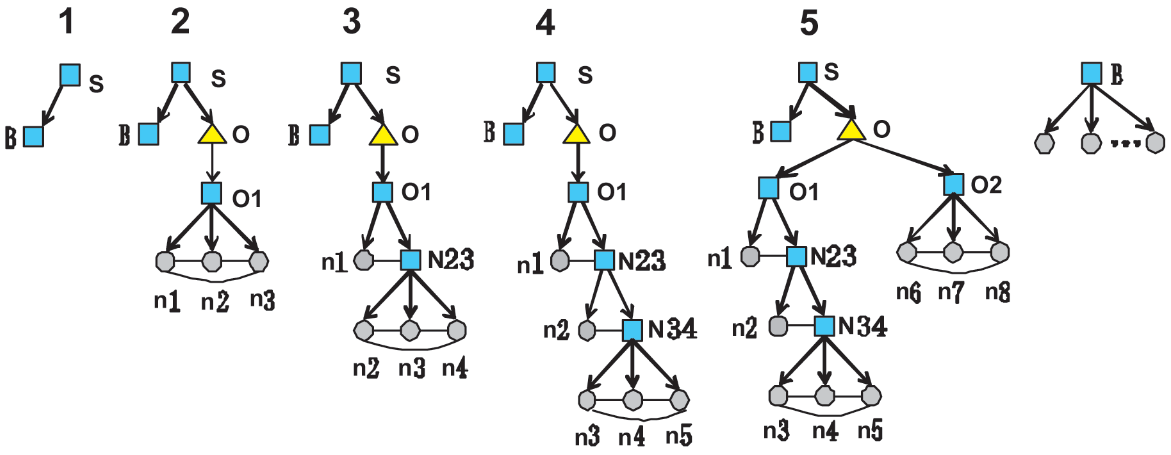

Fig. 6. Graphical models. Squares, triangles, and circles indicate AND, OR, and LEAF nodes, respectively. The horizontal lines denote MRF connections. The far right panel shows the background node generating leaf nodes. The models for $O1$ for panels 2, 3, and 4 correspond to the triplets combinations in Fig. 2. See text for notation.

these regions to a constant size to obtain a set of scaled regions $\{\hat{C}^i(\mathbf{I}_\tau) : i = 1, \ldots, N_\tau\}$. Then, we apply the SIFT operator $L(.)$ [27] to obtain Lowe's feature descriptor $L_i = L(\hat{C}^i(\mathbf{I}_\tau))$ together with an orientation θ_i (also computed by [27]) and set the feature position z_i to be the center of the window C^i. Then, we perform PCA on the appearance attributes (using the data from all images $\{\mathbf{I}_\tau : \tau \in \Lambda\}$) to obtain a 15-dimensional subspace (a reduction from 128 dimensions). Projecting L_i into this subspace gives us the appearance attribute A_i.

The motivation for using these operators is given as follows: First, the Kadir-Brady operator is an *interest operator* that selects the parts of the image that contain interesting features (e.g., edges, triple points, and textured structures). Second, the Kadir-Brady operator adapts geometrically to the size of the feature and, hence, is scale invariant. Third, the SIFT operator is also (approximately) invariant to a range of photometric and geometric transformations of the feature. In summary, the features occur at interesting points in the image and are robust to photometric and geometric transformations.

3.2 The Oriented Triplets

An oriented triplet of three feature points has a geometry specified by $(z_i, \theta_i, z_j, \theta_j, z_k, \theta_k)$ and is illustrated in Figs. 4 and 5. We construct a 15-dimensional *invariant triplet vector* $\vec{l}$ that is invariant to the scale and rotation of the oriented triplet:

$$\begin{aligned}\vec{l}(z_i, \theta_i, z_j, \theta_j, z_k, \theta_k) = (&l_1/L, l_2/L, l_3/L,\\ &\cos\alpha_1, \sin\alpha_1, \cos\alpha_2, \sin\alpha_2, \cos\alpha_3, \sin\alpha_3,\\ &\cos\beta_1, \sin\beta_1, \cos\beta_2, \sin\beta_2, \cos\beta_3, \sin\beta_3),\end{aligned} \tag{1}$$

where l_1, l_2, and l_3 are the length of the three edges, $L = l_1 + l_2 + l_3$, α_1, α_2, and α_3 are the relative angles between the orientations θ_i, θ_j, and θ_k and the orientations of the three edges of the triangle, and β_1, β_2, and β_3 are the angles between edges of the triangle (hence, $\beta_1 + \beta_2 + \beta_3 = \pi$).

This representation is overcomplete. However, we found empirically that it was more stable than lower dimensional representations. If rotation and scale invariance are not needed, then we can use alternative representations of triplets such as $(l_1, l_2, l_3, \theta_1, \theta_2, \theta_3, \beta_1, \beta_2, \beta_3)$. Previous authors [28], [29] have used triples of features, but, to our knowledge, oriented triplets are novel.

4 Probalistic Grammar-Markov Model

We now give an overview of the PGMM, which has characteristics of both a probabilistic grammar, such as a PCFG, and an MRF. The probabilistic grammar component of the PGMM specifies different topological structures, as illustrated in the five leftmost panels in Fig. 6, enabling the ability to deal with variable number of AFs. The MRF component specifies spatial relationships and is indicated by the horizontal connections.

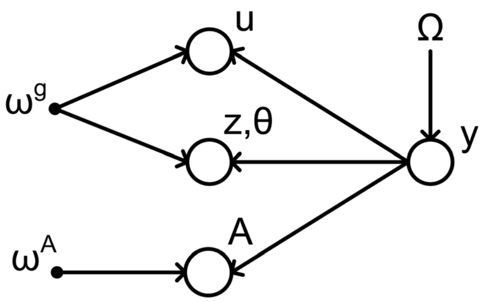

Fig. 7. This figure illustrates the dependencies between the variables. The variables Ω specify the probability for topological structure y. The spatial assignments z of the leaf nodes are influenced by the topological structure y and the MRF variables ω. The probability distribution for the image features x depends on y, ω, and z.

Formally, we represent a PGMM by a graph $G = (V, E)$, where V and E denote the set of vertices and edges, respectively. The vertex set V contains three types of nodes, "OR" nodes, "AND" nodes, and "LEAF" nodes, which are depicted in Fig. 6 by triangles, rectangles, and circles, respectively. The edge set E contains vertical edges defining the topological structure and horizontal edges defining spatial constraints (e.g., MRFs).

The leaf nodes are indexed by a and will correspond to AFs in the image. They have attributes (z_a, θ_a, A_a), where z_a denotes the spatial position, θ_a denotes the orientation, and A_a denotes the appearance. There is also a binary-valued *observability variable* u_a that indicates whether the node is observable in the image (a node may be unobserved because it is occluded or the feature detector has too high a threshold). We set y to be the parse structure of the graph when the OR nodes take specific assignments. We decompose the set of leaves $L(y) = L_B(y) \bigcup L_O(y)$, where $L_B(y)$ are the leaves due to the background model, see the far right panel in Fig. 6, and $L_O(y)$ are the leaves due to the object. We order the nodes in $L_O(y)$ by "dropout," so that the closer the node is to the root, the lower its number is, see Fig. 6.

In this paper, the only OR node is the object category node O. This corresponds to different aspects of the object. The remaining nonterminal nodes are AND nodes. They include a background node B, object aspect nodes O_i, and *clique nodes* of the form $N_{a,a+1}$ (containing points n_a, n_{a+1}). Each aspect O_i corresponds to a set of object leaf nodes $L_O(y)$ with corresponding cliques $C(L_O(y))$. As shown in Fig. 6, each clique node $N_{a,a+1}$ is associated with a leaf node n_{a+2} to form a *triplet-clique* C_a $\{n_a, n_{a+1}, n_{a+2}\}$.

The directed (vertical) edges connect nodes at successive levels of the tree. They connect

1. the root node S to the object node and the background node,
2. the object node to aspect nodes,
3. a nonterminal node to three leaf nodes, see panel 2 in Fig. 6, or
4. a nonterminal node to a clique node and a leaf node, see panel 3 in Fig. 6.

In cases 3 and 4, they correspond to a triplet clique of point features.

Fig. 6 shows examples of PGMMs. The top rectangle node S is an AND node. The simplest case is a pure background model, in panel 1, where S has a single child node B that has an arbitrary number of leaf nodes corresponding to feature points. In the next model, panel 2, S has two child nodes representing the background B and the object category O. The category node O is an OR node that is represented by a triangle. The object category node O has a child node, O_1, which has a triplet of child nodes corresponding to point features. The horizontal line indicates spatial relations of this triplet. The next two models, panels 3 and 4, introduce new feature points and new triplets. We can also introduce a new aspect of the object O_2, see panel 5, to allow for the object having a different appearance.

TABLE 1
The Notations Used for the PGMM

Notation	Meaning
Λ	the set of images
$x_i = (z_i, \theta_i, A_i)$	an attributed feature (AF)
$\{x_i : i = 1, ..., N_\tau\}$	attributed features of image I_τ
N_τ	the number of features in image I_τ
z_i	the location of the feature
θ_i	the orientation of the feature
A_i	the appearance vector of the feature
y	the topological structure
a	the index of the node
n_a	the leaf nodes of PGMM
$C_a = \{n_a, n_{a+1}, n_{a+2}\}$	a triplet clique
$\vec{l}_C()$	the invariance triplet vector of clique C
$u = \{u_a\}$	observability variables
Ω	the parameters of grammatical part
ω	(ω^g, ω^A)
ω^g	the parameters of spatial relation of leaf nodes
ω^A	the parameters of appearances of the AF's
$V = \{i(a)\}$	the correspondence variables

5 The Distribution Defined on the PGMM

The structure of the PGMM is specified by Fig. 7. The PGMM specifies the probability distribution of the AFs observed in an image in terms of parse graph y and model parameters Ω, ω for the grammar and the MRF, respectively. The distribution involves additional hidden variables that include the pose G and the observability variables $u = \{u_a\}$. We set $z = \{z_a\}$, $A = \{A_a\}$, and $\theta = \{\theta_a\}$. See Table 1 for the notation used in the model.

We define the full distribution to be

$$\begin{aligned} P(u, z, A, \theta, y, \omega, \Omega) = P(A|y, \omega^A) P(z, \theta|y, \omega^g) \\ P(u|y, \omega^g) P(y|\Omega) P(\omega) P(\Omega). \end{aligned} \tag{2}$$

The observed AFs are those for which $u_a = 1$. Hence, the observed image features $x = \{(z_a, A_a, \theta_a) : \text{ s.t. } u_a = 1\}$. We can compute the joint distribution over the observed image features x by

$$\begin{aligned} &P(x, y, \omega, \Omega) \\ &= \sum_{\{(z_a, A_a, \theta_a) : \text{ s.t.u}_a = 0\}} P(u, z, A, \theta, y, \omega, \Omega). \end{aligned} \tag{3}$$

We now briefly explain the different terms in (2) and refer to the following sections for details.

$P(y|\Omega)$ is the grammatical part of the PGMM (with prior $P(\Omega)$). It generates the topological structure y that specifies which aspect model O_i is used and the number of background nodes. The term $P(u|y, \omega^g)$ specifies the probability that the leaf nodes are observed (background nodes are always observed). $P(z, \theta|\omega^g)$ specifies the probability of the spatial positions and orientations of the leaf nodes. The distributions on the object leaf nodes are specified in terms of the invariant shape vectors defined on the triplet cliques, while the background leaf nodes are generated independently. Finally, the distribution $P(A|y,\omega^A)$ generates the appearances of the AFs. $P(\omega^g,\omega^A)$ is the prior on ω.

5.1 Generating the Leaf Nodes: $P(y|\Omega)$

This distribution $P(y|\Omega)$ specifies the probability distribution of the leaf nodes. It determines how many AFs are present in the image (except for those that are unobserved due to occlusion or falling below threshold). The output of y is the set of numbered leaf nodes. The numbering determines the object nodes $L_O(y)$ (and the aspects of the object) and the background nodes $L_B(O)$. (The attributes of the leaf nodes are determined in later sections.)

$P(y|\Omega)$ is specified by a set of production rules. In principle, these production rules can take any form, such as those used in PCFGs [8]. Other possibilities are Dechter's AND-OR graphs [4], case-factor diagrams [1], composite templates [5], and compositional structures [30]. In this paper, however, we restrict our implementation to rules of form:

$$\begin{aligned}
&\text{S} \rightarrow \{B,O\} \text{ with prob } 1,\\
&\text{O} \rightarrow \{O_j : j=1,\dots,\rho\} \text{ with prob, } \Omega^O_j,\ j=1,\dots,\rho\\
&\text{O}_j \rightarrow \{n_a, N_{a+1,a+2}\} \text{ with prob. } 1,\ a=\beta_j,\\
&\text{N}_{a,a+1} \rightarrow \{n_a, N_{a+1,a+2}\} \text{ with prob. } 1,\\
&\beta_i+1 \le a \le \beta_{j+1}-4.\\
&\text{N}_{\beta_{j+1}-3,\beta_{j+1}-2} \rightarrow \{n_{\beta_{j+1}-2}, n_{\beta_{j+1}-1}\} \text{ with prob } 1,\\
&\text{B} \rightarrow \{n_{\beta_{\rho+1}},\dots,n_{\beta_{\rho+1}+m}\}\\
&\text{with prob } \Omega^B e^{-m\Omega^B}\ (m=0,1,2\dots).
\end{aligned} \tag{4}$$

Here, $\beta_1 = 1$. The nodes $\beta_j,\dots,\beta_{j+1}-1$ correspond to aspect O_j. Note that these $\{\beta_j\}$ are parameters of the model that will be learned. ρ is the number of aspects and $\{\Omega^O_j\}$ and Ω^B are parameters that specify the distribution (all of these will be learned). We write $\Omega = \{\Omega^B,\Omega^O_1,\dots,\Omega^O_\rho,\beta_1,\dots,\beta_{\rho+1},\rho\}$. These rules are illustrated in Fig. 6 (note that, for the simplicity of the figure, we represent the combination $N_{a,a+1}\mapsto\{n_a,N_{a+1,a+2}\}$ *and* $N_{a+1,a+2}$ by $N_{a,a+1}\mapsto(n_a,n_{a+1},n_{a+2})$).

5.2 Generating the Observable Leaf Nodes: $P(u|y,\omega^g)$

The distribution $P(u|y,\omega^g)$ specifies whether objects leafs are observable in the image (all background nodes are assumed to be observed). The observation variable u allows for the possibility that an object leaf node a is unobserved due to occlusion or because the feature detector response falls below threshold. Formally, $u_a = 1$ if the object leaf node a is observed and $u_a = 0$ otherwise. We assume that the observability of nodes are independent:

$$\begin{aligned}
P(u|y,\omega^g) &= \prod_{a\in L_O(y)} \lambda_\omega^{u_a}(1-\lambda_\omega)^{(1-u_a)}\\
&= \exp\left\{\sum_{a\in L_O(y)} \{\delta_{u_a,1}\log\lambda_\omega + \delta_{u_a,0}\log(1-\lambda_\omega)\}\right\},
\end{aligned} \tag{5}$$

where λ_ω is the parameter of the Bernoulli distribution, and $\delta_{u_a,1}$ are the Kronecker delta function (i.e., $\delta_{u_a,1}=0$ unless $u_a = 1$).

5.3 Generating the Positions and Orientation of the Leaf Nodes: $P(z,\theta|y,\omega^g)$

$P(z,\theta|y,\omega^g)$ is the distribution of the spatial positions z and orientations θ of the leaf nodes. We assume that the spatial positions and orientations of the background leaf nodes are independently generated from a uniform probability distribution.

The distribution on the position and orientations of the object leaf nodes is required to satisfy two properties: 1) It is invariant to the 2D pose (position, orientation, and scale) and 2) it is easily computable. In order to satisfy both of these properties, we make an approximation. We first present the distribution that we use and then explain its derivation and the approximation involved.

The distribution is given by

$$P(z,\theta|y,\omega^g) = K \times P(l(z,\theta)|y,\omega^g), \tag{6}$$

where $P(l(z,\theta)|y,\omega^g)$ (see (7)) is a distribution over the invariant shape vectors l computed from the spatial positions z and orientations θ. We assume that K is a constant. This is an approximation because the full derivation, see below, has $K(z,\theta)$.

We define the distribution $P(z,\theta|y,\omega^g)$ over l to be a Gaussian distribution defined on the cliques:

$$P(l|y,\omega^g) = \frac{1}{Z}\exp\left\{\sum_{a\in Cliques(y)} \psi_a(\vec{l}_a,\omega^g_a)\right\}, \tag{7}$$

where $\vec{l}_a = \vec{l}(z_a,\theta_a,z_{a+1},\theta_{a+1},z_{a+2},\theta_{a+2})$, the triplet cliques are $C_1,\dots,C_{\tau-2}$, where $C_a = (n_a,n_{a+1},n_{a+2})$. The invariant triplet vector $\vec{l}_a$ is given by (1).

The potential $\psi_a(\vec{l}_a,\omega^g_a)$ specifies geometric regularities of clique C_a that are invariant to the scale and rotation. They are of the form

$$\psi_a(\vec{l}_a,\omega^g_a) = -(1/2)(\vec{l}_a - \vec{\mu}^z_a)^T(\Sigma^z_a)^{-1}(\vec{l}_a - \vec{\mu}^z_a), \tag{8}$$

where $\omega^g_a = (\mu^z_a,\Sigma^z_a)$ and $\omega^g = \{\omega^g_a\}$.

Now, we derive (6) for $P(z,\theta|y,\omega^g)$ and explain the nature of the approximation. First, we introduce a pose variable G, which specifies the position, orientation, and scale of the object. We set

$$P(z,\theta,\vec{l},G|y,\omega^g) = P(z,\theta|l,G)P(l|y,\omega^g)P(G), \tag{9}$$

where the distribution $P(z,\theta|l,G)$ is of the form

$$P(z,\theta|l,G) = \delta_{z,z(G,l)}\delta_{\theta,\theta(G,l)}, \tag{10}$$

where $\delta_{a,b}$ denotes the function $\delta(a-b)$. $P(z,\theta|l,G)$ specifies the positions and orientations z,θ by deterministic functions $z(l,G)$, $\theta(l,G)$ of the pose G and shape invariant vectors l. We can invert this function to compute $l(z,\theta)$ and $G(z,\theta)$ (i.e., to compute the invariant feature vectors and the pose from the spatial positions and orientations z,θ).

We obtain $P(z,\theta|y,\omega^g)$ by integrating out l,G:

$$P(z,\theta|y,\omega^g) = \int dG \int dl P(z,\theta,l,G|y,\omega^g). \tag{11}$$

Substituting (10) and (9) into (11) yields

$$\begin{aligned} &P(z,\theta|y,\omega^z) \\ &= \int dG \int dl \delta_{z,z(l,G)} \delta_{\theta,\theta(l,G)} P(l|y,\omega^g) P(G) \\ &= \int\int d\rho d\gamma \frac{\partial(l,G)}{\partial(\rho,\gamma)} \delta_{z,\rho}\delta_{\theta,\gamma} P(l_{z,\theta}|y,\omega^g) P(G_{z,\theta}) \\ &= \frac{\partial(l,G)}{\partial(\rho,\gamma)}(z,\theta) P(l_{z,\theta}|y,\omega^g) P(G_{z,\theta}), \end{aligned} \tag{12}$$

where $l_{z,\theta} = l(z,\theta)$, $G_{z,\theta} = G(z,\theta)$, and we performed a change of integration from variables (l,G) to new variables (ρ,γ) with $\rho = z(l,G)$, $\gamma = \theta(l,G)$, and where $\frac{\partial(l,G)}{\partial(\rho,\gamma)}(z,\theta)$ is the Jacobian of this transformation (evaluated at (z,θ)).

To obtain the form in (6), we simplify (12) by assuming that $P(G)$ is the uniform distribution and by making the approximation that the Jacobian factor is independent of (z,θ) (this approximation will be valid provided the size and shapes of the triplets do not vary too much).

5.4 The Appearance Distribution $P(A|y,\omega^A)$

We now specify the distribution of the appearances $P(A|y,\omega^A)$. The appearances of the background nodes are generated from a uniform distribution. For the object nodes, the appearance A_a is generated by a Gaussian distribution specified by $\omega_a^A = (\mu_a^A, \Sigma_a^A)$:

$$P(A_a|\omega_a^A) = \frac{1}{\sqrt{2\pi}|\Sigma_a^A|} \exp \phi_a, \tag{13}$$

where $\phi_a = -(1/2)(A_a - \mu_a^A)^T (\Sigma_a^A)^{-1} (A_a - \mu_a^A)$.

5.5 The Priors: $P(\Omega), P(\omega^A), P(\omega^g)$

The prior probabilities are set to be uniform distributions, except for the priors on the appearance covariances Σ_a^A that are set to zero mean Gaussians with fixed variance.

5.6 The Correspondence Problem

Our formulation of the probability distributions has assumed an ordered list of nodes indexed by a. However, these indices are specified by the model and cannot be observed from the image. Indeed, performing inference requires us to solve a correspondence problem between the AFs in the image and those in the model. This correspondence problem is complicated because we do not know the aspect of the object and some of the AFs of the model may be unobservable.

We formulate the correspondence problem by defining a new variable $V = \{i(a)\}$. For each $a \in L_O(y)$, the variable $i(a) \in \{0, 1, \ldots, N_\tau\}$, where $i(a) = 0$ indicates that a is unobservable (i.e., $u_a = 0$). For background leaf nodes, $i(a) \in \{1, \ldots, N_\tau\}$. We constrain all image nodes to be matched so that $\forall j \in \{1, \ldots, N_\tau\}$ there exists a unique $b \in L(y)$ such that $i(b) = j$ (we create as many background nodes as is necessary to ensure this). To ensure uniqueness, we require that the object triplet nodes all have unique matches in the image (or are unmatched) and that background nodes can only match AFs that are not matched to object nodes or to other background nodes. (It is theoretically possible that object nodes from different triplets might match the same image AF. However, this is extremely unlikely due to the distribution on the object model and we have never observed it.)

Using this new notation, we can drop the u variable in (5) and replace it by V with prior

$$P(V|y,\omega^g) = \frac{1}{\hat{Z}} \prod_a \exp\{-\log\{\lambda_\omega/(1-\lambda_\omega)\}\delta_{i(a),0}\}. \tag{14}$$

This gives the full distribution (see (2), which is defined over u variable):

$$\begin{aligned} &P(\{z_i, A_i, \theta_i\}|V, y, \omega^g, \omega^A, \Omega) P(V|y,\omega^g) \\ &P(y|\Omega) P(\omega) P(\Omega), \end{aligned} \tag{15}$$

with

$$\begin{aligned} &P(\{z_i, A_i, \theta_i\}|V, y, \omega^g, \omega^A, \Omega) \\ &= \frac{1}{Z} \prod_{a \in L_O(y): i(a) \neq 0} P(A_{i(a)}|y, \omega^A, V) \\ &\prod_{c \in C(L_O(y))} P(\vec{l}_c(\{z_{i(a)}, \theta_{i(a)}\})|y, \omega^g, V). \end{aligned} \tag{16}$$

We have the constraint that $|L_B(y)| + \sum_{a \in L_O(y)}(1 - \delta_{i(a),0}) = N_\tau$. Hence, $P(y|\Omega)$ reduces to two components: 1) the probability of the aspect $P(L_O(y)|\Omega)$ and 2) the probability $\Omega^B e^{-\Omega^B |L_B(y)|}$ of having $|L_B(y)|$ background nodes.

There is one problem with the formulation of (16). There are variables on the right-hand side of the equation that are not observed—i.e., z_a, θ_a such that $i(a) = 0$. In principle, these variables should be removed from the equation by integrating them out. In practice, we replace their values by their best estimates from $P(\vec{l}_c(\{z_{i(a)}, \theta_{i(a)}\})|y, \omega^g)$ using our current assignments of the other variables. For example, suppose we have assigned two vertices of a triplet to two image AFs and decide to assign the third vertex to be unobserved. Then, we estimate the position and orientation of the third vertex by the most probable value given the position and orientation assignments of the first two vertices and relevant clique potential. This is suboptimal, but intuitive and efficient. (It does require that we have at least two vertices assigned in each triplet.)

6 Learning and Inference of the Model

In order to learn the models, we face three tasks: 1) structure learning, 2) parameter learning to estimate (Ω, ω), and 3) inference to estimate (y, V) (from a single image).

Inference requires estimating the parse tree y and the correspondences $V = \{i(a)\}$ from input x. The model parameters (Ω, ω) are fixed. This requires solving

$$\begin{aligned} (y^*, V^*) &= \arg\max_{y,V} P(y, V|x, \omega, \Omega) \\ &= \arg\max_{y,V} P(x, \omega, \Omega, y, V). \end{aligned} \tag{17}$$

As described in Section 6.1, we use DP to estimate y^*, V^* efficiently.

Parameter learning occurs when the structure of the model is known, but we have to estimate the parameters of the model. Formally, we specify a set W of parameters (ω, Ω), which we estimate by MAP. Hence, we estimate

$$\begin{aligned}(\omega^*, \Omega^*) &= \arg\max_{\omega,\Omega \in W} P(\omega, \Omega|x) \propto P(x|\omega, \Omega)P(\omega, \Omega) \\ &= \arg\max_{\omega,\Omega \in W} P(\omega, \Omega) \prod_{\tau\in\Lambda} \sum_{y_\tau, V_\tau} P(x_\tau, y_\tau, V_\tau|\omega, \Omega).\end{aligned} \tag{18}$$

This is performed by an EM algorithm, see Section 6.2, where the summation over the $\{V_\tau\}$ is performed by DP (the summation over the ys corresponds to summing over the different aspects of the object). The ω, Ω are calculated using sufficient statistics.

Structure Learning involves learning the model structure. Our strategy is to grow the structure of the PGMM by adding new aspect nodes or by adding new cliques to existing aspect nodes. We use clustering techniques to propose ways to grow the structure, see Section 6.3. For each proposed structure, we have a set of parameters W that extends the set of parameters of the previous structure. For each new structure, we evaluate the fit to the data by computing the *score*:

$$\text{score} = \max_{\omega,\Omega} P(\omega, \Omega) \prod_{\tau\in\Lambda} \sum_{y_\tau} \sum_{V_\tau} P(x_\tau, y_\tau, V_\tau|\omega, \Omega). \tag{19}$$

We then apply standard model selection by using the score to determine if we should accept the proposed structure or not. Evaluating the score requires summing over the different aspects and correspondence $\{V_\tau\}$ for all of the images. This is performed by using dynamic programming.

6.1 Dynamic Programming for the Max and Sum

Dynamic programming plays a core role for PGMMs. All three tasks—inference, parameter learning, and structure learning—require dynamic programming. First, inference uses dynamic programming via the max rule to calculate the most probable parse tree y^*, V^* for input x. Second, in parameter learning, the E step of the EM algorithm relies on dynamic programming to compute the sufficient statistics by the sum rule and take the expectations with respect to $\{y_\tau\}$, $\{V_\tau\}$. Third, structure learning summing over all configurations $\{y_\tau\}$, $\{V_\tau\}$ uses dynamic programming as well.

The structure of a PGMM is designed to ensure that dynamic programming is practical. Dynamic programming was first used to detect objects in images by Coughlan et al. [31]. In this paper, we use the ordered clique representation to use the configurations of triangles as the basic variables for dynamic programming similar to the junction tree algorithm [25].

We first describe the use of dynamic programming using the max rule for inference (i.e., determining the aspect and correspondence for a single image). Then, we will describe the modification to the sum rule used for parameter learning and structure pursuit.

To perform inference, we need to estimate the best aspect (object model) $L_O(y)$ and the best assignment V. We loop over all possible aspects and, for each aspect, we select the best assignment by dynamic programming (DP). For DP, we keep a table of the possible assignments, including the unobservable assignment. As mentioned above, we perform the suboptimal method of replacing missing values z_a, θ_a such that $i(a) = 0$ by their most probable estimates.

The conditional distribution is obtained from (4), (7), (13), and (14):

$$\begin{aligned}P(y, V, x|\omega, \Omega) = \frac{1}{Z}\exp\Bigg\{ &\sum_{a\in C(L_O(y))} \psi_a(\vec{l}_{i(a)}, \omega_a^g) \\ &+ \sum_{a\in L_O(y)} \{1 - \delta_{i(a),0}\}\phi_{i(a)} \\ &- \sum_{a\in L_O(y)} \log\{\lambda_\omega/(1-\lambda_\omega)\}\delta_{i(a),0} \\ &- \Omega^B(N_\tau - |L_O(y)|) + \sum_{j\in[1,\rho]} I(\beta_j, L_O(y)) \log \Omega_j^O \Bigg\},\end{aligned} \tag{20}$$

where $\phi_{i(a)} = -(1/2)(A_{i(a)} - \mu_a^A)^T(\Sigma_a^A)^{-1}(A_{i(a)} - \mu_a^A)$, $\vec{l}_{i(a)} = \vec{l}(z_{i(a)}, \theta_{i(a)}, z_{i(a+1)}, \theta_{i(a+1)}, z_{i(a+2)}, \theta_{i(a+2)})$ and $I(\beta_j, L_O(y))$ is an indicator that indicates whether the aspect j is active or not. $I(\beta_j, L_O(y))$ is equal to one if $\beta_j \in L_O(y)$, otherwise zero.

We can reexpress this as

$$\begin{aligned}P(y, V, x|\omega, \Omega) = \prod_{a=1}^{|L_O|-2} &\hat{\pi}_a[(z_{i(a)}, A_{i(a)}, \theta_{i(a)}), \\ &(z_{i(a+1)}, A_{i(a+1)}, \theta_{i(a+1)}), (z_{i(a+2)}, A_{i(a+2)}, \theta_{i(a+2)})],\end{aligned} \tag{21}$$

where the $\hat{\pi}_a[.]$ is determined by (20).

We maximize (20) with respect to y and V. The choice of y is the choice of aspect (because the background nodes are determined by the constraint that all AFs in the image are matched). For each aspect, we use DP to maximize over V. This can be done recursively by defining a function $h_a = h_a[(z_{i(a)}, A_{i(a)}, \theta_{i(a)}), (z_{i(a+1)}, A_{i(a+1)}, \theta_{i(a+1)})]$ by a forward pass:

$$h_{a+1} = \max_{i(a)} \hat{\pi}_a h_a. \tag{22}$$

The forward pass computes the maximum value of $P(y, V, x|\omega, \Omega)$. The backward pass of dynamic programming computes the most probable value V^*. The forward and backward passes are computed for all possible aspects of the model. As stated earlier in Section 5.6, we make an approximation by replacing the values $z_{i(a)}, \theta_{i(a)}$ of unobserved object leaf nodes (i.e., $i(a) = 0$) by their most probable values.

We perform the max rule, (22), for each possible topological structure y. In this paper, the number of topological structures is very small (i.e., less than 20) for each object category and, so, it is possible to enumerate them all. The computational complexity of the dynamic programming algorithm is $O(MN^K)$, where M is the number of cliques in the aspect model for the object, $K = 3$ is the size of the maximum clique, and N is the number of image features.

We will also use the dynamic programming algorithm (using the sum rule) to help perform parameter learning and structure learning. For parameter learning, we use the EM algorithm, see Section 6.2, which requires calculating sums over different correspondences and aspects. For structure learning, we need to calculate the score, see (19), which also requires summing over different correspondences and aspects. This requires replacing the $\max$ in (22) by $\sum$. If points are unobserved, then we restrict the sum over their positions for computational reasons (summing over the positions close to their most likely positions).

6.2 EM Algorithm for Parameter Learning

We perform EM to estimate the parameters ω, Ω from the set of images $\{x_\tau : \tau \in \Lambda\}$. The criterion is to find the ω, Ω which maximize

$$P(\omega, \Omega|\{x_\tau\}) = \sum_{\{y_\tau\},\{V_\tau\}} P(\omega, \Omega, \{y_\tau\}, \{V_\tau\}|\{x_\tau\}), \tag{23}$$

where

$$P(\omega, \Omega, \{y_\tau\}, \{V_\tau\}|\{x_\tau\}) = \frac{1}{Z} P(\omega, \Omega) \prod_{\tau\in\Lambda} P(y_\tau, V_\tau|x_\tau, \omega, \Omega). \tag{24}$$

This requires us to treat $\{y_\tau\}, \{V_\tau\}$ as missing variables that must be summed out during the EM algorithm. To do this, we use the EM algorithm using the formulation described in [32]. This involves defining a free energy $F[q, \omega, \Omega]$ by

$$F[q(.,.), \omega, \Omega] = \sum_{\{y_\tau\},\{V_\tau\}} q_{y_\tau,V_\tau} \log q_{y_\tau,V_\tau} - \sum_{\{y_\tau\},\{V_\tau\}} q_{y_\tau,V_\tau} \log \tilde{P}, \tag{25}$$

where $q_{y_\tau,V_\tau} = q(\{y_\tau\}, \{V_\tau\})$ is a normalized probability distribution and $\tilde{P} = P(\omega, \Omega, \{y_\tau\}, \{V_\tau\}|\{x_\tau\})$. It can be shown [32] that minimizing $F[q(.,.), \omega, \Omega]$ with respect to $q(.,.)$ and (ω, Ω) in alternation is equivalent to the standard EM algorithm. This gives the E step and the M step:

E step:

$$q^t_{y_\tau,V_\tau} = P\big(\{y_\tau\}, \{V_\tau\}|\{x_\tau\}, \omega^t, \Omega^t\big). \tag{26}$$

M step:

$$(\omega^{t+1}, \Omega^{t+1}) = \arg\min_{\omega,\Omega} - \sum_{\{y_\tau\},\{V_\tau\}} \Big\{q^t_{y_\tau,V_\tau} \log \tilde{P}\Big\}. \tag{27}$$

The distribution $q(\{y_\tau\}, \{V_\tau\}) = \prod_{\tau\in\Lambda} q_\tau(\{y_\tau\}, \{V_\tau\})$ because there is no dependence between the images. Hence, the E step reduces to

$$q^t_\tau(\{y_\tau\}, \{V_\tau\}) = P\big(\{y_\tau\}, \{V_\tau\}|\{x_\tau\}, \omega^t, \Omega^t\big), \tag{28}$$

which is the distribution of the aspects and the correspondences using the current estimates of the parameters ω^t, Ω^t.

The M step requires maximizing with respect to the parameters ω, Ω after summing over all possible configurations (aspects and correspondences). The summation can be performed using the sum version of dynamic programming, see (22). The maximization over parameters is straightforward because they are the coefficients of Gaussian distributions (mean and covariances) or exponential distributions. Hence, the maximization can be done analytically.

For example, consider a simple exponential distribution $P(h|\alpha) = \frac{1}{Z(\alpha)} \exp\{f(\alpha)\phi(h)\}$, where h is the observable, α is for the parameters, $f(.)$ and $\phi(.)$ are arbitrary functions, and $Z(\alpha)$ is the normalization term. Then, $\sum_h q(h) \log P(h|\alpha) = f(\alpha) \sum_h q(h)\phi(h) - \log Z(\alpha)$. Hence, we have

$$\frac{\partial \sum_h q(h) \log P(h|\alpha)}{\partial\alpha} = \frac{\partial f(\alpha)}{\partial\alpha} \sum_h q(h)\phi(h) - \frac{\partial \log Z(\alpha)}{\partial\alpha}. \tag{29}$$

If the distributions are of simple forms, like the Gaussians used in our models, then the derivatives of $f(\alpha)$ and $\log Z(\alpha)$ are straightforward to compute and the equation can be solved analytically. The solution is of the form

$$\mu(t) = \sum_h q^t(h)h, \quad \sigma^2(t) = \sum_h q^t(h)\{h - \mu(t)\}^2. \tag{30}$$

Finally, the EM algorithm is only guaranteed to converge to a local maxima of $P(\omega, \Omega|\{x_\tau\})$ and, so, a good choice of initial conditions is critical. The triplet vocabularies, described in Section 6.3.1, give a good initialization (so we do not need to use standard methods such as multiple initial starting points).

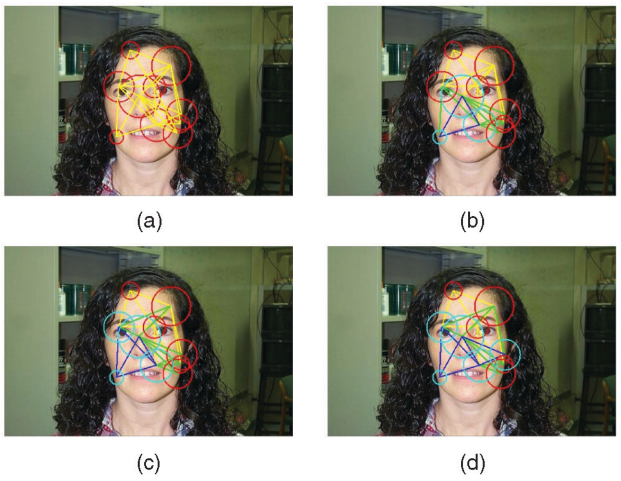

Fig. 8. This figure illustrates structure pursuit. (a) Image with triplets. (b) One triplet induced. (c) Two triplets induced. (d) Three triplets induced. Yellow triplets: all triplets from triplet vocabulary. Blue triplets: structure induced. Green triplets: possible extensions for next induction. Circles with radius: image features with different sizes.

6.3 Structure Pursuit

Structure pursuit proceeds by adding a new triplet clique to the PGMM. This is done either by adding a new aspect node O_j and/or by adding a new clique node $N_{a,a+1}$. This is illustrated in Fig. 6, where we grow the PGMM from panel 1 to panel 5 in a series of steps. For example, the steps from (1) to (2) and from (4) to (5) correspond to adding a new aspect node. The steps from (2) to (3) and from (3) to (4) correspond to adding new clique nodes. Adding new nodes requires adding new parameters to the model. Hence, it corresponds to expanding the set W of nonzero parameters.

Our strategy for structure pursuit is given as follows (see Figs. 8 and 9): We first use clustering algorithms to determine a triplet vocabulary. This triplet vocabulary is used to propose ways to grow the PGMM, which are evaluated by how well the modified PGMM fits the data. We select the PGMM with the best score, see (19). The use of these triplet vocabularies reduces the, potentially enormous, number of ways to expand the PGMM down to a practical number. We emphasize that the triplet vocabulary is only used to assist the structure learning, and it does not appear in the final PGMM.

6.3.1 *The Appearance and Triplet Vocabularies*

We construct appearance and triplet vocabularies using the features $\{x^\tau_i\}$ extracted from the image data set, as described in Section 3.1.

To get the appearance vocabulary Voc_A, we perform k-means clustering on the appearances $\{A^\tau_i\}$ (ignoring the

```
Input: Training Image $\tau = 1,..,M$ and the triplet vocabulary $Voc_2$. Initialize $G$ to be the root node with the background model, and let $G^* = G$.
Algorithm for Structure Induction:
  • STEP 1:
    – OR-NODE EXTENSION
      For $T \in Voc_2$
        ∗ $G' = G \bigcup T$ (OR-ing)
        ∗ Update parameters of $G'$ by EM algorithm
        ∗ If $Score(G') > Score(G^*)$ Then $G^* = G'$
    – AND-NODE EXTENSION
      For Image $\tau = 1,..,M$
        ∗ P = the highest probability parse for Image $\tau$ by $G$
        ∗ For each Triple $T$ in Image $\tau$
          if $T \bigcap P \neq \emptyset$
            · $G' = G \bigcup T$ (AND-ing)
            · Update parameters of $G'$ by EM algorithm
            · If $Score(G') > Score(G^*)$ Then $G^* = G'$
  • STEP 2: $G = G^*$. Go to STEP 1 until $Score(G) - Score(G^*) < Threshold$
Output: $G$
```

Fig. 9. Structure induction algorithm.

spatial positions and orientations $\{(z_i^\tau, \theta_i^\tau)\}$). The means, $\mu^{A,a}$, and covariances, $\Sigma^{A,a}$, of the clusters define the appearance vocabulary:

$$Voc_A = \{(\mu^{A,a}, \Sigma^{A,a}) : a \in \Lambda_A\}, \tag{31}$$

where Λ_A is a set of indexes for the appearance ($|\Lambda_A|$ is given by the number of means).

To get the triplet vocabulary, we first quantize the appearance data $\{A_i^\tau\}$ to the means $\mu^{A,a}$ of the appearance vocabulary using the nearest neighbor (with euclidean distance). This gives a set of modified data features $\{(z_i^\tau, \theta_i^\tau, \mu^{A,a(i,\tau)}\}$, where $a(i,\tau) = \arg\min_{a\in\Lambda_A} |A_i^\tau - \mu^{A,a}|$.

For each appearance triplet $(\mu^{A,a}, \mu^{A,b}, \mu^{A,c})$, we obtain the set of positions and orientations of the corresponding triplets of the modified data features:

$$\Big\{\left(z_i^\tau, \theta_i^\tau\right), \left(z_j^\tau, \theta_j^\tau\right), \left(z_k^\tau, \theta_k^\tau\right) : \\ \text{s.t.}\left(\mu^{A,a(i,\tau)}, \mu^{A,a(j,\tau)}, \mu^{A,a(k,\tau)}\right) = (\mu^{A,a}, \mu^{A,b}, \mu^{A,c})\Big\}. \tag{32}$$

We compute the ITV $\vec{l}$ of each triplet and perform k-means clustering to obtain a set of means $\mu_{abc}^{g,s}$ and covariances $\Sigma_{abc}^{g,s}$ for $s \in d_{abc}$, where $|d_{abc}|$ denotes the number of clusters. This gives the triplet vocabulary:

$$D = \{\mu_{abc}^{g,s}, \Sigma_{abc}^{g,s}, (\mu^{A,a}, \mu^{A,b}, \mu^{A,c}), (\Sigma^{A,a}, \Sigma^{A,b}, \Sigma^{A,c}) : \\ s \in d_{abc},\ a \leq b \leq c \quad a, b, c \in \Lambda_A\}. \tag{33}$$

The triplet vocabulary contains geometric and appearance information (both mean and covariance) about the triplets that commonly occur in the images. This triplet vocabulary will be used to make proposals to grow the structure of the model (including giving initial conditions for learning the model parameters by the EM algorithm).

6.3.2 Structure Induction Algorithm

We now have the necessary background to describe our structure induction algorithm. The full procedure is described in the pseudocode in Fig. 9. Fig. 6 shows an example of the structure being induced sequentially.

Initially, we assume that all of the data is generated by the background model. In the terminology of Section 6, this is equivalent to setting all of the model parameters Ω to be zero (except those for the background model). We can estimate the parameters of this model and score the model, as described in Section 6.

Next, we seek to expand the structure of this model. To do this, we use the triplet vocabularies to make proposals. Since the current model is the background model, the only structure change allowed is to add a triplet model as one child of the category node O (i.e., to create the background plus triple model described in the previous section, see Fig. 6). We consider all members of the triplet vocabulary as candidates, using their cluster means and covariances as initial setting on their geometry and appearance properties in the EM algorithm, as described in Section 6.2. Then, for all of these triples, we construct the background plus triplet model, estimate their parameters, and score them. We accept the one with the highest score as the new structure.

As the graph structure grows, we now have more ways to expand the graph. We can add a new triplet as a child of the category node. This proceeds as in the previous paragraph or we can take two members of an existing triplet and use them to construct a new triplet. In this case, we first parse the data using the current model. Then, we use the triplet vocabulary to propose possible triplets, which partially overlap with the current model (and give them initial settings on their parameters as before), see Fig. 8. Then, for all possible extensions, we use the methods in Section 6 to score the models. We select the one with the highest score as the new graph model. If the score increase is not sufficient, we cease building the graph model. See the structured models in Fig. 11.

7 Experimental Results

Our experiments were designed to give proof of concept for the PGMM. First, we show that our approach gives comparable results to other approaches for classification (testing between images containing the object versus purely background images) when tested on the Caltech 4 (faces, motorbikes, airplanes, and background) [11] and Caltech 101 images [12] (note that most of these approaches are weakly supervised and so are given more information than our unsupervised method). Moreover, our approach can perform additional tasks such as localization (which are impossible for some methods like bag of key points [16]). Our inference algorithm is fast and takes under five seconds (the CPU is an AMD Opteron processor 880, 2.4 GHz). Second, we illustrate a key advantage of our method, that it can both learn and perform inference when the 2D pose (position, orientation, and scale) of the object varies. We check this by creating a new data set by varying the pose of objects in Caltech 101. Third, we illustrate the advantages of having a variable graph structure (i.e., OR nodes) in several ways. We first quantify how the performance of the model improves as we allow the number of OR nodes to increase. Next, we show that learning is possible even when the

TABLE 2
We Have Learned Probability Grammars for 13 Objects in the Caltech Database, Obtaining Scores over 90 percent for Most Objects

Dataset	Size	Ours	Constellation Model
Faces	435	97.7	96.4
Motorbikes	800	92.9	92.5
Airplanes	800	91.8	90.2
Chair	62	90.9	–
Cougar Face	69	90.9	–
Grand Piano	90	96.3	–
Panda	38	90.9	–
Rooster	49	92.1	–
Scissors	39	94.9	–
Stapler	45	90.5	–
Wheelchair	59	93.6	–
Windsor Chair	56	92.4	–
Wrench	39	84.6	–

A score of 90 percent, means that we have a classification rate of 90 percent and a false positive rate of 10 percent $(10\% = (100 - 90)\%)$. *We compare our results with the constellation model.*

training data set consists of a random mixture of images containing the objects and images that do not (and, hence, are pure background). Finally, we learn a *hybrid model*, where we are given training examples that contain one of several different types of object and learn a model that has different OR nodes for different objects.

7.1 Learning Individual Objects Models

In this section, we demonstrate the performance of our models for objects chosen from the Caltech data sets. We first choose a set of 13 object categories (as reported in [22]). Three classes of faces, motorbikes, and airplanes come from that in [11]. We use the identical splitting for training and testing as used in [11]. The remaining categories are selected from the Caltech 101 data set [12]. To avoid concerns about selection bias and to extend the number of object categories, we perform additional experiments on all object categories from that in [12] for which there are at least 80 images (80 is a cutoff factor chosen to ensure that there is a sufficient amount of data for training and testing). This gives an additional set of 26 categories (the same parameter settings

TABLE 3
Localization Rate Is Used to Measure the Proportion of the AFs of the Model that Lie within the Ground-Truth Bounding Box

Dataset	Localization Rate
Faces	96.3
Motorbikes	98.6
Airplanes	91.5

were used on both sets). Each data set was randomly split into two sets with equal size (one for training and the other for testing). Note that, in the Caltech data sets, the objects typically appear in standardized orientations. Hence, rotation invariance is not necessary. To check this, we also implemented a simpler version of our model that was not rotation invariant by modifying the $\vec{l}$ vector, as described in Section 3.2. The results of this simplified model were practically identical to the results of the full model, which we now present.

K-means clustering was used to learn the appearance and triplet vocabularies, where, typically, K is set to 150 and 1,000, respectively. Each row in Fig. 5 corresponds to some triplets in the same group.

We illustrate the results of the PGMMs in Table 2 and Fig. 10. A score of 90 percent means that we get a true positive rate of 90 percent and a false positive rate of 10 percent. This is for classifying between images containing the object and purely background images [11]. For comparison, we show the performance of the Constellation Model [11]. These results are slightly inferior to the bag of keypoint methods [16] (which requires weak supervision). We also evaluate the ability of the PGMMs to localize the object. To do this, we compute the proportion of the AFs of the model that lie within the ground-truth bounding box. Our localization results are shown in Table 3. Note that some alternative methods, such as the bag of keypoints, are unable to perform localization.

The models for individual objects classes, learned from the proposed algorithm, are illustrated in Fig. 11. Observe that the generative models have different tree-width and depth. Each subtree of the object node defines an MRF to describe one aspect of the object. The computational cost of the inference using DP is proportional to the height of the subtree and exponential to the maximum width (only three

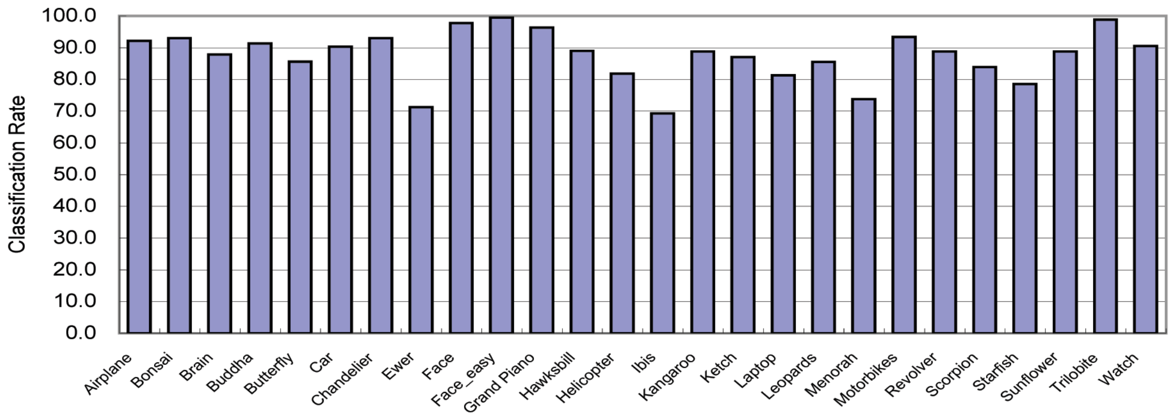

Fig. 10. We report the classification performance for 26 classes that have at least 80 images. The average classification rate is 87.6 percent.

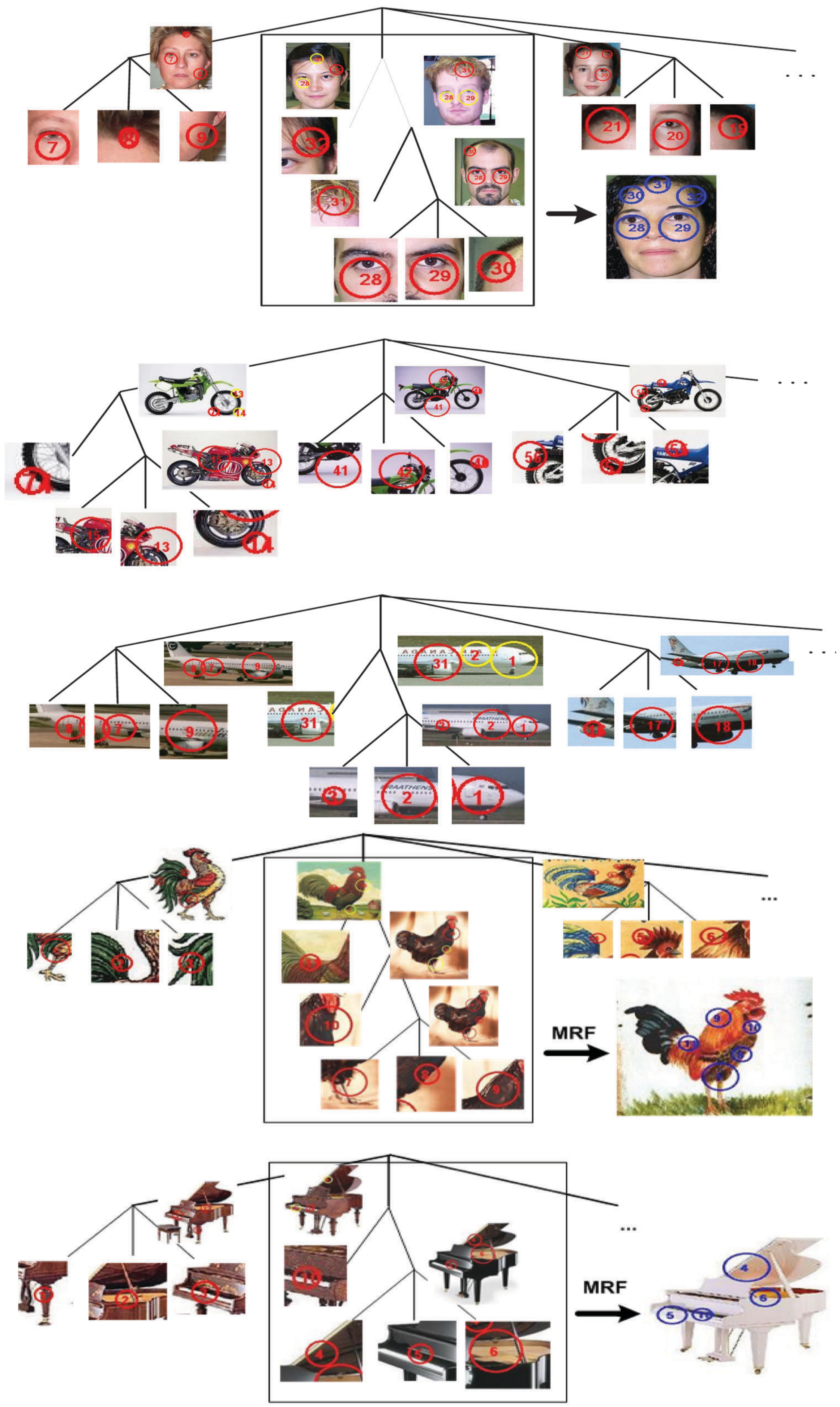

Fig. 11. Individual models learned for faces, motorbikes, airplanes, grand piano, and rooster. The circles represent the AFs. The numbers inside the circles give the a index of the nodes, see Table 1. The Markov Random Fields of one aspect each of faces, roosters, and grand pianos are shown on the right.

in our case). The detection time is less than five seconds (including the processing of features and inference) for the image with the size of $320 * 240$. The training time is around two hours for 250 training images. The parsed results are illustrated in Fig. 12.

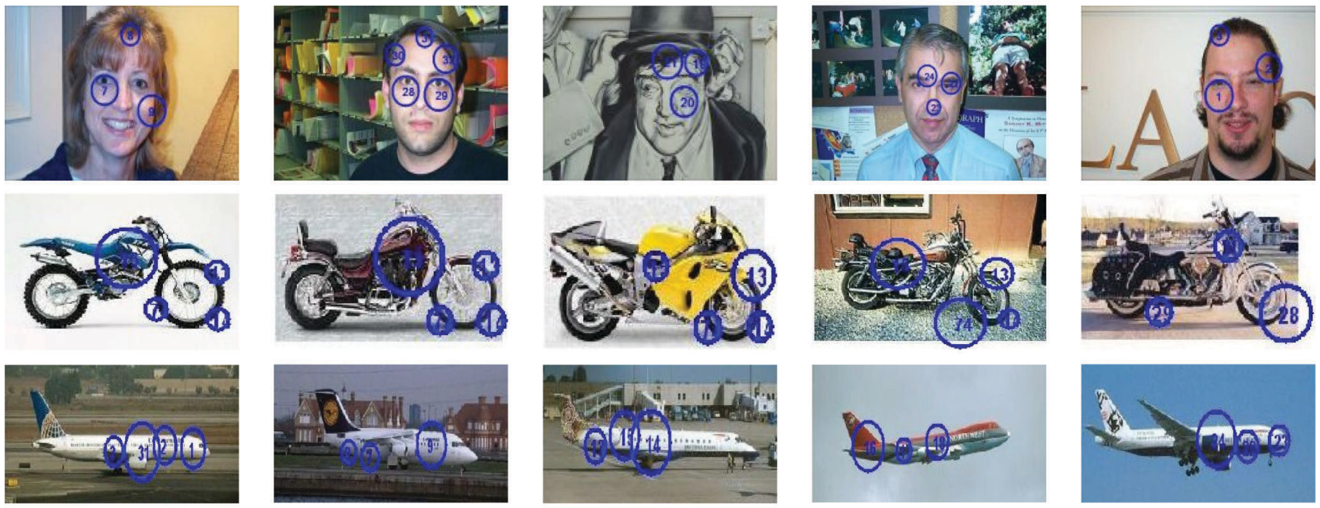

Fig. 12. Parsed results for faces, motorbikes, and airplanes. The circles represent the AFs. The numbers inside the circles give the a index of the nodes, see Table 1.

7.2 Invariance to Rotation and Scale

This section shows that the learning and inference of a PGMM is independent of the pose (position, orientation, and scale) of the object in the image. This is a key advantage of our approach and is due to the triplet representation.

To evaluate PGMMs for this task, we modify the Caltech 101 data set by varying either the orientation or the combination of orientation and scale. We performed learning and inference using images with 360-degree in-plane rotation and another data set with rotation and scaling together (where the scaling range is from 60 percent of the original size to 150 percent, i.e., $180 * 120 - 450 * 300$).

The PGMM showed only slight degradation due to these pose variations. Table 4 shows the comparison results. The parsing results (rotation + scale) are illustrated in Fig. 13.

7.3 The Advantages of Variable Graph Structure

Our basic results for classification and localization, see Section 7.1, showed that our PGMMs did learn variable graph structure (i.e., OR nodes). We now explore the benefits of this ability.

First, we can quantify the use of the OR nodes for the basic tasks of classification. We measure how performance degrades as we restrict the number of OR nodes, see Fig. 14. This shows that performance increases as the number of OR nodes gets bigger, but this increase is jagged and soon reaches an asymptote.

Second, we show that we can learn a PGMM even when the training data set consists of a random mixture of images containing the object and images that do not. Table 5 shows the results. The PGMM can learn in these conditions because it uses some OR nodes to learn the object (i.e., account for the images that contain the object) and other OR nodes to deal with the remaining images. The overall performance of this PGMM is only slightly worse that the PGMM trained on standard images (see Section 7.1).

Third, we show that we can learn a model for an object class. We use a hybrid class that consists of faces, airplanes, and motorbikes. In other words, we know that one object is present in each image, but we do not know which. In the training stage, we randomly select images from the data sets of faces, airplanes, and motorbikes. Similarly, we test the hybrid model on examples selected randomly from these three data sets.

The learned hybrid model is illustrated in Fig. 15. It breaks down nicely into the ORs of the models for each object. Table 6 shows the performance for the hybrid model. This demonstrates that the proposed method can learn a model for the class with extremely large variation.

8 Discussion

This paper introduced PGMMs and showed that they can be learned in an unsupervised manner and perform tasks such as the classification and localization of objects in unknown backgrounds. We also showed that PGMMs were invariant to 2D pose (position, scale, and rotation) for both learning and inference. PGMMs could also deal with different appearances or aspects of the object and also learn hybrid models that include several different types of object.

More technically, PGMMs combine elements of probabilistic grammars and MRFs. The grammar component enables them to adapt to different aspects, while the MRF

TABLE 4
Invariant to Rotation and Scale

Method	Accuracy
Scale Normalized	97.8
Rotation Only	96.3
Rotation + Scale	96.3

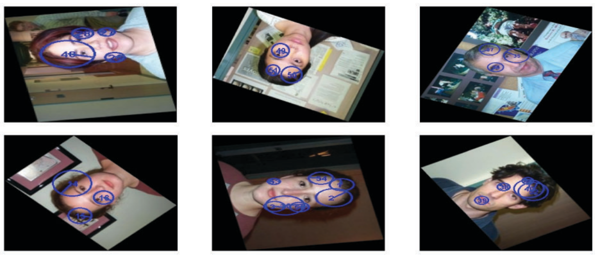

Fig. 13. Parsed results: invariant to rotation and scale.

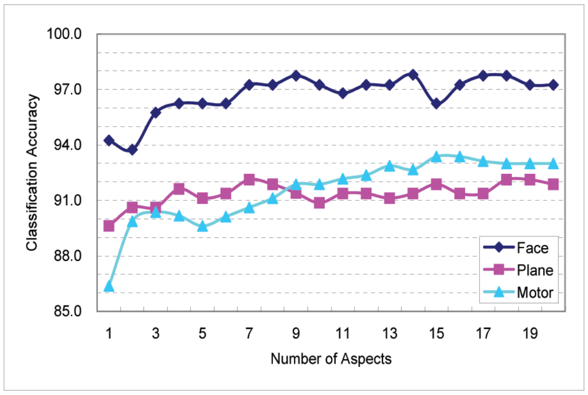

Fig. 14. Analysis of the effects of adding OR nodes. Observe that performance rapidly improves compared to the single MRF model with only one aspect as we add extra aspects. However, this improvement reaches an asymptote fairly quickly. (This type of result is obviously data set dependent.)

enables them to model spatial relations. The nature of PGMMs enables rapid inference and parameter learning by exploiting the topological structure of the PGMM, which enables the use of DP. The nature of PGMMs also enables us to perform structure induction to learn the structure of the model, in this case, by using oriented triplets as elementary building blocks that can be composed to form bigger structures.

Our experiments demonstrated proof of concept of our approach. We showed that 1) we can learn probabilistic models for a variety of different objects and perform rapid inference (less than five seconds), 2) our learning and inference is invariant to scale and rotation, and 3) we can learn models in noisy data for hybrid classes and the use of different aspects improves performance.

PGMMs are the first step in our program for unsupervised learning of object models. Our next steps will be to

TABLE 6
The PGMM Can Learn a Hybrid Class which Consists of Faces, Airplanes, and Motorbikes

Dataset	Single Model	Hybrid Model
Faces	97.8	84.0
Motorbikes	93.4	82.7
Airplanes	92.1	87.3
Overall	–	84.7

TABLE 5
The PGMM Are Learned on Different Training Data Sets which Consist of a Random Mixture of Images Containing the Object and Images which Do Not

	Training Set		Testing Set		
Dataset	Object Images	Background Images	Object Images	Background Images	Classification Rate
Faces	200	0	200	200	97.8
Faces	200	50	200	200	98.3
Faces	200	100	200	200	97.7
Motor	399	0	399	200	93.7
Motor	399	50	399	200	93.2
Motor	399	100	399	200	93.0
Plane	400	0	400	200	92.1
Plane	400	50	400	200	90.5
Plane	400	100	400	200	90.2

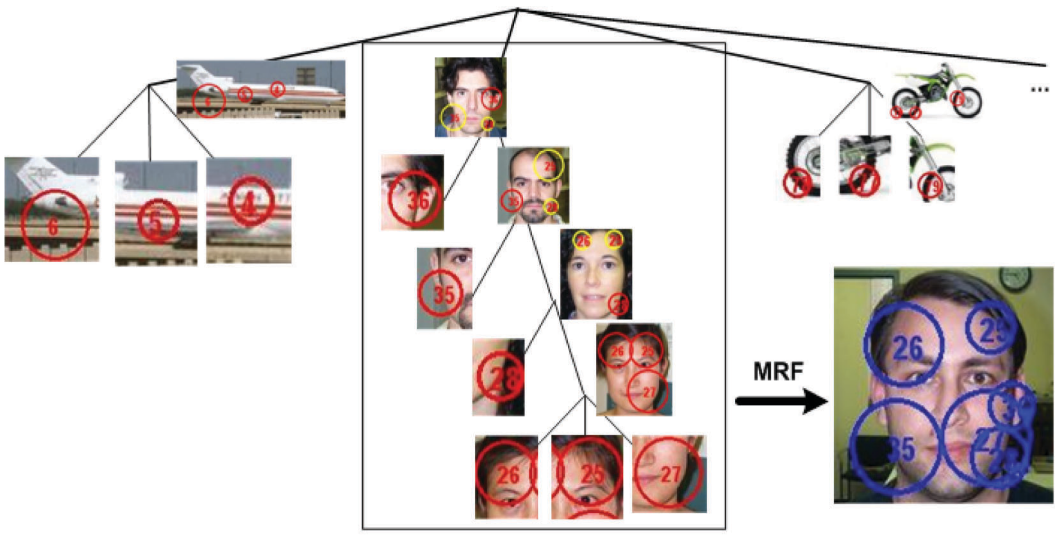

Fig. 15. Hybrid model learned for faces, motorbikes and airplanes.

extend this approach by allowing a more sophisticated representation and using a richer set of image features.

Acknowledgments

The authors gratefully acknowledge the support from the W.M. Keck Foundation, the US National Science Foundation with NSF Grant 0413214, and the US National Institutes of Health Grant RO1 EY015261. They thank Ying-Nian Wu and Song-Chun Zhu for the stimulating discussions. They also thank SongFeng Zheng and Shuang Wu for the helpful feedback on the drafts of this paper.

References

[1] D. McAllester, M. Collins, and F. Pereira, “Case-Factor Diagrams for Structured Probabilistic Modeling,” *Proc. 20th Conf. Uncertainty in Artificial Intelligence,* pp. 382-391, 2004.

[2] J. Lafferty, A. McCallum, and F. Pereira, “Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data,” *Proc. 18th Int'l Conf. Machine Learning,* pp. 282-289, 2001.

[3] D. Klein and C. Manning, “Natural Language Grammar Induction Using a Constituent-Context Model,” *Advances in Neural Information Processing Systems 14,* S.B.T.G. Dietterich and Z. Ghahramani, eds., MIT Press, 2002.

[4] H. Dechter and R. Mateescu, “AND/OR Search Spaces for Graphical Models,” *Artificial Intelligence,* 2006.

[5] H. Chen, Z.J. Xu, Z.Q. Liu, and S.C. Zhu, “Composite Templates for Cloth Modeling and Sketching,” *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 943-950, 2006.

[6] L.S. Zettlemoyer and M. Collins, “Learning to Map Sentences to Logical Form: Structured Classification with Probabilistic Categorial Grammars,” *Proc. 21st Ann. Conf. Uncertainty in Artificial Intelligence,* pp. 658-666, 2005.

[7] B. Ripley, *Pattern Recognition and Neural Networks.* Cambridge Univ. Press, 1996.

[8] C. Manning and C. Schütze, *Foundations of Statistical Natural Language Processing.* MIT Press, 1999.

[9] Z. Tu, X. Chen, A. Yuille, and S.-C. Zhu, “Image Parsing: Unifying Segmentation, Detection, and Recognition,” *Int'l J. Computer Vision,* vol. 63, pp. 113-140, 2005.

[10] H. Barlow, “Unsupervised Learning,” *Neural Computation,* vol. 1, pp. 295-311, 1989.

[11] R. Fergus, P. Perona, and A. Zisserman, “Object Class Recognition by Unsupervised Scale-Invariant Learning,” *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* 2003.

[12] L. Fei-Fei, R. Fergus, and P. Perona, “Learning Generative Visual Models from Few Training Examples: An Incremental Bayesian Approach Tested on 101 Object Categories,” *Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition Workshop Generative-Model Based Vision,* 2004.

[13] R. Fergus, P. Perona, and A. Zisserman, “A Sparse Object Category Model for Efficient Learning and Exhaustive Recognition,” *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* vol. 1, pp. 380-397, 2005.

[14] D.J. Crandall and D.P. Huttenlocher, “Weakly Supervised Learning of Part-Based Spatial Models for Visual Object Recognition,” *Proc. European Conf. Computer Vision,* vol. 1, pp. 16-29, 2006.

[15] D. Crandall, P. Felzenszwalb, and D. Huttenlocher, “Spatial Priors for Part-Based Recognition Using Statistical Models,” *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* vol. 1, pp. 10-17, 2005.

[16] G. Csurka, C. Bray, C. Dance, and L. Fan, “Visual Categorization with Bags of Keypoints,” *Proc. European Conf. Computer Vision Workshop Statistical Learning in Computer Vision,* 2004.

[17] M. Meila and M.I. Jordan, “Learning with Mixtures of Trees,” *J. Machine Learning Research,* vol. 1, pp. 1-48, 2000.

[18] S.D. Pietra, V.J.D. Pietra, and J.D. Lafferty, “Inducing Features of Random Fields,” *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 19, no. 4, pp. 380-393, Apr. 1997.

[19] S. Zhu, Y. Wu, and D. Mumford, “Minimax Entropy Principle and Its Application to Texture Modeling,” *Neural Computation,* vol. 9, no. 8, Nov. 1997.

[20] A. McCallum, “Efficiently Inducing Features of Conditional Random Fields,” *Proc. 19th Conf. Uncertainty in Artificial Intelligence,* 2003.

[21] N. Friedman, “The Bayesian Structural EM Algorithm,” *Proc. 14th Ann. Conf. Uncertainty in Artificial Intelligence,* pp. 13-129, 1998.

[22] L. Zhu, Y. Chen, and A. Yuille, “Unsupervised Learning of a Probabilistic Grammar for Object Detection and Parsing,” *Advances in Neural Information Processing Systems 19,* B. Schölkopf, J. Platt, and T. Hoffman, eds., MIT Press, 2007.

[23] L. Shams and C. von der Malsburg, “Are Object Shape Primitives Learnable?” *Neurocomputing,* vols. 26-27, pp. 855-863, 1999.

[24] J. Ponce, T.L. Berg, M. Everingham, D.A. Forsyth, M. Hebert, S. Lazebnik, M. Marszalek, C. Schmid, amdA. Torralba, C.K.I. Williams, J. Zhang, and A. Zisserman, *Dataset Issues in Object Recognition, Toward Category-Level Object Recognition,* J. Ponce, M. Hebert, C. Schmid, and A. Zisserman, eds., 2006.

[25] F.V. Jensen, S.L. Lauritzen, and K.G. Olesen, “Bayesian Updating in Causal Probabilistic Networks by Local Computations,” *Computational Statistics Quarterly,* vol. 4, pp. 269-282, 1990.

[26] T. Kadir and M. Brady, “Saliency, Scale and Image Description,” *Int'l J. Computer Vision,* vol. 45, no. 2, pp. 83-105, 2001.

[27] D.G. Lowe, “Distinctive Image Features from Scale-Invariant Keypoints,” *Int'l J. Computer Vision,* vol. 60, no. 2, pp. 91-110, 2004.

[28] Y. Amit and D. Geman, “A Computational Model for Visual Selection,” *Neural Computation,* vol. 11, no. 7, pp. 1691-1715, 1999.

[29] S. Lazebnik, C. Schmid, and J. Ponce, “Semi-Local Affine Parts for Object Recognition,” *Proc. British Machine Vision Conf.,* 2004.

[30] Y. Jin and S. Geman, “Context and Hierarchy in a Probabilistic Image Model,” *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 2145-2152, 2006.

[31] J. Coughlan, D. Snow, C. English, and A. Yuille, “Efficient Deformable Template Detection and Localization without User Initialization,” *Computer Vision and Image Understanding,* vol. 78, pp. 303-319, 2000.

[32] R. Neal and G.E. Hinton, *A View of the EM Algorithm that Justifies Incremental, Sparse, and Other Variants.* MIT Press, 1998.

Long Zhu received the BS degree in computer science from Northeastern University, China, in 2001 and the PhD degree in statistics at the University of California, Los Angeles in 2008. His research interests include computer vision and machine learning.

Yuanhao Chen received the BS degree from the Department of Automation, University of Science and Technology of China, in 2003. He is currently a PhD candidate at the University of Science and Technology of China. He is also a student in the joint PhD program of Microsoft Research Asia and the University of Science and Technology of China. His research interests include computer vision, machine learning, and information retrieval.

Alan Yuille received the BA degree in mathematics from the University of Cambridge in 1976. His PhD on theoretical physics, supervised by Prof. S.W. Hawking, was approved in 1981. He was a research scientist in the Artificial Intelligence Laboratory at MIT and the Division of Applied Sciences at Harvard University from 1982 to 1988. He served as an assistant and associate professor at Harvard until 1996. He was a senior research scientist at the Smith-Kettlewell Eye Research Institute from 1996 to 2002. He joined the University of California, Los Angeles, as a full professor with a joint appointment in statistics and psychology in 2002. He obtained a joint appointment in computer science in 2007. His research interests include computational models of vision, mathematical models of cognition, and artificial intelligence and neural networks.