Unsupervised Learning of Visual Feature Hierarchies

Fabien Scalzo and Justus Piater

Montefiore Institute
University of Liège
4000 Liège, Belgium
fscalzo@mednet.ucla.edu,Justus.Piater@ulg.ac.be

Abstract. We propose an unsupervised, probabilistic method for learning visual feature hierarchies. Starting from local, low-level features computed at interest point locations, the method combines these primitives into high-level abstractions. Our appearance-based learning method uses local statistical analysis between features and Expectation-Maximization to identify and code spatial correlations. Spatial correlation is asserted when two features tend to occur at the same relative position of each other. This learning scheme results in a graphical model that constitutes a probabilistic representation of a flexible visual feature hierarchy. For feature detection, evidence is propagated using Belief Propagation. Each message is represented by a Gaussian mixture where each component represents a possible location of the feature. In experiments, the proposed approach demonstrates efficient learning and robust detection of object models in the presence of clutter and occlusion and under view point changes.

1 Introduction

The visual feature representation is one of the most important issues for learning and recognition applications in computer vision. In the present work, we propose a new approach to representing and learning visual feature hierarchies in an unsupervised manner. Our hierarchical representation is inspired by the compositional nature of objects. Most objects encountered in the world, which can be either man-made or natural objects, are composed of a number of distinct constituent parts (e.g., a face contains a nose and two eyes, a phone possesses a keypad). If we examine these parts, it becomes obvious that they are in turn recursively composed of other subparts (e.g., an eye contains an iris and eyelashes, a keypad is composed of buttons). This ubiquitous observation constitutes our main motivation for arguing that a hierarchical representation must be taken into account to model objects in more flexible and realistic way.

Our long-term goal is thus to learn visual feature hierarchies that correspond to object/part hierarchies. The development of a hierarchical and probabilistic framework that is tractable is terms of complexity is a central problem for many computer vision applications such as visual tracking, object recognition and categorization, face recognition, stereo matching and image retrieval.

This work was supported by the Wallonian Ministry of Research (D.G.T.R.E.) under Contract No. 03/1/5438.

In this paper, we combine the approaches of local, appearance-based feature detection and unsupervised model learning in a new visual feature recognition scheme. The principal objective is to obtain a probabilistic framework that allows the organization of complex visual feature models. The main idea is to use a graphical model to represent the hierarchical feature structure. In this representation, which is detailed in Section 2, the nodes correspond to the visual features. The edges model both the spatial arrangement and the statistical dependence between nodes. The formulation in terms of graphical models is attractive because it provides a statistical model of the variability of shape and appearance. The shape and appearance models are specified separately by the edges and the leaf nodes of the graph, respectively.

An unsupervised feature learning method that allows the construction of a hierarchy of visual features is introduced in Section 4. The proposed framework accumulates statistical evidence from feature observations in order to find *conspicuous coincidences* of visual feature co-occurrences. The structure of the graph is iteratively built by combining correlated features into higher-level abstractions. Our learning method is best explained by first presenting the detection process, which is described in Section 3. During detection, our scheme starts by computing local, low-level features at interest point locations. These features serve to annotate the observable leaf nodes of the graph. Then, at the second step, Belief Propagation [9], a message-passing algorithm, is used to propagate the observations up the graph, thus inferring the belief associated with higher-level features that are not directly observable. The functioning and the efficacy of our method are illustrated in Section 5. Finally, Section 6 provides a discussion of related work.

2 Representation

In this section, we introduce a new part-based and probabilistic representation of visual features (Figure 1). In the proposed graphical model, nodes represent visual features and are annotated with the detection information for a given scene. The edges represent two types of information: the relative spatial arrangement between features, and their hierarchical composition. We employ the term *visual feature* in two distinct contexts:

Primitive visual features are low-level features. They are represented by a local descriptor. For this work, we used simple descriptors constructed from normalized pixel values and located at Harris interest points [4], but our system does not rely on any particular feature detector. Any other feature detector [7] can be used to detect and extract more robust information.

Compound visual features consist of flexible geometrical combinations of other subfeatures (primitive or compound features).

Formally, our graph $\mathcal{G}$ is a mathematical object made up of two sets: a vertex set $\mathcal{V}$, and a directed edge set $\overrightarrow{\mathcal{E}}$. For any node $s \in \mathcal{V}$, the set of parents and the set of children are respectively defined as $U(s) = \{u_i \in \mathcal{V} | (u_i, s) \in \overrightarrow{\mathcal{E}}\}$ and $C(s) = \{c_i \in \mathcal{V} | (s, c_i) \in \overrightarrow{\mathcal{E}}\}$. Information about feature types and their specific occurrences in an image will be represented in the graph by annotations of vertices and edges, as described next.

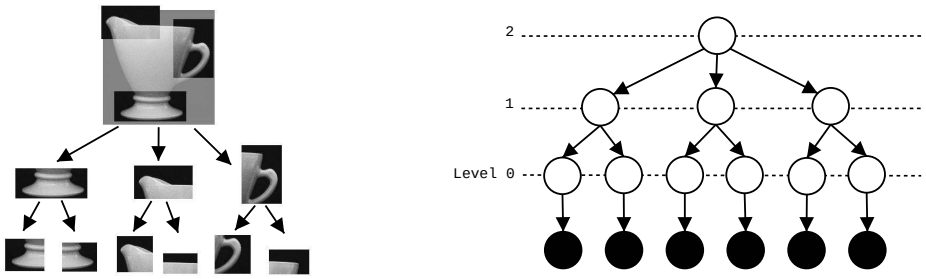

Fig. 1. Object part decomposition (left) and corresponding graphical model (right).

2.1 The Vertex Set

The vertices $s \in \mathcal{V}$ of the graph represent features. They contain the feature activations for a given image. Our graphical model associates each node with a hidden random variable $x \in \mathcal{R}^2$ representing the spatial probability distribution of the feature in the image. This random variable is continuous and defined in a two-dimensional space, where the dimensions $\mathcal{X}, \mathcal{Y}$ are the location in the image. For simplicity, we assume that the distribution of x can be approximated by a Gaussian mixture. In order to retain information about feature orientation, we associate a mean orientation $\theta_i \in [0, 2\pi[$ to each component of the Gaussian mixture. Primitive feature classes lie at the first level of the graph. They are represented by a local descriptor and are associated with an observable random variable y, defined in the same two-dimensional space as the hidden variables x, and are likewise approximated by a Gaussian mixture.

2.2 The Edge Set

An edge $e \in \overrightarrow{\mathcal{E}}$ of the graph models two aspects of the feature structure. First, whenever an edge links two features it signifies that the destination feature is a part of the source feature. Since the observation of a part may have more or less impact on the perceived presence of a compound feature, we use a parameter $b_{x_c}^{x_u}$ to quantify the relative contribution of the subfeature x_c to the presence of the parent feature x_u.

Second, an edge describes the spatial relation between the compound feature and its subfeature in terms of the distance and the relative orientation of the subfeature with respect to the compound feature. The shape uncertainty is represented by a Gaussian probability density that models the relative position of one feature versus the other. This Gaussian, which is described by a relative location and a covariance matrix Σ, allows us to deal with deformable features and offers invariance properties.

The annotation associated with an edge, $\{x_c, x_u, d, \theta_r, \Sigma, b_{x_c}^{x_u}\}$, describes both the geometric relation (distance d, relative orientation θ_r) between two feature classes and the combination coefficient $b_{x_c}^{x_u}$. It does not contain any specific information about a given scene but represents a static view of the feature hierarchy and shape.

3 Feature Detection

In the preceding section, we defined the structure of our graphical model used to represent visual features. We now turn to the detection of a given visual feature in an image, given its graphical-model representation. The goal of the feature detection process is to estimate the probability density function $\hat{p}(x|y)$ of features given the primitives detected in the image. The detection process can be summarized by two steps. First, we use the primitives of a scene to instantiate the observed variables y in the graph. Second, we apply Belief Propagation to infer the probability densities of the hidden variables.

3.1 Primitive Detection

Primitive detection matches the detected local descriptors in an image with the most similar existing feature descriptor at the leaf nodes of the available graphs. We use the observed descriptor parameters (location, orientation, similarity) to annotate the variable y_s of the corresponding node $s \in \mathcal{V}$. This observed variable y_s represents the degree of presence of the observed primitive feature, and is modeled by a Gaussian mixture. Each component Gaussian represents a possible location of the feature and is set to the detected descriptor location. The weight w_i of a component is inversely proportional to the Mahalanobis distance between the observed descriptor and the most similar feature class. The orientation of the feature is determined from the corresponding descriptor orientation, and is associated to each Gaussian. In summary, each component of an observed random variable y_s is defined by a location α_i, a weight w_i and an orientation θ_i that are determined by the detected descriptor occurrences.

3.2 Compound Feature Detection

The detection process that computes the presence of high-level features is based on Belief Propagation (BP) [9]. To initialize the graph, we annotate the observed variables with the detected primitives (Section 3.1). Then the main idea of this inference algorithm is to propagate information by successively transmitting upward (λ) and downward (π) messages in the graph. The λ-messages represent the possible presence of a parent feature, given the presence information of the current node. Similarly, each π-message represents the location of a node with respect to its parent nodes.

In our representation, evidence is incorporated exclusively via the variables y_i representing primitives at the leaves of the graph. Higher-level features are not directly observable. Thus, the standard BP algorithm can be simplified to a single upward propagation step where it successively estimates the probability of each node, going from the leaves to the root of the graph. This procedure is implemented by the following update rule that links each variable x of our graphical model to its child nodes $c_i \in C(x)$:

$$x = b^x_{c_1}\vartheta(c_1) + b^x_{c_2}\vartheta(c_2) + \cdots + b^x_{c_n}\vartheta(c_n) \tag{1}$$

where $b^x_{c_i}$ is the combination coefficient of node c_i. The location and the orientation between a compound feature and a subfeature may be different. We therefore introduce a

linear function ϑ that performs a spatial transformation. It translates the probability density of each variable c_i according to the direction and the distance of the corresponding edge $e(x, c_i)$.

J. Pearl demonstrated that BP converges to the optimal solution on tree-structured graphical models, while our graph may contain loops. However, BP has been shown empirically to provide good approximate solutions on a variety of loopy graphs.

As we explained in Section 2.1, our representation models nodes as continuous random variables approximated by mixtures of Gaussians. Likewise, the estimation of the conditional probability density of a node will take the form of a mixture of Gaussians $\hat{p}(x|y) = \sum_{i=1}^{N} w_{x(i)} \mathcal{G}(x; \mu_{x(i)}, \Sigma_{x(i)})$, where μ_x, Σ_x, w_x are respectively vectors of mean, covariance and weight. They are computed by combining the belief obtained at the child nodes (Eq. 2). For simplicity of notation, the following formulas are given for child nodes c_i composed of a single Gaussian (the case of mixtures is analogous).

$$\mu_x = \frac{\sum_{i=1}^{N_c} \mu_{c_i}^{\vartheta} \Sigma_{c_i}}{\sum_{i=1}^{N_c} \Sigma_{c_i}} \qquad \Sigma_x = \left(\sum_{i=1}^{N_c} \frac{1}{\Sigma_{c_i}} \right)^{-1} \qquad \mu_{c_i}^{\vartheta} = b_{c_i}^{x} \vartheta(c_i) \tag{2}$$

Feature Orientation In order to estimate the most likely orientation of a feature, we use the orientations associated to each component of the current Gaussian mixture. We compute the mean orientation $\overline{\theta}_x(l)$ of mixture components weighted by their corresponding weights w_i: $\tan \overline{\theta}_x(l) = \frac{\mathcal{S}_x(l)}{\mathcal{C}_x(l)}$ where $\mathcal{C}_x(l) = \sum_{i=1}^{n} v_i w_i \cos\theta_i$ and $\mathcal{S}_x(l) = \sum_{i=1}^{n} v_i w_i \sin\theta_i$. In these equations, l is a location in the image, θ_i is the main orientation of component x_i, n is the number of components and $v_i = \mathcal{R}(l, x_i)$ is the response of Gaussian component x_i at point l.

4 Visual Feature Learning

In this section, we introduce our unsupervised feature learning method that allows the construction of a hierarchy of visual features. The general idea of our algorithm is to accumulate statistical evidence from the relative positions of observed features in order to find frequent visual feature co-occurrences. The structure of our graphical model is iteratively built by combining correlated features into new visual feature abstractions. First, the learning process votes to accumulate information on the relative position of features and it extracts the feature pairs that tend to be located in the same neighborhood. Secondly, it estimates the parameters of the geometrical relations using either Expectation-Maximization (EM) or a voting scheme. It finally creates new feature nodes in the graph by combining spatially correlated features. In the following sections, we describe the three main steps of this unsupervised learning procedure.

4.1 Spatial Correlation

The objective of this first step of our learning process is to find spatially correlated features. A spatial correlation exists between two features if they are often detected in the same neighborhood. Co-occurrence statistics are collected from multiple feature

occurrences within one or across many different images. The procedure to find correlated features is summarized in Algorithm 1. After its completion, we obtain a vote array $\mathcal{S}$ concerning the relative locations of correlated features. Before the first iteration we apply K-means clustering algorithm to the set of feature descriptors. This identifies primitive classes from the training set and is used to create the first level of the graph.

Algorithm 1 Detection of Spatial Correlations

Successively extract each image I from the training set
Detect all features $f_I = \{f_{i_0} \dots f_{i_n}\} \in \mathcal{G}$ in image I
for each pair $[f_i, f_j]$ where f_j is observed in the neighborhood of f_i **do**
  Compute the relative position $p_r \in \mathcal{R}^2$ of f_j given f_i
  Vote for the corresponding observation $[f_i, f_j, p_r]$ in table $\mathcal{S}$
end for
Keep all class pairs $[f_i, f_j]$ where $\sum_{p_r} \mathcal{S}[f_i, f_j, p_r] > t_c$

4.2 Spatial Relations

In our framework, spatial relations are defined in terms of distance and direction between features. We implemented two solutions to estimate these parameters. The first method uses the Expectation-Maximization (EM) algorithm, and the second implements a fast discrete voting scheme to find location evidence. The estimated geometrical relations are used during feature generation (Section 4.3) in order to create new features. First, however, we give some details on both methods for the estimation of spatial relations.

Expectation-Maximization In principle, a sample of observed spatial relations x_r between two given features can be approximated by a Gaussian mixture, where each component represents a cluster of relative positions μ_k of one of the two features f_j with respect to the other, the *reference feature* f_i: $p(x_r; \Theta) = \sum_{k=1}^{K} w_k \mathcal{G}_k(x_r; \mu_k, \theta_k)$. EM is used to estimate the parameters of the spatial relation between each correlated feature pair $[f_i, f_j] \in \mathcal{S}$. It maximizes the likelihood of the observed spatial relations over the model parameters $\Theta = (w_{1\dots K}; \mu_{1\dots K}; \theta_{1\dots K})$. The Expectation (E) and Maximization (M) steps of each iteration of the algorithm are defined as follows:

Step E Compute the current expected values of the component indicators $t_k(x_i)$, $1 \leq i \leq n$, $1 \leq k \leq K$, where n is the number of observations, K is the number of components and q is the current iteration:

$$t_k^{(q)}(x_i) = \frac{\hat{w}_k^{(q)} \mathcal{G}\left(x_i; \hat{\mu}_k^{(q)}, \hat{\theta}_k^{(q)}\right)}{\sum_{l=1}^{K} \hat{w}_l^{(q)} \mathcal{G}\left(x_i; \hat{\mu}_l^{(q)}, \hat{\theta}_l^{(q)}\right)} \tag{3}$$

Step M Determine the value of parameters Θ^{q+1} containing the estimates $\hat{w}_k, \hat{\mu}_k, \hat{\theta}_k$ that maximize the likelihood of the data $\{x\}$ given the $t_k(x_i)$:

$$\begin{aligned} \hat{w}_k^{(q+1)} &= \tfrac{1}{n}\textstyle\sum_{i=1}^{n} t_k^{(q)} \qquad \hat{\mu}_k^{(q+1)} = \textstyle\sum_{i=1}^{n} t_k^{(q)}(x_i) \Big/ \sum_{i=1}^{n} t_k^{(q)} \\ \hat{\theta}_k^{(q+1)} &= \textstyle\sum_{i=1}^{n} t_k^{(q)} \left(x_i - \hat{\mu}_k^{(q+1)}\right)\left(x_i - \hat{\mu}_k^{(q+1)}\right)^T \Big/ \sum_{i=1}^{n} t_k^{(q)} \end{aligned} \tag{4}$$

In our implementation, a mixture of only two Gaussian components ($K = 2$) is used to model spatial relations. The first component represents the most probable relative position, and the second is used to model the noise. When the location μ_1 of the first component is estimated, it is projected into a cylindrical space defined by distance d and orientation θ parameters. We store the corresponding information $[f_i, f_j, d, \theta, \Sigma]$ in a table $\mathcal{T}$.

Voting A faster method to estimate spatial relations is to discretize distance and direction between features. The idea is to create a bi-dimensional histogram for every correlated feature pair $[f_i, f_j] \in \mathcal{S}$. The dimensions of these histograms are the distance d and the relative direction θ from features f_i to f_j. Each observation $[f_i, f_j, p_r]$ stored in table $\mathcal{S}$ is projected into a cylindrical space $[d, \theta]$ and votes for the corresponding entry $[d, \theta]$ of histogram $\mathcal{H}[f_i, f_j]$. After the completion of this voting procedure, we look for significant local maxima in the 2D histograms and store them in the table $\mathcal{T}$. In our implementation, the distances are expressed relative to the part size and are discretized into 36 bins, while the directions are discretized into 72 bins (5-degree precision).

4.3 Feature generation

When a reliable reciprocal spatial correlation is detected between two features $[f_i, f_j]$, the generation of a new feature in our model is straightforward. We combine these features to create a new higher-level feature by adding a new node f_n to the graph. We connect it to its subfeatures $[f_i, f_j]$ by two edges e_i, e_j that are added to $\overrightarrow{\mathcal{E}}$. Their parameters are computed using the spatial relation $\{\mu_{i,j}, \mu_{j,i}\}$ obtained from the preceding step, and are stored in table $\mathcal{T}$.

The generated feature is located at the midpoint between the subfeatures. Thus the distance from the subfeatures to the new feature is set to the half distance between the subfeatures $[f_i, f_j]$; $\mu_1 = \mu_{i,j}/2$, $\mu_2 = \mu_{j,i}/2$ and is copied to the new edges; $e_i(f_i, f_n) = \{\mu_1, \Sigma_1\}$, $e_j(f_j, f_n) = \{\mu_2, \Sigma_2\}$.

5 Experiments

In this section, we illustrate our visual feature learning scheme on an object recognition task using several objects of the Columbia University Object Image Library (COIL-100) [8]. This library is very commonly used in object recognition research and contains color images of 100 different objects. Each image was captured by a fixed camera at pose intervals of 5 degrees. For our experiments, we used 5 neighboring views of an object to build the corresponding object model. When the learning process is completed, the model is thus tuned for a given view of the object.

As we mentioned before, our system does not depend on any particular feature detector. We used Harris interest points and rotation invariant descriptors comprising 256 pixel values. Any other feature detector can be used to detect and extract more robust information. We deliberately used simple feature to demonstrate the functioning of our method. To estimate the primitives of each object model, we used K-Means to cluster the feature space. The number of classes was selected according to the BIC criterion [13]. For the object presented in Figure 2, the learning process used 16 feature classes (generated by K-Means) to extract correlated features of the same level in the graphical model. For the first level of the graph, it found 7 spatial relations between features that were then used to build the next level of the graph. In order to avoid excessive growth of the graph due to the feature combinatorics, we only kept the most salient spatial relations between features. Figure 3 shows the graph learned on different objects.

Figure 5 illustrates the viewpoint invariance of the object model. To generate the graph, we ran the detection process on a series of images differing in viewing angle by increments of 5 degrees. We show the maximum response of our model for each image (the detection for each image is presented in Figure 2). The model responded maximally to one of the training views, with the response gradually falling off as the image was transformed away from its preferred view. We can determine the invariance range of the object model by comparing the maximum response of all views with the responses of distractors (20 images were taken randomly from coil-100 database, some of these are presented in Figure 4). The invariance range is then defined as the range over which the model response is greater than to any of the distractor objects. For the test image presented in Figure 5, the minimum response to establish the presence of the object was approximately 0.45. We obtained an average viewpoint invariance over 50 degrees. These results are remarkable considering the fact that we did not use affine-invariant features at the leaf level.

We also tested, in Figure 4, the robustness of our model in a cluttered scene containing three instances of the object. As we explained in Section 3, the detection process starts with low-level feature detection and then propagate evidence in graph. In this image, many primitive features of the object class are detected in the image. This is due to the fact that the local appearance of some interest points is similar to the primitives of the model. However, the use of geometric relations to infer the presence of higher-level feature allows an unambiguous detection. As we can see for the object on the right of Figure 4, only a few features are needed to detect the object. The response of the model increases with the number of spatially coherent features detected. On the left, a major portion of the object is present in the image and leads to a strong response of the model.

6 Discussion

During the past years, great attention has been paid to unsupervised model learning applied to object models [10]. In theses techniques, objects are represented by parts, each modeled in terms of both shape and appearance by Gaussian probability density functions. This concept, which originally operated in batch mode, has been improved by introducing incremental learning [5]. Another improvement [3] used information obtained from previously learned models. In parallel, Agarwal *et al.* [1] presented a

method for learning to detect objects that is based on a sparse, part-based representations. The main limitation of these schemes lies in the representation because it only contains two levels, the features and the models.

In previous work, we used a Bayesian network classifier for constructing visual features by hierarchical composition of primitive features[11]. However, the spatial arrangement of primitives was rigid and limited the robustness of the system.

Arguing that existing techniques fail to exploit the structure of the graphical models describing many vision problems, Sudderth *et al.*[14] presented Nonparametric Belief Propagation (NBP) applicable to general distributions. Our framework can be extended using NBP instead of classical BP in order to perform a more robust detection.

A hierarchical model of recognition in the visual cortex has been proposed by Riesenhuber *et al.* [12] where spatial combinations are constructed by alternately employing a maximum and sum pooling operation. Wersing *et al.* [15] used a sparse-coding learning rule to derive local feature combinations in a visual hierarchy. However, in such models there exists no explicit representation of object structure.

In neuroscience, recent evidence [2] reinforces the idea that the coding of geometrical relations in high-level visual features is essential. Moreover, recent work suggests that the visual cortex represents objects hierarchically by parts or subfeatures [6].

The framework presented in this paper offers several significant improvements over current methods proposed in the literature. Taking advantage of graphical models, we represent shape and appearance separately. This allows us to deal with shape deformation and appearance variability. The hierarchical model presented here opens a wide door to other computer vision applications. Several directions can be pursued; the most promising and challenging is certainly the unsupervised discovery of object categories. Future work will focus on the use of Nonparametric Belief Propagation (NBP) and the integration of the hierarchical model in a supervised learning environment.

References

1. S. Agarwal and D. Roth. Learning a sparse representation for object detection. In *ECCV*, volume 4, pages 113–130, 2002.
2. E.E. Cooper and T.J. Wojan. Differences in the coding of spatial relations in face identification and basic-level object recognition. In *J. of Exp. Psychology*, volume 26, pages 470–488, 2000.
3. L. Fei-Fei, R. Fergus, and P. Perona. A Bayesian approach to unsupervised one-shot learning of object categories. In *ICCV*, pages 1134–1141, 2003.
4. C. Harris and M. Stephens. A combined corner and edge detector. In *ALVEY Vision Conference*, pages 147–151, 1988.
5. S. Helmer and D. G. Lowe. Object recognition with many local features. In *Workshop on Generative Model Based Vision (GMBV)*, Washington, D.C., July 2004.
6. Y. Lerner, T. Hendler, D. Ben-Bashat, M. Harel, and R. Malach. A hierarchical axis of object processing stages in the human visual cortex. In *Cerebral Cortex*, volume 4, pages 287–97, 2001.
7. K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. In *CVPR*, volume 2, pages 257–263, June 2003.
8. S. A. Nene, S. K. Nayar, and H. Murase. Columbia object image library (coil-100), 1996.

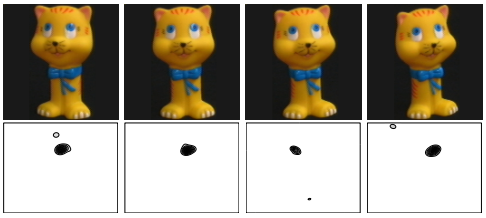

Fig. 2. Evidence map of an object model on a series of images differing in viewing angle.

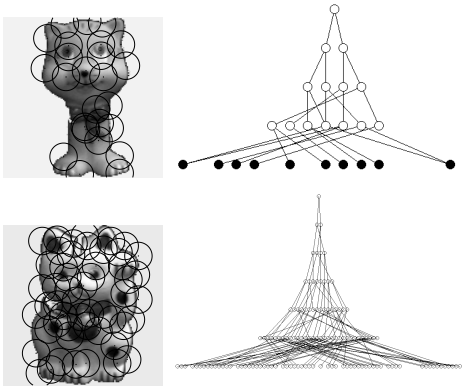

Fig. 3. Graphical models learned on two objects.

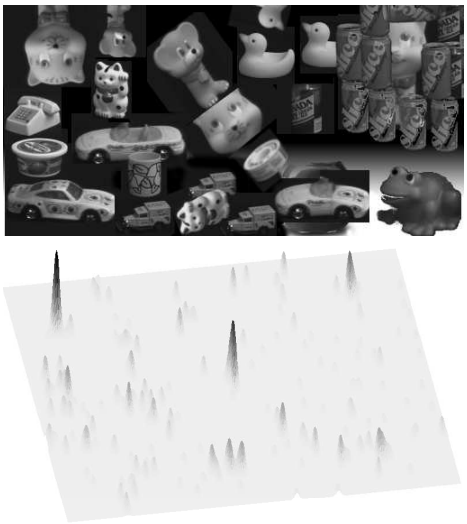

Fig. 4. Cluttered scene containing three instances of the object (top) and corresponding response of the detection process for the object model (bottom). The three major peaks observed in the density map correspond to the most probable object model locations.

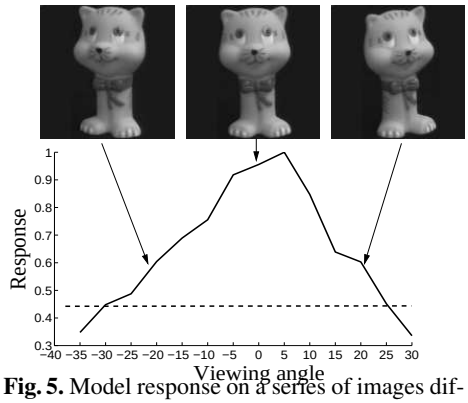

Fig. 5. Model response on a series of images differing in viewing angle by 5 degrees.

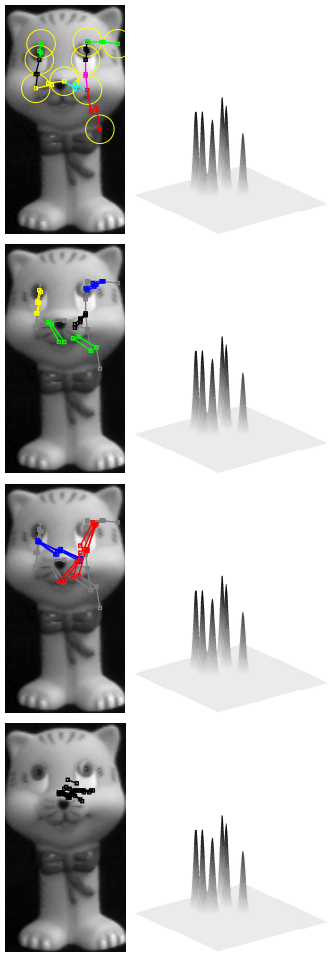

Fig. 6. Starting from the first level (top), the detection process uses the presence of primitives to infer the location of the higher level features. The sum of the density probability function for each feature of the level is shown on the right of the image.

9. J. Pearl. *Probabilistic reasoning in intelligent systems: Networks of plausible inference*. Morgan Kaufmann Publishers Inc., San Francisco, CA, 1988.
10. P. Perona, R. Fergus, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. In *CVPR*, volume 2, page 264, Madison, Wisconsin, June 2003.
11. J. H. Piater and R. A. Grupen. Distinctive features should be learned. In *BMCV*, pages 52–61, 2000.
12. M. Riesenhuber and T. Poggio. Hierarchical models of object recognition in cortex. In *Nature Neuroscience*, volume 2, pages 1019–1025, 1999.
13. G. Schwartz. Estimating the dimension of a model. *Ann. Stat.*, 6(2):461–464, 1978.
14. E. B. Sudderth, A. T. Ihler, W. T. Freeman, and A. S. Willsky. Nonparametric belief propagation. In *CVPR*, pages 605–612, 2003.
15. H. Wersing and E. Koerner. Unsupervised learning of combination features for hierarchical recognition models. In *ICANN*, pages 1225–1230, 2002.