
Unsupervised Learning of Probabilistic Grammar-Markov Models for Object Categories

Long (Leo) Zhu¹, Yuanhao Chen², Alan Yuille¹

¹Department of Statistics, University of California at Los Angeles, Los Angeles, CA 90095
{lzh, yuille}@stat.ucla.edu

²University of Science and Technology of China, Hefei, Anhui 230026 P.R.China
yhchen4@ustc.edu

Revised version. Resubmitted 10/Oct/2007 to IEEE Transactions on Pattern Analysis and Machine Intelligence.

Unsupervised Learning of Probabilistic Grammar-Markov Models for Object Categories

Long (Leo) Zhu¹, Yuanhao Chen², Alan Yuille¹

¹Department of Statistics, University of California at Los Angeles, Los Angeles, CA 90095
{lzh,yuille}@stat.ucla.edu

²University of Science and Technology of China, Hefei, Anhui 230026 P.R.China
yhchen4@ustc.edu

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 pose (position, orientation, and scale) in both inference and learning, different appearances, or aspects, of the model. They are learnt in an unsupervised manner where the input is a set of images with the object somewhere in the image (at different poses) and with variable background. We also extend this to learning PGMMs for classes of objects (where an unknown object from the class is present in each image). 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 101 dataset (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

I. INTRODUCTION

Remarkable progress in 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]–[6]. Unlike more traditional models, such as Markov Random Fields (MRF’s) [7] and Conditional Random Fields (CRF’s) [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 longterm goal is to provide a theoretical framework for the unsupervised learning of probabilistic models for generating, and interpreting (or parsing), 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. Firstly, images are arguably far more complex than sentences so learning a probabilistic model to generate natural images is too ambitious to start with. Secondly, even if we restrict ourselves to the simpler task of generating an image containing a single object we must deal with: (i) the cluttered background (similar to learning a natural language grammar when the input contains random symbols as well as words), (ii) the unknown pose (size, scale, and position) of the object, and (iii) the different appearances, or *aspects*, of the object. Thirdly, 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 the same object but with unknown pose (position, scale, and orientation). The object is allowed to have several different appearances, or aspects. We represent these images in terms of attributed features (AF’s). The task is to learn a probabilistic model for generating the AF’s (both those of the object and the background). Our learning is unsupervised in the sense that we know that

there is an object present in the image but we do not know its pose and, in some cases, we do not know the identity of the object. We require that the probability model must allow: (i) rapid inference, (ii) rapid parameter learning, and (iii) *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) which is motivated by this goal and its requirements. The PGMM combines elements of MRF's [7] and probabilistic context free grammars (PCFG's) [8]. The requirement that we can deal with a variable number of AF's (e.g. caused by different aspects of the object) motivates the use of grammars (instead of fixed graph models like MRF's). But PCFG's, see figure (1), are inappropriate because they make independence assumptions on the production rules and hence must be supplemented by MRF's to model the spatial relationships between AF's of the object. The requirement that we deal with pose (both for learning and inference) motivates the use of oriented triangles of AF's as our basic building blocks for the probabilistic model, see figure (2). These oriented triangles are represented by features, such as the internal angles of the triangle, which are invariant to the pose of the object. The requirement that we can perform rapid inference on new images is achieved by combining the triangle building blocks to enable dynamic programming. The ability to perform rapid inference ensures that parameter estimation and structure learning is practical.

We decompose the learning task into: (a) learning the structure of the model, and (b) 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 which 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 101 database [11]. Performance on this database has been much studied [11]–[15]. But we stress that the goal of our paper is

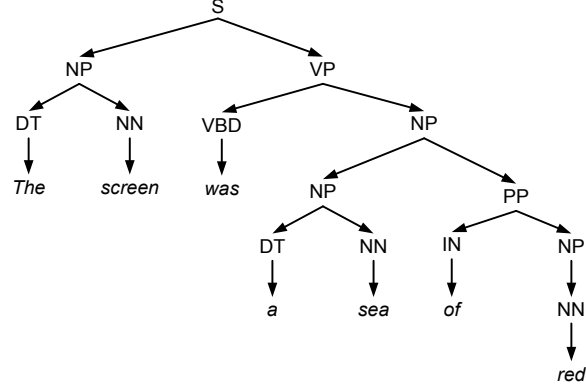


Fig. 1. Probabilistic Context Free Grammar. The grammar applies production rules with probability to generate a tree structure. Different random sampling will generate different tree structures. The production rules are applied independently on different branches of the tree. There are no sideways relations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

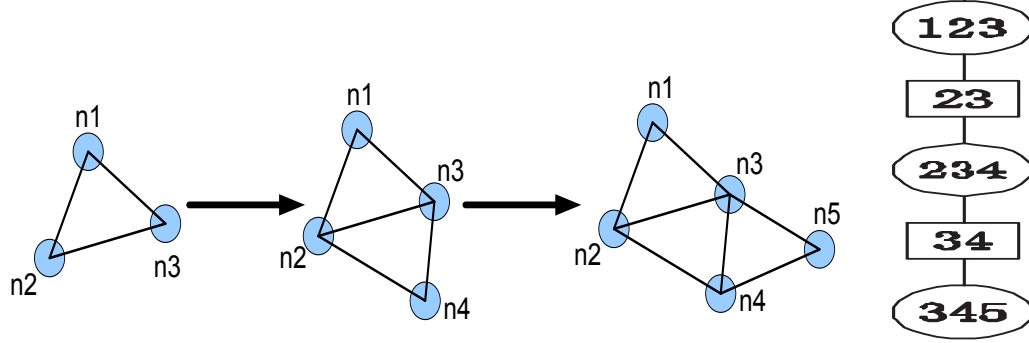


Fig. 2. This paper uses triplets of nodes as building blocks. We can grow the structure by adding new triangles. The junction tree (the far right panel) is used to represent the combination of triplets to allow efficient inference.

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. Firstly, 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. Secondly, 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, lacking a standard database where these variations occur naturally). Thirdly, the approach is able to learn from noisy data

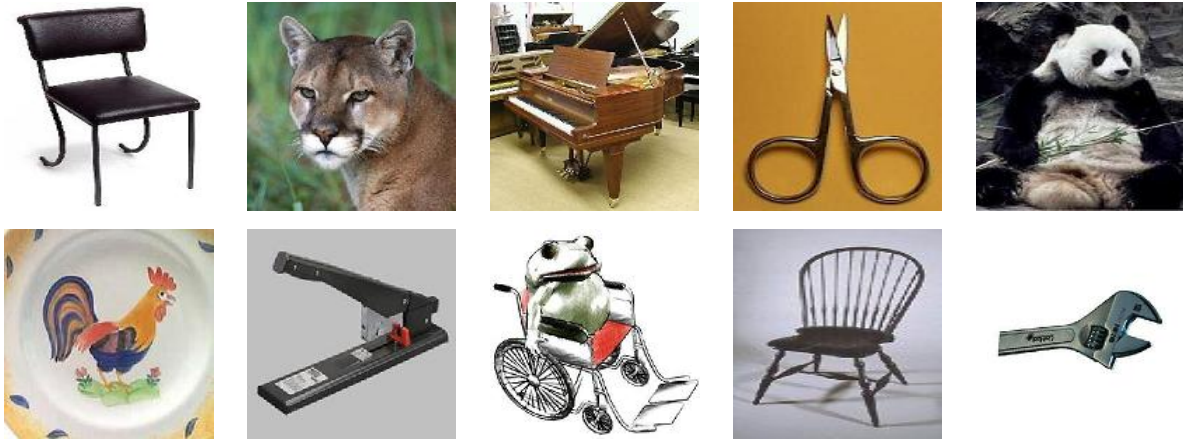


Fig. 3. Ten of the object categories from Caltech 101 which we learn in this paper.

(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 (II). Section (III) describes the features we use to represent the images. In section (IV) we give an overview of PGMMs. Section (V) specifies the probability distributions defined over the PGMM. In section (VI), we describe the algorithms for inference, parameter learning, and structure learning. Section (VII) illustrates our approach by learning models for 13 object, demonstrating invariance to scale and rotation, and performing learning for object classes.

II. 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 MRF's [7] or Conditional Random Fields [2], or Probabilistic Context Free Grammars (PCFG's) [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 MRF's and PCFG's. In this paper, we will be concerned with models that combine

probabilistic grammars with MRF's. The grammars are based on AND-OR graphs [1], [4], [5], which relate to mixtures of trees [16]. This merging of MRF's with probabilistic grammars results in structured models which have the advantages of variable graph structure (e.g. from PCFG's) combined with the rich spatial structure from the MRF's.

There has been considerable interest in inference algorithms for these structured models, for example McAllester *et al* [1] describe how dynamic programming algorithms (e.g. Viterbi and inside-outside) can be used to rapidly compute properties of interest for Case-Factor diagrams. But inference on arbitrary models combining PCFG's and MRF's 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 [17] which is also known as feature pursuit [18]. A similar strategy is also used to learn CRF's [19] where the learning is fully supervised. For Bayesian network, there is work on learning the structure using the EM algorithm [20].

Learning the structure of grammars in an unsupervised way is more difficult. Klein and Manning [3] have developed unsupervised learning of PCFG's 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. But an extremely compressed version of part of our work appeared in NIPS [21].

There has been a considerable amount of work for learning MRF models for visual tasks such as object detection. An early attempt was described in [22]. The constellation model [11] is a nice example of an unsupervised algorithm which represents objects by a fully connected (fixed) graph. Huttenlocher and collaborators al. [13], [14] explore different simpler MRF structures, such as k-fans models, which enable rapid inference.

There is also a large literature [11]–[15] on computer vision models for performing object recognition which have been evaluated on the Caltech databases [23]. Several of these models

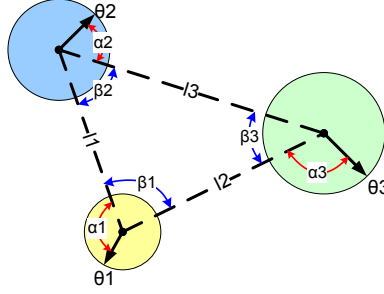


Fig. 4. The oriented triplet is specified by the internal angles β , the orientation of the vertices θ , and the relative angles α between them.

are learnt but suffer from over-generalization – for example, performance degrades significantly if the objects are kept fixed but the background changes [24].

III. THE IMAGE REPRESENTATION: FEATURES AND ORIENTED TRIPLETS

In this paper we will represent images in terms of isolated attributed features, which will be described in section (III-A). 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 figures (4,5). The advantages of using conjunctions of basic features is well-known in natural language processing and leads to unigram, bigram, and trigram features [8].

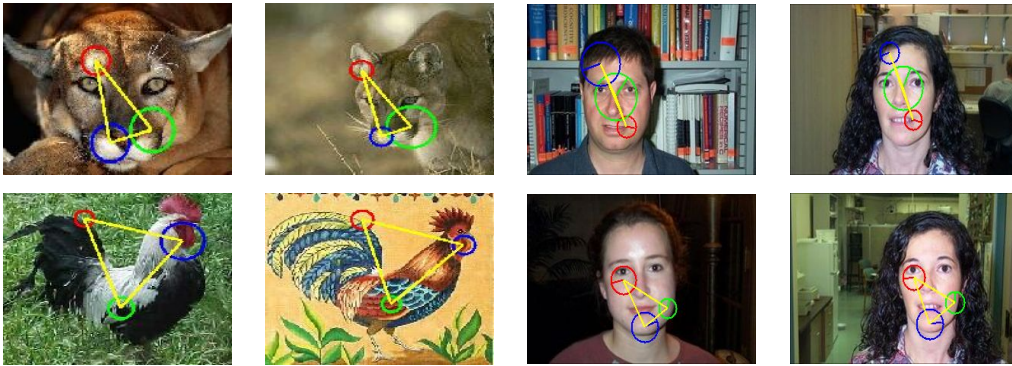


Fig. 5. This figure illustrates the features and triplets without orientation (left two panels) and oriented triplets (next two panels).

There are several reasons for using oriented triplets in this paper. Firstly, they contain geometrical properties which 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 figures (4,5). These properties can be used both for learning and inference of a PGMM when the scale and rotation are unknown. Secondly, they lead to a representation which is well suited to dynamic programming, similar to the junction tree algorithm [25], which enables rapid inference, see figures (6,2). Thirdly, 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 figures (2,6).

A. The Image Features

We represent an image by attributed features $\{x_i : i = 1, \dots, N_\tau\}$, where N_τ is the number of features in image \mathbf{I}_τ 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}_τ such that $K_b(C^i(\mathbf{I}_\tau)) > T, \forall i$, where T is a fixed threshold. We scale these regions to a constant size to obtain a set of scaled regions $\{\hat{C}^i(\mathbf{I}_\tau) : i = 1, \dots, N_\tau\}$. Then we apply the SIFT operator $L(\cdot)$ [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 as follows. Firstly, the Kadir-Brady operator is an *interest operator* which selects the parts of the image which contain interesting features (e.g. edges, triple points, and textured structures). Secondly, the Kadir-Brady operator adapts geometrically to the size of the feature, and hence is scale-invariant. Thirdly, 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.

B. The Oriented Triplets

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

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

where l_1, l_2, l_3 are the length of the three edges, $L = l_1 + l_2 + l_3$, $\alpha_1, \alpha_2, \alpha_3$ are the relative angles between the orientations $\theta_i, \theta_j, \theta_k$ and the orientations of the three edges of the triangle, and $\beta_1, \beta_2, \beta_3$ are the angles between edges of the triangle (hence $\beta_1 + \beta_2 + \beta_3 = \pi$).

This representation is over-complete. But 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.

IV. PROBABILISTIC GRAMMAR-MARKOV MODEL

We now give an overview of the Probabilistic Grammar-Markov Model (PGMM), which has characteristics of both a probabilistic grammar, such as a Probabilistic Context Free Grammar (PCFG), and a Markov Random Field (MRF). The probabilistic grammar component of the PGMM specifies different topological structures, as illustrated in the five leftmost panels of figure (6), enabling the ability to deal with variable number of attributed features. The MRF component specifies spatial relationships and is indicated by the horizontal connections.

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,

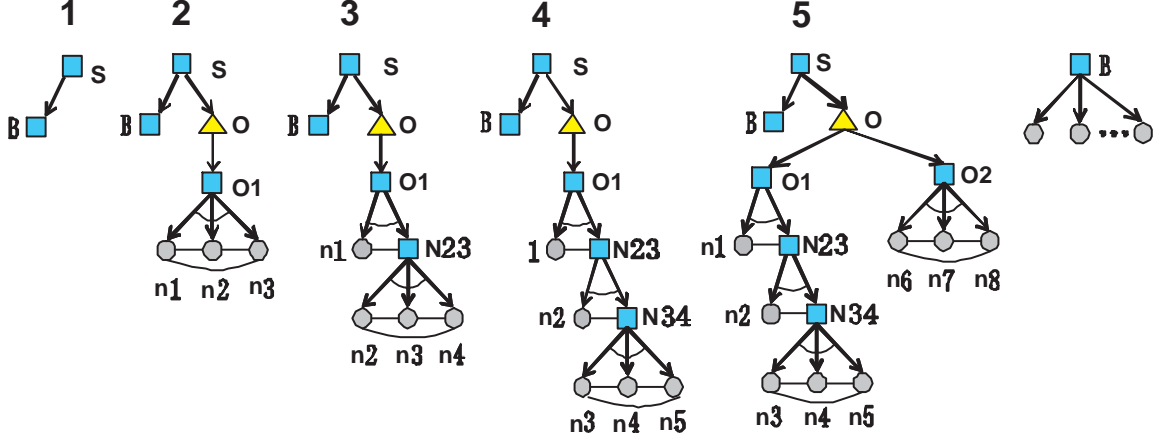


Fig. 6. Graphical Models. Squares, triangles, and circles indicate AND, OR, and LEAF nodes respectively. The horizontal lines denote MRF connections. The far right panel shows the background node generating leaf nodes. The models for $O1$ for panels 2,3 and 4 correspond to the triplets combinations in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

”AND” nodes and ”LEAF” nodes which are depicted in figure (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. MRF’s).

The leaf nodes are indexed by a and will correspond to AF’s in the image. They have attributes (z_a, θ_a, A_a) , where z_a denotes the spatial position, θ_a the orientation, and A_a the appearance. There is also a binary-valued *observability variable* u_a which 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 $L(y) = L_B(y) \cup L_O(y)$, where $L_B(y)$ are the leaves due to the background model, see the far right panel of figure (6), and $L_O(y)$ are the leaves due to the object. We order the nodes in $L_O(y)$ by ”drop-out”, so that the closer the node to the root the lower its number, see figure (6).

In this paper, the only OR node is the object category node O . This corresponds to different aspects of the object. The remaining non-terminal nodes are AND nodes. They include a background node B , object aspect nodes O_i and *clique nodes* of 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 figure (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: (a) the root node S to the object node and the background node, (b) the object node to aspect nodes, (c) a non-terminal node to three leaf nodes, see panel (ii) of figure (6), or (d) a non-terminal node to a clique node and a leaf node, see panel (iii) of figure (6). In case (c) and (d), they correspond to a triplet clique of point features.

Figure (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 which 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 which is represented by a triangle. The object category node O has 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 to have a different appearance.

V. THE DISTRIBUTION DEFINED ON THE PGMM

The structure of the PGMM is specified by figure (7). The PGMM specifies the probability distribution of the AF's 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 which 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 (I) for the notation used in the model.

We define the full distribution to be:

$$P(u, z, A, \theta, y, \omega, \Omega) = P(A|y, \omega^A)P(z, \theta|y, \omega^g)P(u|y, \omega^g)P(y|\Omega)P(\omega)P(\Omega). \quad (2)$$

TABLE I
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)
$\{x_i : i = 1, \dots, N_\tau\}$	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
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

The observed AF's 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:

$$P(x, y, \omega, \Omega) = \sum_{(z_a, A_a, \theta_a) \text{ s.t. } u_a=0} P(u, z, A, \theta, y, \omega, \Omega). \quad (3)$$

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

$P(y|\Omega)$ is the grammatical part of the PGMM (with prior $P(\Omega)$). It generates the topological structure y which 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

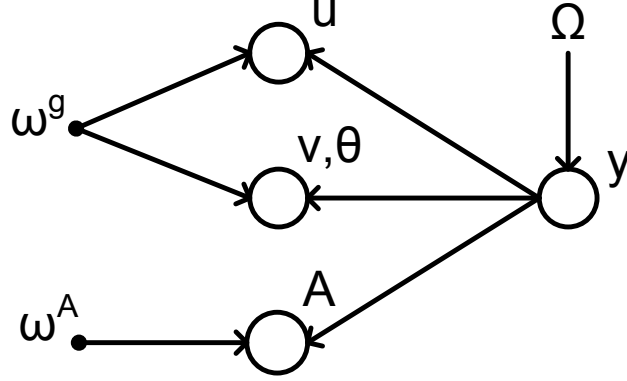


Fig. 7. This figure illustrates the dependencies between the variables. The variables Ω specify the probability for topological structure y . The spatial assignments z of the leaf nodes are influenced by the topological structure y and the MRF variables ω . The probability distribution for the image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

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 AF's. $P(\omega^g, \omega^A)$ is the prior on ω .

A. Generating the leaf nodes: $P(y|\Omega)$

This distribution $P(y|\Omega)$ specifies the probability distribution of the leaf nodes. It determines how many AF's are present in the image (except for those which 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 PCFG's [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.

$$\begin{aligned}
S &\rightarrow \{B, O\} \text{ with prob } 1, \\
O &\rightarrow \{O_i : i = 1, \dots, \rho\} \text{ with prob, } \Omega_i^R, \quad i = 1, \dots, \rho \\
O_i &\rightarrow \{n_a, N_{a+1, a+2}\} \text{ with prob. } 1, \quad a = \beta_i^\Omega, \\
N_{a, a+1} &\rightarrow \{n_a, N_{a+1, a+2}\} \text{ with prob. } 1, \quad \beta_i^\Omega + 1 \leq a \leq \beta_{i+1}^\Omega - 4. \\
N_{\beta_{i+1}^\Omega - 3, \beta_{i+1}^\Omega - 2} &\rightarrow \{n_{\beta_{i+1}^\Omega - 3}, n_{\beta_{i+1}^\Omega - 2}, n_{\beta_{i+1}^\Omega - 1}\} \text{ with prob } 1, \\
B &\rightarrow \{n_{\beta_{\rho+1}^\Omega}, \dots, n_{\beta_{\rho+1}^\Omega + m}\} \text{ with prob } \Omega^B e^{-m\Omega^B} \quad (m = 0, 1, 2, \dots). \tag{4}
\end{aligned}$$

Here $\beta_1^\Omega = 1$. The nodes $\beta_i^\Omega, \dots, \beta_{i+1}^\Omega - 1$ correspond to aspect O_i . Note that these $\{\beta_i^\Omega\}$ are parameters of the model which will be learnt. ρ is the number of aspects and $\{\Omega_i^R\}$ and Ω^B are parameters that specify the distribution (all these will be learnt). We write $\Omega = \{\Omega^B, \Omega_1^R, \dots, \Omega_\rho^R, \beta_1^\Omega, \dots, \beta_{\rho+1}^\Omega, \rho\}$.

B. 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:

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

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 $u_a = 1$).

C. 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: (a) it is invariant to the pose (position, orientation, and scale), and (b) it is easily computable. In order to satisfy both 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:

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

where $P(l(z, \theta) | y, \omega^g)$ (see next equation) 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:

$$P(l | y, \omega^g) = \frac{1}{Z} \exp \left\{ \sum_{a \in \text{Cliques}(y)} \psi_a(\vec{l}(z_a, \theta_a, z_{a+1}, \theta_{a+1}, z_{a+2}, \theta_{a+2}), \omega_a^g) \right\}, \quad (7)$$

where 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 $\vec{l}(z_a, \theta_a, z_{a+1}, \theta_{a+1}, z_{a+2}, \theta_{a+2})$ is given by equation (1).

The potential $\psi_a(\vec{l}(z_a, \theta_a, z_{a+1}, \theta_{a+1}, z_{a+2}, \theta_{a+2}), \omega_a^g)$ specifies geometric regularities of clique C_a which are invariant to the scale and rotation. They are of form:

$$\begin{aligned} & \psi_a(\vec{l}(z_a, \theta_a, z_{a+1}, \theta_{a+1}, z_{a+2}, \theta_{a+2}), \omega_a^g) = \\ & -(1/2)(\vec{l}(z_a, \theta_a, z_{a+1}, \theta_{a+1}, z_{a+2}, \theta_{a+2}) - \vec{\mu}_a^z)^T (\Sigma_a^z)^{-1} (\vec{l}(z_a, \theta_a, z_{a+1}, \theta_{a+1}, z_{a+2}, \theta_{a+2}) - \vec{\mu}_a^z). \end{aligned} \quad (8)$$

where $\omega_a^g = (\mu_a^z, \Sigma_a^z)$ and $\omega^g = \{\omega_a^g\}$.

Now we derive equation (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:

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

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

$$P(z, \theta | l, G) = \delta(z - z(G, l)) \delta(\theta - \theta(G, l)). \quad (10)$$

$P(z, \theta|l, G)$ specifies the positions and orientations z, θ by deterministic functions $z(G, l), \theta(G, l)$ 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 :

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

Substituting equation (10) into equation (11) yields:

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

where 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 $\frac{\partial(\rho, \gamma)}{\partial(l, G)}(l, G)$ is the Jacobian of this transformation (evaluated at (l, G)).

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

D. 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 probability distribution specified by $\omega_a^A = (\mu_a^A, \Sigma_a^A)$:

$$P(A_a|\omega_a^A) = \frac{1}{\sqrt{2\pi}|\Sigma_{A,a}|} \exp\{-(1/2)(A_a - \mu_a^A)^T(\Sigma_a^A)^{-1}(A_a - \mu_a^A)\}. \quad (13)$$

E. 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 which are set to zero mean Gaussians with fixed variance.

F. The Correspondence Problem:

Our formulation of the probability distributions has assumed an ordered list of nodes indexed by a . But 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 AF's 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 AF's 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 $i(a) \in \{0, 1, \dots, N_\tau\}$, where $i(a) = 0$ indicates that a is unobservable (i.e. $u_a = 0$). For background leaf nodes, $i(a) \in \{1, \dots, N_\tau\}$. We constrain all image nodes to be matched so that $\forall j \in \{1, \dots, N_\tau\}$ there exists a unique $b \in L(y)$ s.t. $i(b) = j$ (we create as many background nodes as is necessary to ensure this). To ensure uniqueness, we require that object triplet nodes all have unique matches in the image (or are unmatched) and that background nodes can only match AF's which 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. But 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 equation (5) and replace it by V with prior:

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

This gives the full distribution:

$$P(\{z_i, A_i, \theta_i\}|V, y, \omega^g, \omega^A, \Omega)P(V|y, \omega^g)P(y|\Omega)P(\omega)P(\Omega), \quad (15)$$

with

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

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: (i) the probability of the aspect $P(L_O(y) | \Omega)$ and 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 equation (16). There are variables on the right hand side of the equation which 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 AF's 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 sub-optimal but intuitive and efficient. (It does require that we have at least two vertices assigned in each triplet).

VI. LEARNING AND INFERENCE OF THE MODEL

In order to learn the models, we face three tasks: (I) structure learning, (II) parameter learning to estimate (Ω, ω) , and (III) inference to estimate (y, V) .

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

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

As described in section (VI-A) we use dynamic programming to estimate y^*, V^* efficiently.

Parameter learning occur 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

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

This is performed by an EM algorithm, see section (VI-B), where the summation over the $\{V_\tau\}$ is performed by dynamic programming (the summation over the y 's 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 (VI-C). For each proposed structure, we have a set of parameters W which extends the set of parameters of the previous structure. For each new structure, we evaluate the fit to the data by computing the *score*:

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

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 the images. This is performed by using dynamic programming.

A. 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. Firstly, inference uses dynamic programming via the max rule to calculate the most probable parse tree y^*, V^* for input x . Secondly, 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

$\{y_\tau\}, \{V_\tau\}$. Thirdly, 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. For DP we keep a table of the possible assignments including the unobservable assignment. As mentioned above, we perform the sub-optimal method of replacing missing values z_a, θ_a s.t. $i(a) = 0$ by their most probable estimates.

The conditional distribution is obtained from equations (4,7,13,14).

$$P(y, V, x|\omega, \Omega) = \frac{1}{Z} \exp\left\{ \sum_{a \in C(L_O(y))} \psi_a(\vec{l}(z_{i(a)}, \theta_{i(a)}, z_{i(a+1)}, \theta_{i(a+1)}, z_{i(a+2)}, \theta_{i(a+2)}), \omega_a^g) \right. \\ \left. - (1/2) \sum_{a \in L_O(y)} \{1 - \delta_{i(a),0}\} (A_{i(a)} - \mu_a^A)^T (\Sigma_a^A)^{-1} (A_{i(a)} - \mu_a^A) \right. \\ \left. - \log\{\lambda_\omega / (1 - \lambda_\omega)\} \delta_{i(a),0} + \Omega^B (N_\tau - |L_O(y)|) + \sum_{a \in L_O(y)} \delta_{i(a),0} \right\}. \quad (20)$$

We can re-express this as

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

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

We maximize equation (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 AF's in the image are matched). For each aspect, we use dynamic programming to maximize over V . This can

be done recursively by defining a function $h_a[(z_{i(a)}, A_{i(a)}, \theta_{i(a)}), (z_{i(a+1)}, A_{i(a+1)}, \theta_{i(a+1)})]$ by a forward pass:

$$\begin{aligned}
 & h_{a+1}[(z_{i(a+1)}, A_{i(a+1)}, \theta_{i(a+1)}), (z_{i(a+2)}, A_{i(a+2)}, \theta_{i(a+2)})] = \\
 & \max_{i(a)} \hat{\pi}_a[(z_{i(a)}, A_{i(a)}, \theta_{i(a)}), (z_{i(a+1)}, A_{i(a+1)}, \theta_{i(a+1)}), (z_{i(a+2)}, A_{i(a+2)}, \theta_{i(a+2)})] \\
 & h_a[(z_{i(a)}, A_{i(a)}, \theta_{i(a)}), (z_{i(a+1)}, A_{i(a+1)}, \theta_{i(a+1)})] \quad (22)
 \end{aligned}$$

The forward pass computes the maximum value of $P(y, V, x|\omega, \Omega)$. The backward pass of dynamic programming compute the most probable value V^* . The forward and backward passes are computed for all possible aspects of the model.

As stated earlier in section (V-F), 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, equation (22), for each possible topological structure y . In this paper, the number of topological structures is very small (i.e. less than twenty) 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 next subsection, which requires calculating sums over different correspondences and aspects. For structure learning we need to calculate the score, see equation (19), which also requires summing over different correspondences and aspects. This requires replacing the max in equation (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).

B. EM Algorithm for Parameter Learning

To 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:

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

where:

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

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:

$$\begin{aligned} F[q(\cdot, \cdot), \omega, \Omega] &= \sum_{\{y_\tau\}, \{V_\tau\}} q(\{y_\tau\}, \{V_\tau\}) \log q(\{y_\tau\}, \{V_\tau\}) \\ &\quad - \sum_{\{y_\tau\}, \{V_\tau\}} q(\{y_\tau\}, \{V_\tau\}) \log P(\omega, \Omega, \{y_\tau\}, \{V_\tau\} | \{x_\tau\}), \end{aligned} \quad (25)$$

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

E-step:

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

M-step:

$$\omega^{t+1}, \Omega^{t+1} = \arg \min_{\omega, \Omega} \left\{ - \sum_{\{y_\tau\}, \{V_\tau\}} q^t(\{y_\tau\}, \{V_\tau\}) \log P(\omega, \Omega, \{y_\tau\}, \{V_\tau\} | \{x_\tau\}) \right\}. \quad (27)$$

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:

$$q_\tau^t(\{y_\tau\}, \{V_\tau\}) = P(\{y_\tau\}, \{V_\tau\} | \{x_\tau\}, \omega^t, \Omega^t), \quad (28)$$

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 equation (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 the parameters, $f(\cdot)$ and $\phi(\cdot)$ 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

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

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 form:

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

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 subsection (VI-C1), give good initialization (so we do not need to use standard methods such as multiple initial starting points).

C. 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 $N_{i,i+1}$. This is illustrated in figure (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 non-zero parameters.

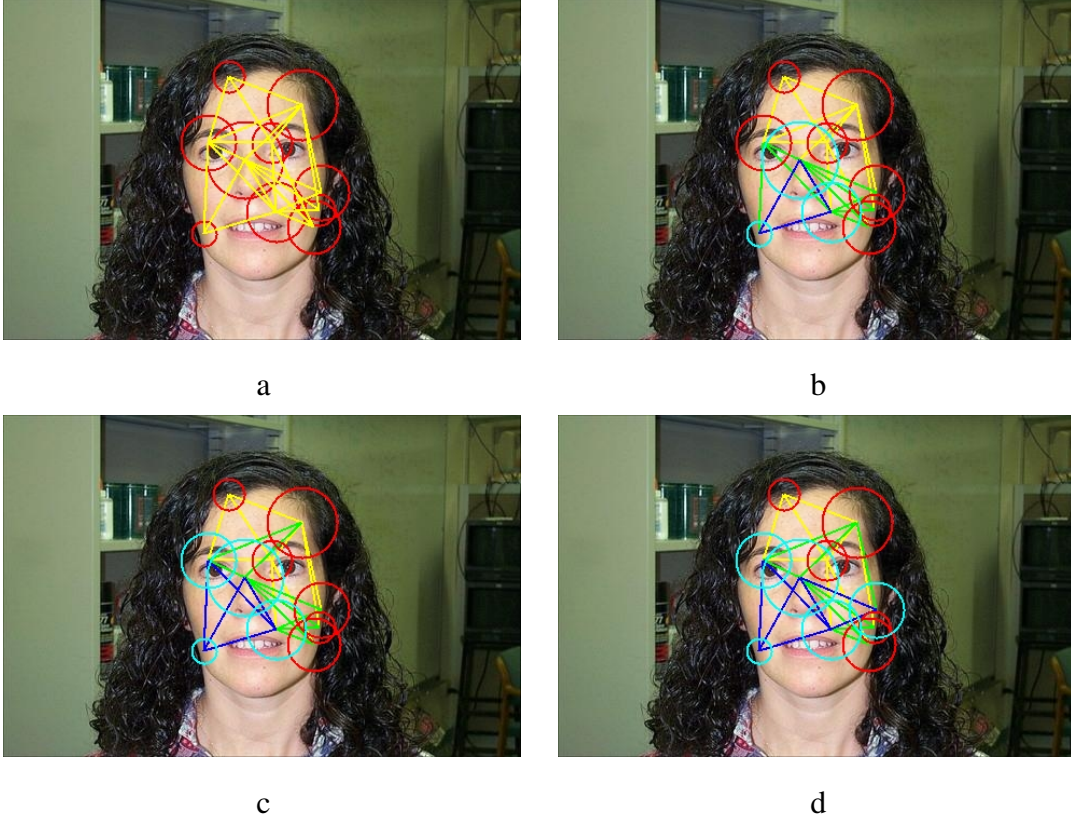


Fig. 8. This figure illustrates structure pursuit. a)image with triplets. b)one triplet induced. c)two triplets induced. d)three triplets induced. Yellow triplets: all triplets from triplet vocabulary. Blue triplets: structure induced. Green triplets: possible extensions for next induction. Circles with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

Our strategy for structure pursuit is as follows, see figures (8,9). 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 equation (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.

1) *The feature and triplet vocabularies* : We construct feature and triplet vocabularies using the features $\{x_i^T\}$ extracted from the image dataset as described in section (III-A).

To get the feature vocabulary, we perform k-means clustering on the attribute data $\{A_i^T\}$

(ignoring the spatial positions $\{v_i^\tau\}$). The centers of the clusters, denoted by $\{F^a\}$, give the feature vocabulary

$$Voc_1 = \{F^a : a \in \Lambda_A\}. \quad (31)$$

To get the triplet vocabulary, we first quantize the attribute data $\{A_i^\tau\}$ to the appearance vocabulary Voc_1 using nearest neighbor (with Euclidean distance). This gives a set of data $D\{z_i^\tau, \theta_i^\tau, F_i^\tau\}$ where each $F_i^\tau \in Voc_1$.

We now construct sets of triplets of features (the features must occur in the same image) where corresponding features have the same appearances (as specified by the vocabulary):

$$D_{abc} = \{(z_\mu, \theta_\mu, z_\nu, \theta_\nu, z_\rho, \theta_\rho) : (z_\mu, \theta_\mu, F_a), (z_\nu, \theta_\nu, F_b), (z_\rho, \theta_\rho, F_c) \in D\} \quad a, b, c \in \Lambda_a. \quad (32)$$

We represent each triplet by its triplet vector \vec{l} , see section (III-B). Next we perform K-means clustering on the set of triplet vectors \vec{l} for each set D_{abc} . This gives a triplet vocabulary Voc_2 which contains the means, the covariances, and the attributes.

2) *Structure Induction Algorithm:* We now have the necessary background to describe our structure induction algorithm. The full procedure is described in the pseudo code in figure (9). Figure (6) shows an example of the structure being induced sequentially.

Initially we assume that all the data is generated by the background model. In the terminology of section (VI), 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 (VI).

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 figure (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 subsection (VI-B). Then, for all these triples we construct the background plus triplet model,

Input: Training Image $\tau = 1, \dots, M$ 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 $T \in Voc_2$
 - * $G' = G \cup T$ (ORing)
 - * Update parameters of G' by EM algorithm
 - * If $Score(G') > Score(G^*)$ Then $G^* = G'$
 - AND-NODE EXTENSION
 - For Image $\tau = 1, \dots, M$
 - * P = the highest probability parse for Image τ by G
 - * For each Triple T in Image τ
 - if $T \cap P \neq \emptyset$
 - $G' = G \cup T$ (ANDing)
 - 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

estimate their parameters and score them. We accept the one with 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 figure (8). Then, for all possible extensions, we use the methods in section (VI) to score the models. We select the one with 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 figure (10).

VII. EXPERIMENTAL RESULTS

Our experiments were designed to give proof of concept for the PGMM. Firstly, 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 101 images (note that most of these approaches are 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 [15]). Our inference algorithm is fast and takes under five seconds (the CPU is AMD Opteron processor 880, 2.4G Hz). Secondly, we illustrate a key advantage of our method that it can both learn and perform inference when the pose (position, orientation, and scale) of the object varies. We check this by creating a new dataset by varying the pose of objects in Caltech 101. Thirdly, we illustrate the advantages of having 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 unsupervised learning is possible even when the training dataset consists of a random mixture of images containing the objects and images which do not (and hence are pure background). Finally we learn a *hybrid model*, where we are given training examples which contain one of several different types of object and learn a model which has different OR nodes for different objects.

A. Learning Individual Objects Models

In this section, we demonstrate the performance of our models for thirteen objects chosen from the Caltech-101 dataset. Each dataset was randomly split into two sets with equal size

(one for training and the other for testing). Note that this particular dataset, the objects appear in standardized orientations. Hence rotation invariance is not necessary. To check this, we also implemented a simpler version of our model which was not rotation invariant by modifying the \vec{l} vector, as described in subsection (III-B). The results of this simplified model were practically identical to the results of the full model, that we now present.

K-means clustering was used to learn the appearance and triplet vocabularies where, typically, K is set to 150 and 1000 respectively. Each row in figure 5 corresponds to some triplets in the same group.

We illustrate the results of the PGMMs in Table (II). A score of 90% means that we get a true positive rate of 90% and a false positive rate of 10%. 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 [15] (which require supervised learning) We also evaluate the ability of the PGMMs to localize the object. To do this, we compute the proportion of AF's of the model that lie within the groundtruth bounding box. Our localization results are shown in Table (III). Note that some alternative methods, such as the bag of keypoints, are unable to perform localization.

The models for individual objects classes, learnt from the proposed algorithm, are illustrated in figure (10). Observe that the generative models have different tree-width and depth. Each subtree of the object node defines a Markov Random Field to describe one aspect of the object. The computational cost of the inference, using dynamic programming, is proportional to the height of the subtree and exponential to the maximum width (only three 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 figure (11).

TABLE II

WE HAVE LEARNT PROBABILITY GRAMMARS FOR 13 OBJECTS IN THE CALTECH 101 DATABASE, OBTAINING SCORES OVER 90% FOR MOST OBJECTS. A SCORE OF 90%, MEANS THAT WE HAVE A CLASSIFICATION RATE OF 90% AND A FALSE POSITIVE RATE OF 10% ($10\% = (100 - 90)\%$). WE COMPARE OUR RESULTS WITH CONSTELLATION MODEL

Dataset	Size	Ours	Constellation Model
Faces	435	97.8	96.4
Motorbikes	800	93.4	92.5
Airplanes	800	92.1	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	–

TABLE III

LOCALIZATION RATE IS USED TO MEASURE THE PROPORTION OF AF'S OF THE MODEL THAT LIE WITHIN THE GROUNDTRUTH BOUNDING BOX.

Dataset	Localization Rate
Faces	96.3
Motorbikes	98.6
Airplanes	91.5

B. 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

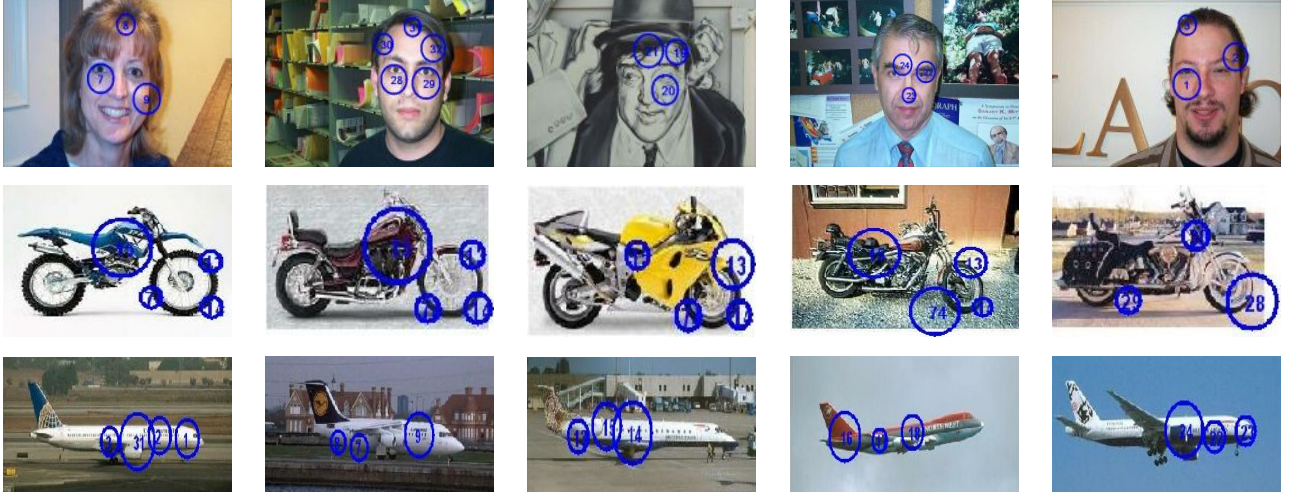


Fig. 11. Parsed Results for Faces, Motorbikes and Airplanes. The circles represent the AF's. The numbers inside the circles give the α index of the nodes, see Table (I).

approach and is due to the triplet representation.

To evaluate PGMMs for this task, we modify the Caltech 101 dataset 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 dataset with rotation and scaling together (where the scaling range is from 60% of the original size to 150% – i.e. $180 * 120 - 450 * 300$).

The PGMM showed only slight degradation due to these pose variations. Table (IV) shows the comparison results. The parsing results (rotation+scale) are illustrated in figure (12).

TABLE IV
INVARIANT TO ROTATION AND SCALE

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

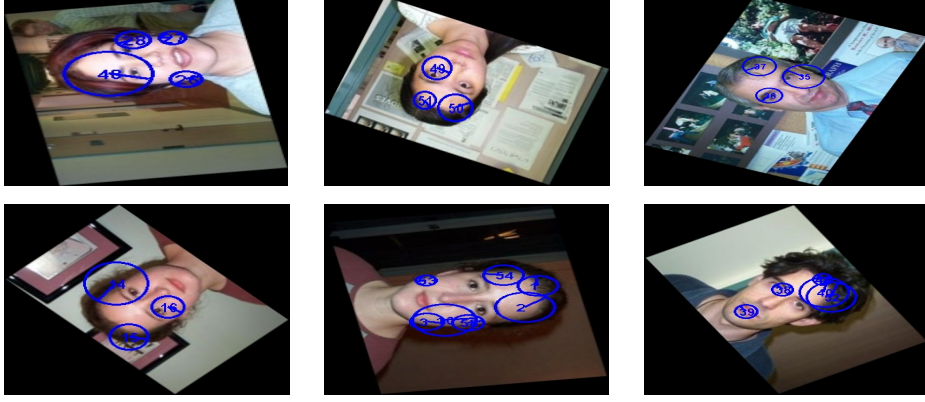


Fig. 12. Parsed Results: Invariant to Rotation and Scale.

C. The Advantages of Variable Graph Structure

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

Firstly, 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 figure (13). This shows that performance increases as the number of OR nodes gets bigger, but this increase is jagged and soon reaches an asymptote.

Secondly, we show that we can learn a PGMM even when the training dataset consists of a random mixture of images containing the object and images which do not. Table (V) 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 which contain the object) and other OR nodes to deal with the remaining images. The overall performance of this PGMM is only slightly worse than the PGMM trained on standard images (see section (VII-A)). Note that the performance of alternative methods [15] would tend to degrade badly under these conditions.

Thirdly, we show that we can learn a model for an object class. We use a hybrid class which 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

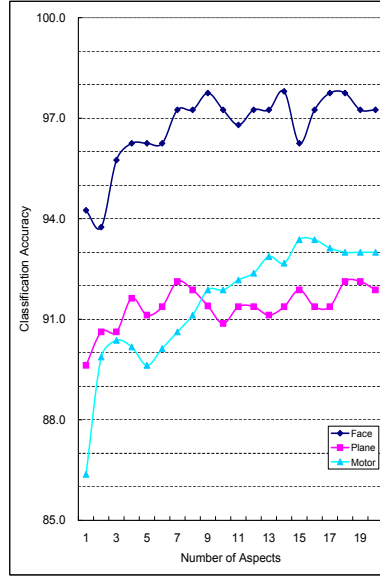


Fig. 13. Analysis of the effects of adding OR nodes. Observe that performance rapidly improves, compared to the single MRF model with only one aspect, as we add extra aspects. But this improvement reaches an asymptote fairly quickly. (This type of result is obviously dataset dependent).

TABLE V

THE PGMM ARE LEARNT ON DIFFERENT TRAINING DATASETS WHICH CONSIST OF A RANDOM MIXTURE OF IMAGES CONTAINING THE OBJECT AND IMAGES WHICH DO NOT.

Dataset	Training Set		Testing Set		Classification Rate
	Object Images	Background Images	Object Images	Background Images	
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

TABLE VI

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

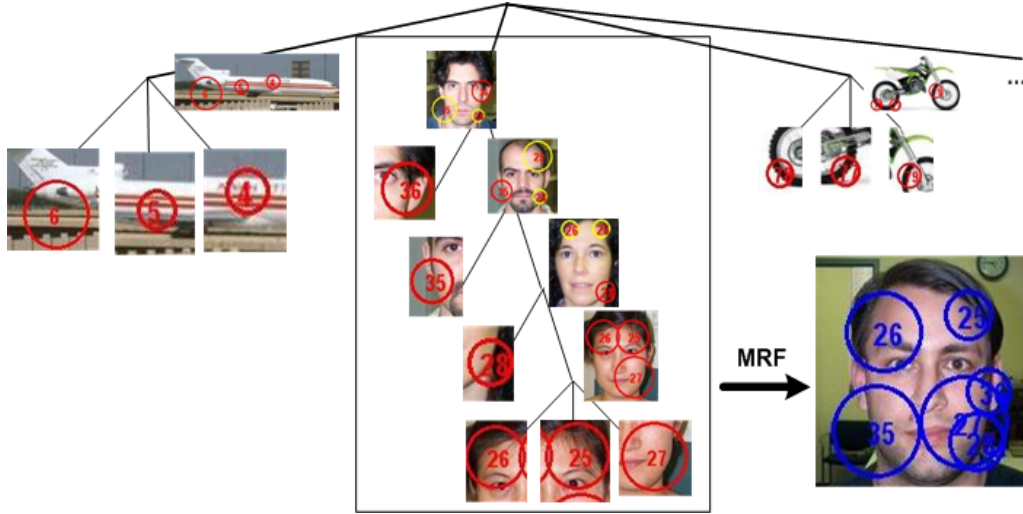


Fig. 14. Hybrid Model learnt for Faces, Motorbikes and Airplanes.

the datasets of faces, airplanes, and motorbikes. Similarly, we test the hybrid model on examples selected randomly from these three datasets.

The learnt hybrid model is illustrated in figure (14). It breaks down nicely into OR's of the models for each object. Table (VI) shows the performance for the hybrid model. This demonstrates that the proposed method can learn a model for the class with extremely large variation.

VIII. DISCUSSION

This paper introduced PGMMs and showed that they can be learnt in an unsupervised manner and perform tasks such as classification and localization of objects in unknown backgrounds. We also showed that PGMMs were invariant to 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 which include several different types of object.

More technically, PGMMs combine elements of probabilistic grammars and Markov random fields (MRFs). The grammar component enables them to adapt to different aspects while the MRF 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 dynamic programming. 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: (a) we can learn probabilistic models for a variety of different objects and perform rapid inference (less than five seconds), (b) that our learning and inference is invariant to scale and rotation, (c) that we can learn models in noisy data, for hybrid classes, and that 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 extend this approach by allowing a more sophisticated representation and using a richer set of image features.

ACKNOWLEDGEMENTS

We gratefully acknowledge support from the W.M. Keck Foundation, from the National Science Foundation with NSF grant number 0413214, and from the National Institute of Health with grant RO1 EY015261. We thank Ying-Nian Wu and Song-Chun Zhu for stimulating discussions. We thank SongFeng Zheng for helpful feedback on drafts of this paper.

REFERENCES

- [1] D. McAllester, M. Collins, and F. Pereira, “Case-factor diagrams for structured probabilistic modeling,” in *Proc. of the 20th conference on Uncertainty in artificial intelligence*, Arlington, Virginia, United States, 2004, pp. 382–391.
- [2] J. Lafferty, A. McCallum, and F. Pereira, “Conditional random fields: Probabilistic models for segmenting and labeling sequence data,” in *Proc. 18th International Conf. on Machine Learning*. Morgan Kaufmann, 2001, pp. 282–289.
- [3] D. Klein and C. Manning, “Natural language grammar induction using a constituent-context model,” in *Adv. in Neural Information Proc. 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,” in *IEEE Proc. of the Conf. on Computer Vision and Pattern Recognition*, Washington, DC, USA, 2006, pp. 943–950.
- [6] L. S. Zettlemoyer and M. Collins, “Learning to map sentences to logical form: Structured classification with probabilistic categorical grammars,” in *Proc. of the 21th Annual Conference on Uncertainty in Artificial Intelligence*, Arlington, Virginia, 2005, pp. 658–66.
- [7] B. Ripley, *Pattern Recognition and Neural Networks*. Cambridge University Press, 1996.
- [8] C. Manning and C. Schütze, *Foundations of Statistical Natural Language Processing*. Cambridge, MA: MIT Press, 1999.
- [9] Z. Tu, X. Chen, A. Yuille, and S.-C. Zhu, “Image parsing: Unifying segmentation, detection, and recognition,” *International Journal of 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,” in *Proc. of IEEE Conf. Computer Vision and Pattern Recognition*, 2003.
- [12] —, “A sparse object category model for efficient learning and exhaustive recognition,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, San Diego*, vol. 1, 2005, pp. 380–397.
- [13] D. J. Crandall and D. P. Huttenlocher, “Weakly supervised learning of part-based spatial models for visual object recognition,” in *ECCV (1)*, 2006, pp. 16–29.
- [14] D. Crandall, P. Felzenszwalb, and D. Huttenlocher, “Spatial priors for part-based recognition using statistical models,” in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition - Volume 1*, Washington, DC, USA, 2005, pp. 10–17.
- [15] G. Csurka, C. Bray, C. Dance, and L. Fan, “Visual categorization with bags of keypoints,” in *In Workshop on Statistical Learning in Computer Vision, ECCV*, 2004.
- [16] M. Meila and M. I. Jordan, “Learning with mixtures of trees,” *Journal of Machine Learning Research*, vol. 1, pp. 1–48, 2000.
- [17] S. D. Pietra, V. J. D. Pietra, and J. D. Lafferty, “Inducing features of random fields,” *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 19, no. 4, pp. 380–393, 1997.
- [18] S. Zhu, Y. Wu, and D. Mumford, “Minimax entropy principle and its application to texture modeling,” *Neural Computation*, vol. 9, no. 8, Nov. 1997.
- [19] A. McCallum, “Efficiently inducing features of conditional random fields,” in *Nineteenth Conference on Uncertainty in Artificial Intelligence*, 2003.
- [20] N. Friedman, “The bayesian structural em algorithm,” in *Proc. of the 14th Annual Conference on Uncertainty in Artificial Intelligence*. San Francisco, CA: Morgan Kaufmann, 1998, pp. 129–13.

- [21] L. Zhu, Y. Chen, and A. Yuille, “Unsupervised learning of a probabilistic grammar for object detection and parsing,” in *Advances in Neural Information Processing Systems 19*, B. Schölkopf, J. Platt, and T. Hoffman, Eds. Cambridge, MA: MIT Press, 2007.
- [22] L. Shams and C. von der Malsburg, “Are object shape primitives learnable?” *Neurocomputing*, vol. 26-27, pp. 855–863, 1999.
- [23] L. FeiFei, R. Fergus, and P. Perona, “One-shot learning of object categories,” *IEEE Trans. Pattern Recognition and Machine Intelligence*, 2006.
- [24] T. L. B. J. Ponce, M. Everingham, D. A. Forsyth, M. Hebert, S. Lazebnik, M. Marszalek, C. Schmid, B. C. R. and A. Torralba, C. K. I. Williams, J. Zhang, and A. Zisserman, “Dataset issues in object recognition,” in *Toward Category-Level Object Recognition (Sicily Workshop 2006)*, J. Ponce, M. Hebert, C. Schmid, and A. Zisserman, Eds. LNCS, 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. J. Comput. Vision*, vol. 45, no. 2, pp. 83–105, 2001.
- [27] D. G. Lowe, “Distinctive image features from scale-invariant keypoints,” *Int. J. Comput. 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,” in *BMVC*, 2004.
- [30] Y. Jin and S. Geman, “Context and hierarchy in a probabilistic image model,” in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, Washington, DC, USA, 2006, pp. 2145–2152.
- [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.