Learning Grammatical Models for Object Recognition

Meg Aycinena Lippow[1], Leslie Pack Kaelbling[2], and Tomas Lozano-Perez[3]

[1] MIT, Computer Science and Artificial Intelligence Laboratory
32 Vassar Street 32-G585, Cambridge, MA 02139 USA
aycinena@csail.mit.edu
[2] MIT, Computer Science and Artificial Intelligence Laboratory
32 Vassar Street 32-G486, Cambridge, MA 02139 USA
lpk@csail.mit.edu
[3] MIT, Computer Science and Artificial Intelligence Laboratory
32 Vassar Street 32-G492, Cambridge, MA 02139 USA
tlp@csail.mit.edu

Abstract. In object recognition, the ability to share common parts or structure among related object classes allows information about parts and relationships in one class to be generalized to other classes. We present a recognition framework that uses probabilistic geometric grammars (PGGs) to capture structural variability and shared structure within and among object classes. We describe an efficient inference algorithm and a set of parameter and structure learning algorithms for PGGs, and demonstrate experimentally that the system provides a benefit in performance.

Keywords. grammatical inference, model selection and structure learning, computer vision

1 Introduction

Many current approaches to object recognition represent an object class as a collection of parts with some local appearance properties, and a model of the spatial relations among them. This representation is intuitive and attractive; object classes are often too variable to be described well using a single shape or appearance model, but they can be modeled as a distribution over a set of parts and the relationships among them.

Most of these systems, however, cannot share common parts or spatial structure among related object classes. This capability would allow information about parts and relationships in one object class to be generalized to other relevant classes. For example, we might like to transfer knowledge about the relationships among the arms and back of a chair to all chair classes with arms and backs, whether the base is composed of four legs or an axle and wheel-legs. We argue that modeling structural variability and shared part structure will allow effective learning from fewer examples and better generalization to unseen data.

Dagstuhl Seminar Proceedings 08091
Logic and Probability for Scene Interpretation
http://drops.dagstuhl.de/opus/volltexte/2008/1611

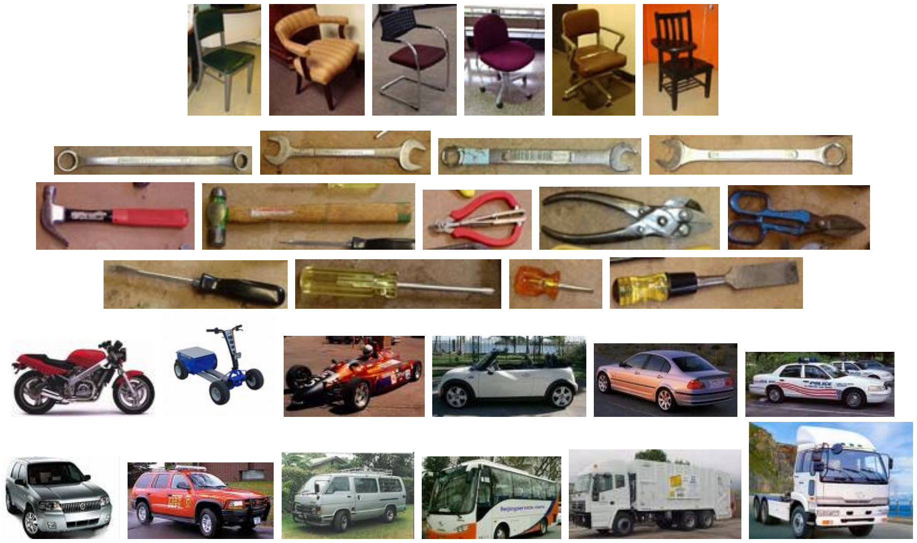


Fig. 1. Many object classes exhibit structural variability.

With these goals in mind, we present a recognition framework that captures structural variability within and among object classes (see Figure 1). We introduce *probabilistic geometric grammars* (PGGs), which represent object classes recursively in terms of their parts. PGGs extend probabilistic context-free grammars (PCFGs) for language, supplementing the traditional PCFG representation with models for the geometry and appearance of object parts. Context-free grammars capture structural variability by compactly modeling hierarchical groups and substitution of subparts, and they naturally represent conditional independences between subgroups with the context-free assumption. Probabilistic grammars further model distributions over the combination of subparts.

2 Related Work

The PGG framework is inspired by recent object class models that use a collection of parts, an appearance model for each part, and a statistical model of the geometric relations among the parts. This approach has existed in the literature for decades [1], but there has been an increase of activity in this area in the last few years; e.g., the constellation or star model [2,3], and statistical pictorial structures or k-fans [4,5,6].

In contrast to these approaches, in which each object class consists of an unstructured set of parts, the PGG model uses a fundamentally hierarchical notion of object and part. The use of part hierarchies to model object classes and enable parts sharing among classes has become increasingly popular in recent years, and

chair:
1.0 chair → top (ϕ_{000}) base (ϕ_{001})
top:
0.55 top → seat (ϕ_{100}) back (ϕ_{101})
0.45 top → seat (ϕ_{110}) back (ϕ_{111}) arm (ϕ_{112}) arm (ϕ_{113})
base:
0.65 base → leg (ϕ_{200}) leg (ϕ_{201}) leg (ϕ_{202}) leg (ϕ_{203})
0.35 base → axle (ϕ_{210}) wheel-leg (ϕ_{211}) wheel-leg (ϕ_{212}) wheel-leg (ϕ_{213})

seat (A_3) arm (A_5) axle (A_7)
back (A_4) leg (A_6) wheel-leg (A_8)

Fig. 2. A textual description of a PGG for chairs.

researchers have shown it can improve the learning rate and recognition accuracy [7,8,9].

The PGG model, while similar in spirit to these hierarchical approaches, differs fundamentally in that it allows choice ("OR") nodes in addition to the "AND" nodes that exist in simple part hierarchies; this difference is what makes it a grammar. Most recent uses of grammars in vision have focused on modeling the segmentation of entire images, rather than object classes, or on detecting mid-level visual objects, such as curves and rectangles [10,11,12]. The work of Zhu, Chen, and Yuille [13] is a notable exception; our approach contrasts with theirs in that our model exploits stronger conditional independence assumptions, allowing simple and robust algorithms for recognition and grammar learning. Finally, our structure learning operators and score resemble those in many approaches to grammar learning for language [14,15].

3 The PGG Model

The probabilistic geometric grammar model augments traditional PCFGs with geometry and appearance models. A PGG is a set of part models, each of which is either primitive or composite (similar to terminals and nonterminals in PCFGs). A composite part model consists of a set of rules which define how the part can be broken down into subparts. A primitive part model consists of an appearance model which describes a distribution over the part's image appearance. Figure 2 shows an example of a PGG for chairs.

An image can be broken down into a finite set of windows or regions. For each part model c and each window w in the image, we define a binary random variable X_{cw} denoting that window w contains an object or part of type c.

3.1 Rules

Each rule r in a composite part model c defines one way that the part can be composed of subparts. A rule consists of an expansion probability $\gamma_{cr} \in [0, 1]$, and a set of rule parts. The expansion probabilities must sum to 1 for a fixed parent part c. Thus, the set of rules defines a distribution over the choice of ways to expand the part—an "OR" node—while each rule defines an "AND" relationship over its rule parts.

Each child rule part k in rule r of parent part model c has two components: an index d_{crk} which refers to another part model in the grammar, and a conditional geometry model ϕ_{crk}, which defines a distribution on the geometric properties of this subpart given those of its parent part (of type c). We assume that the geometric and image properties of subparts are conditionally independent given the properties of the parent c.

One way to interpret a rule is that it expresses a compositional relationship; it states that a part with class c can be composed of a set of subparts, where the kth subpart in rule r has class d_{crk}. More generally, composite part models may be viewed as hidden variables that represent geometric information upon which their child parts depend. In either of these cases, however, composite parts do not directly model pixels in the image—only primitive parts do this.

We can also think of the expansion probabilities as defining a distribution over Bayes nets. Each internal node in each Bayes net represents the geometry of a composite object part, while each leaf node represents the geometry *and* appearance of a primitive part.

3.2 Conditional Geometry Models

Each rule part k has a geometry model ϕ_{crk}, which models a conditional distribution over the geometric attributes of the kth child part, given the attributes of the parent part. The attributes over which the models are defined may be anything: location, scale, orientation, shape, etc. In this paper, we model only the relative location of part centroids, defining a Gaussian over the position of a child v relative to its parent w:

$$P(v|w;\phi_{crk}) = \mathcal{N}(x_v - x_w, y_v - y_w; \boldsymbol{\mu}_{crk}, \boldsymbol{\Sigma}_{crk}) \ .$$

To avoid overfitting, we use a diagonal covariance. Despite this simple model, multimodal distributions can be handled using multiple rules with the same symbols but different geometry models, effectively yielding Gaussian mixtures.

3.3 Appearance Models

Each primitive part model c has an appearance model A_c which defines the appearance or material properties of the part. The PGG formulation is modular with respect to the representation for the appearance model; the only requirement is that the model enable the calculation of the image likelihood ratio $P(I_w|X_{cw})/P(I_w)$: the ratio of the likelihood of the image pixels I_w in window w, given that a primitive part of type c occupies that window, to the unconditional image likelihood $P(I_w)$. Using the ratio ensures that every image pixel that is not assigned to a specific primitive part is evaluated according to the unconditional model. This allows us to compare object detections occupying different numbers of pixels.

4 Efficient Recognition with PGGs

In this section, we present a top-down dynamic programming algorithm for object classification and localization in an image. It extends the pictorial structures algorithm [4] to hierarchical part models and choice nodes. The algorithm depends on a discretization of the image into a finite set of locations or regions, so it can recursively calculate a score for each region w and part model c while caching and reusing intermediate results.

Two assumptions will enable our derivation: that primitive parts explain different parts of the image, and that subparts are conditionally independent given their parent. If the primitive parts overlap, then it is approximate. But we do not assume the union of subparts equals the parent part, since the background model explains pixels not assigned to a primitive part.

Given an image with pixels or features I, we frame the recognition problem as finding the root part model c and window w that maximizes $P(X_{cw}|I)$. We apply Bayes rule, remove the constant term $P(I)$, partition the features into those inside and outside w, and evaluate the features $I_{\overline{w}}$ not in w with a background model:

$$\operatorname*{argmax}_{c,w} P(X_{cw}|I) = \operatorname*{argmax}_{c,w} P(I_w|X_{cw})P(I_{\overline{w}})P(X_{cw})$$

We make the common assumption that foreground and background pixels are independent, so that $P(I_{\overline{w}}) = P(I)/P(I_w)$, and also that $P(X_{cw})$ is uniform so that all object classes and locations are equally likely (we could also naturally incorporate a contextual prior).

$$= \operatorname*{argmax}_{c,w} \frac{P(I_w|X_{cw})}{P(I_w)}$$

We can recursively decompose the image likelihood ratio according to the grammar. Sum over all rules r for part c, and let X_{crw} denote that window w is occupied by a part of type c and expanded by rule r:

$$\beta(c,w) = \frac{P(I_w|X_{cw})}{P(I_w)} = \sum_r P(r|c)\frac{P(I_w|X_{crw})}{P(I_w)}$$

We must consider the unknown geometry of the child parts in r; let $\mathbf{v}$ be a vector specifying their locations:

$$= \sum_r P(r|c)\frac{1}{P(I_w)}\sum_{\mathbf{v}} P(I_w|\mathbf{v}, X_{crw})P(\mathbf{v}|X_{crw})$$

Partition I_w into I_{v_k}, the pixels in child region v_k, and $I_{w-\mathbf{v}}$, the pixels in w but not in any region v_k, and assume conditional independence of the children given the parent. Let d_{crk} denote the part model referred to by the kth rule part, and ϕ_{crk} be its geometry model.

$$= \sum_r P(r|c)\frac{1}{P(I_w)}\sum_{\mathbf{v}} P(I_{w-\mathbf{v}})\prod_k P(I_{v_k}|X_{d_{crk}v_k})P(v_k|w;\phi_{crk})$$

Due to our independence assumptions, we have that $P(I_{w-\mathbf{v}}) = \frac{P(I_w)}{\prod_k P(I_{v_k})}$, so we can substitute and cancel:

$$
\begin{aligned}
&= \sum_r P(r|c) \sum_{\mathbf{v}} \prod_k \frac{P(I_{v_k}|X_{d_{crk},v_k})}{P(I_{v_k})} P(v_k|w;\phi_{crk}) \\
&= \sum_r P(r|c) \prod_k \sum_v \beta(d_{crk}, v) P(v|w;\phi_{crk}) \qquad (1)
\end{aligned}
$$

The result is a recursive expression for the likelihood ratio, where $P(r|c)$ is the rule probability γ_{cr}, $\beta(d_{crk}, v)$ is the recursive likelihood ratio for the kth child part, and $P(v|w;\phi_{crk})$ is the likelihood of the geometry of v conditioned on the attributes of w. The base case occurs when c is primitive: $\beta(c, w)$ is defined directly by the appearance model for part c.

Equation 1 leads to a top-down algorithm that recursively calculates a score for each region and part model, caching intermediate results. It has complexity $O(|G||I|^2)$, where $|G|$ is the number of part models, rules, and rule parts in the grammar, and $|I|$ is the number of image regions. In practice, we limit the sum over child regions v to those near the expected location (we use three standard deviations around the mean), greatly reducing the effect of the squared term.

It is crucial that we not approximate the sums over r and v in Equation 1 with max, although this would enable the use of the distance transform. This would result in scoring individual parse trees, and we cannot compare the scores of two parse trees that have different numbers of parts or edges because an unequal number of terms is contributing to the likelihood function in each case. To control for the structural difference among trees, we sum them out entirely.

5 Parameter Learning in PGGs

We assume for now a fixed grammar structure, and develop an EM algorithm to estimate its parameters from data, extending the standard inside-outside algorithm for PCFGs. We have a set of training images $\{I^i | i = 1 \ldots N\}$ labeled with root bounding boxes u^i and root object classes ρ^i. Let I^i_w be the image pixels in region w of the ith training image. The internal tree structure and geometry of each object is not labeled.

The parameters Θ are the rule probabilities γ_{cr} and the geometry model parameters $(\boldsymbol{\mu}_{crk}, \boldsymbol{\Sigma}_{crk})$. In this paper, we will not address learning the appearance models A_c, assuming a fixed vocabulary of primitive part detectors. We need to estimate the parameters Θ' given the parameters Θ from the previous iteration.

E-step We need to calculate the likelihood of the hidden variables X_{cw}, X_{crw}, and $X_{d_{crk}v}$; these responsibilities will be used to reestimate the parameters.

In the inside-outside algorithm for PCFGs, the *inside probability* is the likelihood that a substring was generated by a nonterminal, summing out all possible parse trees. The analogous quantity in our context is $P(I_w|X_{cw})$, but because the PGG framework actually models the image likelihood ratio $P(I_w|X_{cw})/P(I_w)$, we shall use the notion of the inside probability *ratio* $\beta(c, w)$ instead, which we derived in Equation 1.

In PCFGs, the *outside probability* is the total likelihood of seeing the symbols that are on either side of a substring and a nonterminal covering the substring. The analogous quantity for us is $P(I_{\overline{w}}, X_{cw})$, the likelihood of seeing the pixels $I_{\overline{w}}$ *outside* the window w and the part model c in window w. Again, we use the outside probability *ratio* instead, which we can define recursively (derivation in our technical report):

$$\begin{aligned}\alpha(c, w) &= \frac{P(I_{\overline{w}}, X_{cw})}{P(I_{\overline{w}})} \\ &= \sum_{c',w'} \alpha(c', w') \sum_{r'} \gamma_{c'r'} \sum_{\substack{k \text{ s.t.} \\ d_{c'r'k}=c}} P(w|w'; \phi_{c'r'k}) \prod_{k' \neq k} \sum_{v} \beta(d_{c'r'k'}, v) P(v|w'; \phi_{c'r'k'})\end{aligned}$$

The base case occurs when c is the labeled object class and w is the labeled bounding box, when $\alpha(c, w) = 1.0$.

Let α_i and β_i be the inside and outside probability ratios applied to the ith training image. Let $X_{\rho^i u^i}$ denote that the labeled object class ρ^i occupies the labeled bounding box u^i in image I^i. Then the responsibilities are given by:

$$\begin{aligned}f_i(c, w) &= P(X_{cw}|I^i, X_{\rho^i u^i}) = \frac{\alpha_i(c, w)\beta_i(c, w)}{\beta_i(\rho^i, u^i)} \\ g_i(c, r, w) &= P(X_{crw}|I^i, X_{\rho^i u^i}) \\ &= \frac{1}{\beta_i(\rho^i, u^i)} \alpha_i(c, w)\gamma_{cr} \prod_{k} \sum_{v} \beta_i(d_{crk}, v) P(v|w; \phi_{crk}) \\ h_i(c, r, k, w, v) &= P(X_{crw}, X_{d_{crk}v}|I^i, X_{\rho^i u^i}) \\ &= \frac{1}{\beta_i(\rho^i, u^i)} \alpha_i(c, w)\gamma_{cr}\beta_i(d_{crk}, v) P(v|w; \phi_{crk}) \\ &\quad \times \prod_{k' \neq k} \sum_{v'} \beta_i(d_{crk'}, v') P(v'|w; \phi_{crk'})\end{aligned}$$

M-step Now we can reestimate the parameters Θ' for the next iteration using the responsibilities:

$$\begin{aligned}\gamma'_{cr} &= \frac{\sum_i \sum_w g_i(c, r, w)}{\sum_i \sum_w f_i(c, w)} \\ \boldsymbol{\mu}'_{crk} &= \frac{\sum_i \sum_w \sum_v h_i(c, r, k, w, v) T(v, w)}{\sum_i \sum_w \sum_v h_i(c, r, k, w, v)} \\ \boldsymbol{\Sigma}'_{crk} &= \frac{\sum_i \sum_w \sum_v h_i(c, r, k, w, v)(T(v, w) - \boldsymbol{\mu}'_{crk})^2}{\sum_i \sum_w \sum_v h_i(c, r, k, w, v)}\end{aligned}$$

where $T(v, w)$ transforms the child region v by subtracting the centroid of the parent region w.

Given initial parameters Θ_0, we iterate the E- and M-steps until the difference in the log likelihood scores of the data under Θ and Θ' is no greater than 0.01 of the log likelihood score under Θ.

6 Structure Learning in PGGs

Structure learning aims to find a compact grammar that explains the training data. By targeting compactness, we encourage sharing of parts and substructure among object classes. Here we describe a search-based optimization approach to structure learning.

6.1 Search Initialization

In this paper, we assume a pre-specified set of primitive parts, with appearance models. We also assume the labeled initial locations of a set of primitive parts making up each training object, although these are free to change during EM.

For each unique pattern of labeled primitive parts in the training data, we write down a rule with the labeled object class on the left side of the rule and the primitive parts on the right side. To initialize the geometry models ϕ_{crk}, we estimate the mean and variance of the primitive part positions relative to the bounding boxes' centroids, across training images with the same object class and set of primitive labels.

6.2 Structure Search Operators

As in other approaches to grammar learning, our search operators move the algorithm through the space of candidate grammars by proposing changes to the current grammar. We use four types of operators.

Create a new AND composite part. The role of this operator is to recognize common subsets of rule parts, and create new composite parts to stand for these patterns. A new AND part $\mathrm{C}_{\mathrm{and}}$ may be proposed whenever a pattern of rule parts ξ with size no greater than n_{and} occurs on the right side of at least two rules (we use $n_{\mathrm{and}} = 3$). For example:

$$\begin{array}{lcl} \mathrm{C1} \rightarrow \mathrm{X1}\ \mathbf{X2}\ \mathbf{X3}\ \mathrm{X4} & & \mathrm{C1} \rightarrow \mathrm{X1}\ \mathbf{C_{and}}\ \mathrm{X4} \\ \mathrm{C2} \rightarrow \mathbf{X2}\ \mathbf{X3}\ \mathrm{X5} & \Rightarrow & \mathrm{C2} \rightarrow \mathbf{C_{and}}\ \mathrm{X5} \\ & & \mathbf{C_{and}} \rightarrow \mathbf{X2}\ \mathbf{X3} \end{array}$$

The initial geometry parameters for $\mathrm{C}_{\mathrm{and}}$ are a weighted average of the transformed parameters[1] of the instances of the pattern ξ that contributed to its

[1] We must transform the geometry models to be relative to a new local parent (we use the centroid of the selected parts), so the models will be invariant to their positions in the context of their original rule. Then, to average a set of geometry models with parameters (μ_i, σ_i^2) and weights γ_i to produce the initial model for a merged rule part, we use $\mu = \sum_i \gamma_i \mu_i$ for the mean and $\sigma^2 = \sum_i \gamma_i \big(\sigma_i^2 + (\mu_i - \mu)^2\big)$ for the variance. Because EM is notoriously sensitive to initialization, it is important that we are able to choose reasonable initial values for these parameters.

creation. The initial mean for each replaced instance of $\mathrm{C_{and}}$ in the old rules is the centroid of the instance of ξ which was replaced; the initial variance parameter in each dimension is the average of the variances of the component parts that were replaced.

Create a new OR composite part. This operator plays the opposite role: it notices *differences* among sets of rules, and creates composite parts to more compactly express those differences. A new OR composite part $\mathrm{C_{or}}$ may be proposed whenever at least two rules would become identical were a pair or small subset of part models $(\mathrm{X1}, \mathrm{X2}, ...)$, to be renamed to $\mathrm{C_{or}}$ in the context of those rules. We search for sets of symbols that are in common among the rules with size no greater than n_{or}, *or* sets that are different among the rules with size no greater than n_{or} (we use $n_{\mathrm{or}} = 3$).

$$\begin{array}{lcl} \mathrm{C1} \rightarrow \mathrm{X1\ X2}\ \mathbf{X3} & & \mathrm{C1} \rightarrow \mathrm{X1\ X2}\ \mathbf{C_{or}} \\ \mathrm{C1} \rightarrow \mathrm{X1\ X2}\ \mathbf{X4} & \Rightarrow & \mathbf{C_{or} \rightarrow X3} \\ & & \mathbf{C_{or} \rightarrow X4} \end{array}$$

The initial geometry parameters of the merged rule are again an average of those of the contributing rules, weighted by their rule probabilities. The initial probabilities on the new rules for $\mathrm{C_{or}}$ are a renormalized version of the contributing rule probabilities. One rule in the new part may be entirely empty, expressing the notion of an optional set of parts, such as chair arms.

Apply an existing AND or OR composite part. The creation operators we just defined need not be applied immediately to all applicable rules. Thus, we have operators to apply existing composite part models rather than creating new ones.

An existing AND part $\mathrm{C_{and}}$ with a single rule $\mathrm{C_{and}} \rightarrow \xi$ may be applied whenever the pattern ξ occurs on the right side of at least one other rule.

$$\begin{array}{lcl} \mathrm{C1} \rightarrow \mathbf{X1\ X2}\ \mathrm{X3} & \Rightarrow & \mathrm{C1} \rightarrow \mathbf{C_{and}}\ \mathrm{X3} \\ \mathbf{C_{and} \rightarrow X1\ X2} & & \mathbf{C_{and} \rightarrow X1\ X2} \end{array}$$

An existing OR composite part $\mathrm{C_{or}}$ with rule patterns $(\xi_1, \xi_2, ...)$ may be applied whenever at least two rules would become identical were the instances of the patterns $(\xi_1, \xi_2, ...)$ in those rules renamed to $\mathrm{C_{or}}$.

$$\begin{array}{lcl} \mathrm{C2} \rightarrow \mathrm{X1\ X2}\ \mathbf{X3} & & \mathrm{C2} \rightarrow \mathrm{X1\ X2}\ \mathbf{C_{or}} \\ \mathrm{C2} \rightarrow \mathrm{X1\ X2}\ \mathbf{X4} & \Rightarrow & \mathbf{C_{or} \rightarrow X3} \\ \mathbf{C_{or} \rightarrow X3} & & \mathbf{C_{or} \rightarrow X4} \\ \mathbf{C_{or} \rightarrow X4} & & \end{array}$$

6.3 Structure Score and Search Control

The structure score evaluates a structure G given the image data D. The score we use is a simple combination of the quality of the training data under the model and a penalty on the model's complexity:

$$\mathrm{score}(G; D) = (1 - \lambda)\ell(D; G) - \lambda \frac{\log N}{2} \dim(G)$$

where $\ell(D;G)$ is the log likelihood ratio of the training data given the model G, N is the number of training images, $\dim(G)$ is the number of parameters in G, and λ trades off between the likelihood and the model complexity.[2] We use $\lambda = 0.25$ (determined empirically).

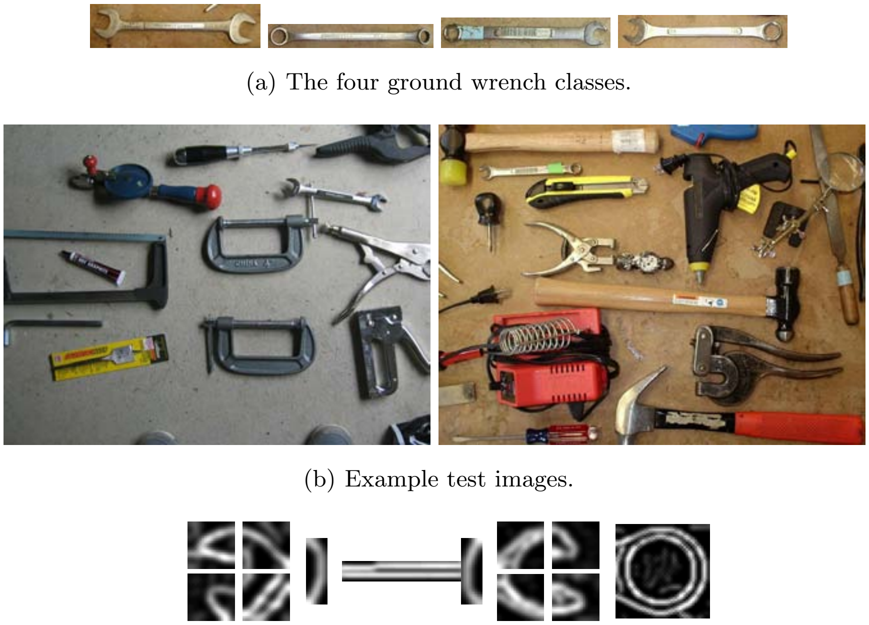
The branching factor for this search problem is quite high. However, we can apply an insight about our structure score: a grammar's score before EM has been run is a lower bound on its score after EM. We can exploit this property to find a proposal that is guaranteed to be an uphill step, rather than searching exhaustively for the maximal gradient proposal. Specifically, rather than running EM on each candidate grammar and choosing the operation with the best post-EM score, we can rank the proposals according to their pre-EM scores, run EM on the grammars in ranked order, and accept the first proposal whose post-EM score improves on the current score.

Furthermore, we can use the guiding principle of compactness to inspire several greedy search heuristics, which will bias us towards desirable structures while controlling the branching factor. First, to encourage compact structure, we always consider applying existing AND and OR parts before creating new ones, and we collapse unnecessary hierarchy by inlining part models that have a single rule and are referred to only once in the rest of the grammar.

Second, because we want to encourage low variance geometry distributions, we bias the search towards geometrically plausible proposals. Although the number of proposals at each search step is usually large, most proposals will lower the overall structure score because they will be merging parts and rules that are not geometrically compatible. With this in mind, we prune proposals that result in high variance geometry models (in this paper, any model that has a standard deviation greater than 0.2 of the corresponding dimension—width or height—of the objects in the training data).

At each step there may be multiple lexicographically identical operations (e.g., creating an AND part for a pattern that occurs three or more times). So, again to encourage low variance distributions, we only propose the single operation from the set of identical operations whose geometry models match best, and for efficiency we only consider *pairs* of rules or rule part subsets at any given time. In order to compare the geometry for two sets of rule parts, we can impose a canonical ordering on parts. There may still be ambiguity about how to match up rule parts if there is more than one part of the same type in each of the rules (e.g., two chair legs), but because we expect the number of such parts to be quite small, we can enumerate all the possible ways to assign the rule parts in one rule to the other. Then, for each fixed assignment, we transform the geometry models to be relative to a new local parent, and use symmetrized KL divergence as a distance metric to choose the operation that would merge parts with the best matching geometry models.

[2] Although this formula resembles the BIC score, we are not actually using a Bayesian approach to construct our structure score.



(a) The four ground wrench classes.

(b) Example test images.

(c) Hand-chosen edge intensity templates. From top to bottom and left to right: tip3, tip4, side3, side4, open-left, dedge, open-right, side2, side1, tip2, tip1, full-circle.

Fig. 3. A wrench domain.

7 Experimental Results

We have collected a wrench data set to use as a test bed for the framework. It consists of four ground classes of wrench, shown in Figure 3(a). This domain has inherent "or" structure that makes it a natural place to start testing the PGG model.

We focus on a localization task: given a large complicated image that contains a single wrench but also many distractor items, the goal is to correctly localize the wrench. We are not yet modeling or searching over scale or orientation, so the images have been rotated and scaled so that the wrench is horizontal and of roughly uniform width, although there is some variation. Figure 3(b) shows example test images.

In this paper, we use hand-chosen edge intensity templates (shown in Figure 3(c)) for our appearance models, and define the image likelihood ratio as follows:[3]

$$\frac{P(I_w|X_{cw})}{P(I_w)} = \frac{f(cc(I_w, T_c))^q}{\sum_{I'_w} P(I'_w) f(cc(I'_w, T_c))^q}$$

[3] The right expression is not in terms of distributions; our technical report justifies this approximation in detail.

```
wrench:
  1.0     wrench → C0001 C0002 dedge dedge
C0001:
  0.9752    C0001 → open-right side1 side2 tip1 tip2
  0.0248    C0001 → full-circle
C0002:
  0.0174    C0002 → full-circle
  0.9826    C0002 → open-left side3 side4 tip3 tip4
```

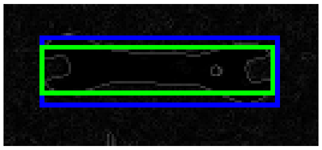


Fig. 4. On the left, a typical learned grammar for wrenches. On the right, an example of 68% overlap. The blue (outer) box is the correct labeled bounding box, while the green (inner) box is the prediction.

where cc denotes (unnormalized) cross correlation between the image patch I_w and the template T_c, f normalizes the correlation coefficient to be in $[0, 1]$, and the exponent q serves to strengthen strong responses and suppress weak ones (a similar approach to that of Torralba et al. [9]). We use $q = 6$ for all templates except the full-circle wrench end; because of its larger area, we found that $q = 8$ worked better. We can estimate the denominator by sampling over training set foreground and background patches and computing the expected correlation response. The ratio will be greater than 1 in cases where the template response is better than "average", and less than 1 otherwise.

Figure 4 *(left)* shows a typical learned grammar in the wrench domain. The structure makes sense: a wrench consists of a right end (C0001), a left and (C0002), and two horizontal bars; each end can be closed or open. However, the rule probabilities for the wrench ends strongly prefer the "open" choice; we have learned that, given our fixed set of appearance models, all four types of wrenches can be explained well by a model of open-open wrenches. Despite this surprising result, the model performs impressively. We expect that learning the primitive appearance models so that they span the representational space and avoid redundancy would result in even better structures and results.

7.1 Comparison With Simpler Models

To evaluate whether the PGG framework provides a performance benefit, we compared the full PGG model against simpler versions of the model:

- A set of grammars, one for each wrench type, where each grammar has a single flat rule.[4]
- A single grammar with a set of flat rules, one for each wrench type.

The first might be considered the simplest derivative of the PGG model. The second ensures that any benefit achieved by the full model over the first baseline

[4] A flat rule has the object class on the left hand side and the set of primitive parts on the right.

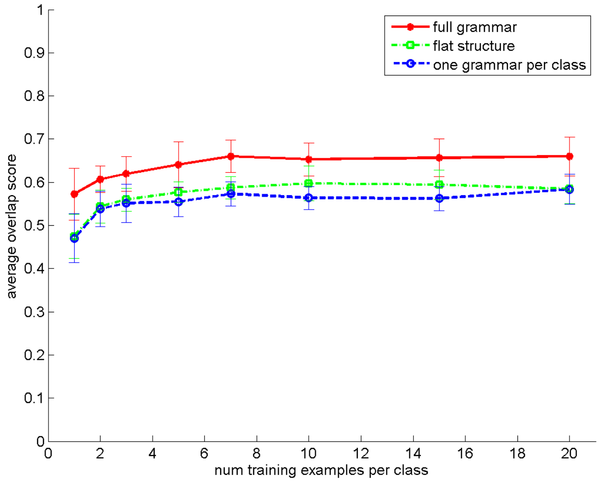


Fig. 5. A comparison of the full PGG model against simpler versions of the model. Because there are four ground classes, the total training data is four times each x-axis value. Error bars represent 95% confidence intervals.

is not due to tweaking the rule probabilities or geometry parameters during EM, but rather due to some aspect of the structure learning: the hierarchy introduced by building up structure, or the sharing of common substructure among different types of wrenches.

Our performance metric is the percentage overlap between the predicted and labeled bounding boxes: the ratio of the area of intersection of the windows to the area of their union. This will be one when the windows are identical and zero when they do not overlap at all.

We trained each model on training sets of increasing size, and tested the learned models on a set of 40 images (10 of each wrench type). We report the mean percentage overlap score on the test set for each training set size. We repeated this procedure 10 times for different splits of training and test data, and averaged the resulting curves together to produce Figure 5.

Reassuringly, the full PGG model enjoys a significant advantage over the two baseline models. Furthermore, the flat structure slightly outperforms the one-grammar-per-class approach, but not significantly so for most training set sizes. Therefore, we can feel confident that the structure learning process is responsible for most of the advantage of the full PGG model over the simplest model.

To put the scores shown in Figure 5 in perspective, Figure 4 *(right)* 68% overlap looks like in an image—it is quite a high localization score.

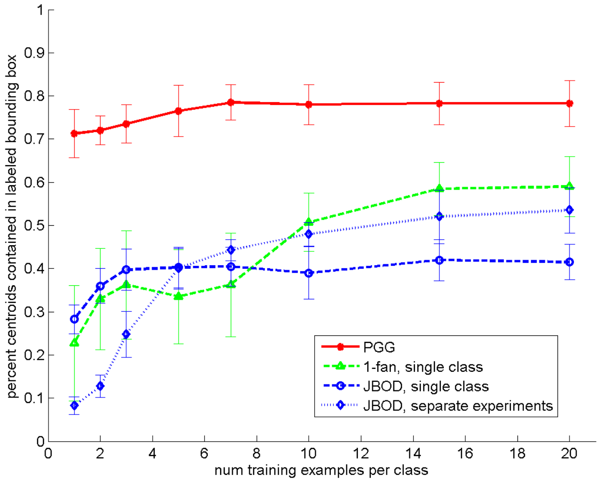


Fig. 6. A comparison of the PGG model against other leading object recognition systems [6,9]. Because there are four ground classes, the total training data is four times each x-axis value. Error bars represent 95% confidence intervals.

7.2 Comparison With Other Approaches

It is important to validate our overall approach through comparisons with other leading object recognition systems. Therefore, we also compared the full PGG model against the following systems:

- The 1-fan object detector [6], treating all wrench types as one object class.
- Two variations of the joint boost object detector (JBOD) [9]: treating all wrench types as one object class, and running a separate detection experiment for each wrench type and averaging the results.[5]

In the first case, we used the authors' published implementation, while in the second we used our own reimplementation. We used the same values for all experimental parameters as were reported in the publications. We also controlled for object scale and orientation.

The methods to which we compare do not predict tight bounding boxes, so we cannot usefully measure localization performance using the percentage overlap metric. Instead, we measure the percentage of times that the predicted object centroid falls within the labeled bounding box, across the test set. Otherwise, we followed a similar experimental procedure as described in Section 7.1. The results are shown in Figure 6.

These experiments demonstrate that the localization task on the wrench data is a challenging one. Figure 6 shows that the PGG model outperforms the other

[5] We also tried treating each wrench type as its own object class; the results were similar to the one-class case.

systems by a significant margin on this dataset. It seems that, especially for small training set sizes, this task is indeed nontrivial; therefore we may conclude that the good performance of the PGG model is promising.

Nonetheless, it is important to point out that both of the systems we compared against learn their appearance models, while we assume a pre-specified set of models. Although it may seem that our approach has an advantage due to the extra supervision, several recent authors have shown that choosing primitive parts automatically based on data outperforms using hand-chosen primitive parts (notably, Crandall and Huttenlocher [6]). Thus, it seems possible that learning our appearance models would result in even better performance, and this is an important area of future work.

References

1. Fischler, M., Elschlager, R.: The representation and matching of pictorial structures. IEEE Transactions on Computers **C-22** (1973)
2. Fergus, R., Perona, P., Zisserman, A.: Object class recognition by unsupervised scale-invariant learning. In: CVPR. (2003)
3. Fergus, R., Perona, P., Zisserman, A.: A sparse object category model for efficient learning and exhaustive recognition. In: CVPR. (2005)
4. Felzenszwalb, P.F., Huttenlocher, D.P.: Pictorial structures for object recognition. IJCV **61** (2005)
5. Crandall, D., Felzenszwalb, P., Huttenlocher, D.: Spatial priors for part-based recognition using statistical models. In: CVPR. (2005)
6. Crandall, D.J., Huttenlocher, D.P.: Weakly supervised learning of part-based spatial models for visual object recognition. In: ECCV. (2006)
7. Sudderth, E.B., Torralba, A., Freeman, W.T., Willsky, A.S.: Learning hierarchical models of scenes, objects, and parts. In: ICCV. (2005)
8. Ullman, S., Epshtein, B.: Visual classification by a hierarchy of extended fragments. In: Towards Category-Level Object Recognition, Lecture Notes on Computer Science, Springer-Verlag (2006)
9. Torralba, A., Murphy, K.P., Freeman, W.T.: Sharing visual features for multiclass and multiview object detection. PAMI **29** (2007)
10. Tu, Z., Chen, X., Yuille, A.L., Zhu, S.C.: Image parsing: Unifying segmentation, detection, and recognition. IJCV **63** (2005)
11. Tu, Z., Zhu, S.C.: Parsing images into regions, curves, and curve groups. IJCV **69** (2006)
12. Siskind, J., Sherman, Jr., J., Pollak, I., Harper, M., Bouman, C.: Spatial random trees grammars for modeling hierarchical structure in images with regions of arbitrary shape. PAMI **29** (2007)
13. Zhu, L.L., Chen, Y., Yuille, A.: Unsupervised learning of a probabilistic grammar for object detection and parsing. In: NIPS. (2006)
14. de Marcken, C.G.: Unsupervised Language Acquisition. PhD thesis, Massachusetts Institute of Technology (1996)
15. Nevill-Manning, C.G., Witten, I.H.: Identifying hierarchical structure in sequences: A linear-time algorithm. JAIR **7** (1997)