in Proc. IEEE Computer Vision Pattern Recognition (CVPR), vol. 1, pp. 927-934, New York, NY, June 2006

Extracting Subimages of an Unknown Category from a Set of Images

Sinisa Todorovic and Narendra Ahuja
Beckman Institute for Advanced Science and Technology
University of Illinois at Urbana-Champaign, U.S.A.
{sintod, ahuja}@vision.ai.uiuc.edu

Abstract

Suppose a set of images contains frequent occurrences of objects from an unknown category. This paper is aimed at simultaneously solving the following related problems: (1) unsupervised identification of photometric, geometric, and topological (mutual containment) properties of multiscale regions defining objects in the category; (2) learning a region-based structural model of the category in terms of these properties from a set of training images; and (3) segmentation and recognition of objects from the category in new images. To this end, each image is represented by a tree that captures a multiscale image segmentation. The trees are matched to find the maximally matching subtrees across the set, the existence of which is itself viewed as evidence that a category is indeed present. The matched subtrees are fused into a canonical tree, which represents the learned model of the category. Recognition of objects in a new image and image segmentation delineating all object parts are achieved simultaneously by finding matches of the model with subtrees of the new image. Experimental comparison with state-of-the-art methods shows that the proposed approach has similar recognition and superior localization performance while it uses fewer training examples.

1. Introduction

Suppose we are given a set of images in which objects having a canonical photometric and geometric structure, i.e., belonging to a category, occur frequently. By geometric structure here we mean the layout, and intrinsic and relative geometric properties of image regions that comprise an object, and by photometric structure we mean the intrinsic and relative properties of intensity or color variation that characterize the object regions. Whether, and where, any objects from the category occur in a specific image is not known. We are interested in discovering whether a category does indeed occur in the image set, and if yes, in obtaining a compact model of the canonical geometric and photometric structure defining the category. A model derived from such training can then be used to determine whether a new test image contains objects from the category, and when it does, to segment all instances of the category in the image by precisely delineating all defining regions of each instance.

Our approach involves the following major steps. (1) Images are represented as multiscale segmentation trees, obtained by using a segmentation algorithm discussed in [2, 15]. Nodes in the segmentation tree at upper levels correspond to more salient regions, while their children nodes capture less salient details, e.g., more homogeneous subregions with smaller intensity contrasts, as illustrated in Fig. 1. The segmentation tree may be highly unbalanced, with nonleaf nodes having varying numbers of children and leaf nodes occurring at multiple levels, as dictated by the image topology. Any cutset of the tree partitions the image, thus providing a two-dimensional layout of the regions. The parent-child relationship yields an additional dimension, that of scale. The resulting three-dimensional structure serves as a rich description of the image for deriving category models. (2) The segmentation tree involves specification of geometric properties of segmented regions, including region area, boundary shape, relative distances between regions, and photometric properties including gray level, and gray-level variance of regions. If a category occurs, we expect that subimages with category specific values of the above properties will be abundant. Each such subimage will correspond to one or more subtrees in the segmentation tree, thus leading to frequent occurrences of subtrees with similar properties. These subtrees are detected by matching segmentation trees of the image set, where the matching algorithm minimizes a cost defined in terms of the above properties. The result is a set of subtrees from each image that have cross-image matching cost below a minimum level. Their existence indicates that a category is present. (3) The tree matching algorithm identifies exactly which properties are shared by the matching subtrees. These properties and the subtree structure are then used to define the category model. The model thus specifies: how segmented regions are recursively laid out to comprise an object from the category, and what their intrinsic and rela-

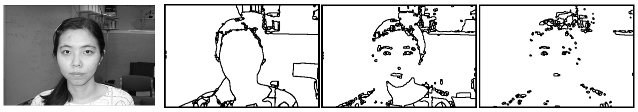
(a) Input (b) Tree levels: 2, 3, 4

Figure 1. Segmentation tree; regions (nodes) at a specific level projected onto the image plane; the root is at level 1; regions at level 3 are contained within those at level 2, etc.

tive geometric and photometric properties are. (4) When a new image is encountered, its segmentation tree is matched against the category tree. If one or more matches are found with sufficiently low costs, the matching subtrees specify the exact boundaries of the recognized objects, and their constituent regions.

2. Relationship to Prior Work

The problem of unsupervised category modeling has been attracting increasing interest in recent years. Almost entire work in this area models a category as a statistical distribution of appearances and relative locations of point feature descriptors and curve fragments, here both referred to in the sequel as *local features*, [6, 7, 8, 10, 1, 14]. In contrast, as the basic vocabulary of a category model, we use segmented regions, which are less sensitive to lighting, scale, and viewpoint changes than local features. Also, segments provide for richer geometric and photometric descriptions of perceptually meaningful object parts. All this allows extraction of more robust and reliable structural category models.

Majority of the existing approaches to object categorization define object detection as *image classification* [1, 7, 6, 14]. They typically suffer from detecting multiple occurrences of the same object, and, hence, have to hypothesize the number of objects and their parts present in the image. Moreover, when objects are detected, it is based only on finding their associated local features, which does not delineate the objects within the image. In [10], for example, a *probabilistic map* is computed which is consistent with the detected local features. To ultimately estimate segment boundaries, it is necessary to threshold their probabilistic map, but this suffers from errors because both detected local feature locations as well as thresholds are not consistent across images. In contrast, the proposed framework allows us to redefine object detection and segmentation as precisely localizing boundaries of regions comprising all objects from the category present in the image. Since our structural model is in terms of image segments, the matched subtrees between the model and a new image automatically identify all parts of the image occupied by the recognized objects, and, hence, their precise boundaries.

Further, our approach offers two more advantages over the aforementioned prior work. First, in the prior work, the training set must be diligently selected to ensure that it does contain the category. We do not suffer from this constraint, since we need to examine only the relative frequencies of the matching subtrees which automatically disregard images without an object from the category of interest. Second, we do not need to model the background as a category by itself, and, hence, do not require a careful preparation of the background database, as opposed to prior work. Unlike discriminative approaches [1], we accomplish a satisfactory detection even when the training set contains less than ten images, similar to generative approaches [6]. In some existing work [14], local features and their higher-level groupings (e.g., topics) can be shared across categories. In our approach, analogous sharing occurs when the same nodes and subtrees occur in the canonical models of multiple categories. Therefore, our framework inherently captures shared model attributes across categories.

Related to ours is also work on learning a canonical shape model from example images discussed in [9]. Instead of trees, they use planar, region-adjacency graphs, obtained from a segmentation algorithm, to represent images. Learning is conducted as a search for the lowest common abstraction in the space of all possible region groupings of given region-adjacency-graph exemplars. Similarly, in [11], the authors represent images as planar graphs, where nodes are detected blobs, and edges capture proximity relations between node pairs. They learn a hierarchical shape model through many-to-many spectral-based matching of blob graphs across the training set. Next, in [17, 16], the authors propose to learn dynamic-structure trees as generative models of objects. Similar to our approach, they use the dynamic tree to conduct object detection and localization simultaneously through inference of both random variables and random tree structure. The same image representation as in our approach is used in [13]. The authors incrementally refine the canonical model through matching it against image trees in the training set. Their learning is realized within a neural network framework, while matching is done top-down, in a greedy manner, only between regions at the same tree level, such that a bad match between two regions penalizes attempts to match their respective descendants.

In the following sections, we present region properties associated with their corresponding nodes in segmentation trees (Sec. 3), the tree matching algorithm (Sec. 4), the algorithm to obtain a canonical category model from a set of trees (Sec. 5), the complexity of learning (Sec. 6), and finally the experimental validation of our approach (Sec. 7).

3. Specification of Node Attributes

In this paper, we assume that the segmentation trees of input images are given. Details of the segmentation algorithm can be found in [2, 15]. The attributes associated with each tree node (i.e., region) comprise intrinsic region geometric and photometric properties, as well as extraneous properties of that region's neighborhood. The intrinsic region properties we use are concerned with its gray level distribution, size and shape. Let μ_v and σ_v^2 denote the mean and variance of the gray level values of all pixels in detected region v. Also, let a_v denote the area of region v, whose center of mass (CM) is located at image coordinates (x_v, y_v). To describe the boundary shape of v, we parse the image into K pie slices, each of which begins at the CM of v, (x_v, y_v), and subtends the the same angle $2\pi/K$. Next, we compute the normalized histogram $h_v(k)$, $k = 1 \dots K$, of the number of pixels of region v that fall in pie slice k. The histogram is made rotation invariant by assigning $k = 1$ to the slice having the largest histogram value, valid under the assumption of negligible variations in viewpoint of scenes without occlusion. In case rotation invariance is not desirable, the slice with label $k = 1$ is aligned with the "x" image axis. The entropy of the normalized histogram is defined as $H_v \triangleq -\sum_{k=1}^{K} h_v(k) \log h_v(k)$. Clearly, the outlined list of useful region intrinsic properties, can be easily modified to reflect the demands of different applications.

Together, the intrinsic region properties are used to compute a measure of the nodes salience, which is in turn used to weigh the nodes influence on the overall matching process. A positive weight, w_v, defined as

$$w_v \triangleq \lambda[\frac{|\mu_v - \mu_p|}{\max(\mu_v, \mu_p)} + \frac{|\sigma_v^2 - \sigma_p^2|}{\max(\sigma_v^2, \sigma_p^2)}] + (1-\lambda)[\frac{a_v}{a_p} + H_v], \tag{1}$$

is assigned to node v, where p is the parent of v, and $\lambda \in [0, 1]$ represents the significance given to photometric relative to geometric properties.

The single extraneous region property we use captures the distribution of other salient regions in v's neighborhood. Specifically, for each node v, we compute a context vector, $\overrightarrow{\Phi}_v$, following a model similar to that used in our basic segmentation algorithm [2, 15]. Suppose each node v represents a charged particle with positive charge w_v. Then, $\overrightarrow{\Phi}_v$ is defined as the residual attraction force at the CM of region v due to all other regions u occurring in a suitably defined neighborhood $\mathcal{N}_v$, given by

$$\overrightarrow{\Phi}_v = \sum_{u \in \mathcal{N}_v} \frac{w_u}{d_{uv}^2} \overrightarrow{r}_{uv} \ , \tag{2}$$

where d_{uv} is a distance between the CMs of regions u and v, and $\overrightarrow{r}_{uv}$ is a unit vector representing the direction of attraction, and pointing from the CM of u to that of v. Since (w_u/d_{uv}^2) decays fast for nodes $u \in \mathcal{N}_v$ beyond v's siblings, we define $\mathcal{N}_v$ as consisting of v's siblings, only. The context vector, $\overrightarrow{\Phi}_v {=} [|\overrightarrow{\Phi}_v|, \phi_v]$, is made rotation invariant if angle ϕ_v is computed relative to the geometry of v's parent, e.g., counter-clockwise from the axis of the pie slice of v's parent having the largest histogram value. Otherwise, ϕ_v is computed with respect to the "x" image axis.

Finally, the attribute vector associated with each node v is $\Omega_v {=} [\mu_v, \sigma_v^2, a_v, x_v, y_v, h_v(1), \dots, h_v(K), \overrightarrow{\Phi}_v]$.

4. The Tree Matching Algorithm

We use a modified version of the tree matching algorithm discussed in [18, 19]. Given two trees, the objective is to find their maximum common subtree that maximizes a match measure. In [18, 19] this measure is maximized recursively, for subtrees rooted at different candidate matching nodes, as the two trees are traversed bottom-up. We expand the scope of the match measure to include the candidate nodes' context vectors. To formalize our approach, below, we first present necessary definitions.

Given two trees $t {=} (V_t, E_t, \Omega_t)$ and $t' {=} (V_{t'}, E_{t'}, \Omega_{t'})$, where V is the set of nodes, $E {\subseteq} V {\times} V$ is the set of edges, and Ω is the set of node attributes, the goal of tree matching is to find the *maximum subtree isomorphism*. Any bijection $f{:}U_t {\to} U_{t'}$, where $U_t {\subseteq} V_t$ and $U_{t'} {\subseteq} V_{t'}$, is called subtree isomorphism if it preserves the connectivity and ancestor-descendent relations between the nodes in t and t'. The objective is to find a subtree isomorphism that maximizes a quality measure, called *utility*, defined as

$$\mathcal{U}(U_t, f(U_t)) = \sum_{v \in U_t} [w_v + w_{f(v)} - m_{vf(v)}]. \tag{3}$$

Thus, the goal of the subisomorphism algorithm is to maximize $\mathcal{U}$ by matching salient nodes with large w_v and $w_{f(v)}$ values, while discarding those node pairs whose cost of matching, $m_{vf(v)}$, is high.

Given node attributes, Ω_v and $\Omega_{v'}$, we specify the cost of matching regions v and v' as

$$m_{vv'} {=} \lambda \frac{(\mu_v {-} \mu_{v'})^2}{\sigma_v^2 {+} \sigma_{v'}^2} + (1{-}\lambda)[|\frac{a_v}{a_p} - \frac{a_{v'}}{a_{p'}}| + \rho_{vv'}] + ||\overrightarrow{\Phi}_v - \overrightarrow{\Phi}_{v'}||, \tag{4}$$

where p and p' are the parents of v and v', respectively, and $\lambda \in [0, 1]$ is a desired weight of the photometric properties relative to geometric properties. The cost of matching the shapes of two regions v and v', represented by their normalized histograms h_v and $h_{v'}$, is defined as the χ^2 test statistic: $\rho_{vv'} {=} \frac{1}{2} \sum_{k=1}^{K} \frac{(h_v(k) {-} h_{v'}(k))^2}{h_v(k) {+} h_{v'}(k)}$.

The tree-matching problem is solved recursively through a set of sub-matching problems. Suppose that at any stage during matching, for all descendants u of v in t, and for all descendants u' of v' in t' we have previously computed the maximum utility, given by (3), of subtrees rooted at u and

u', denoted as $\mathcal{U}(u,u')$. Then, our goal is to find the optimal set of (u,u') pairs, referred to as *consistent* descendant pairs, that preserve connectedness and ancestor-descendant relationships in t and t', while maximizing $\mathcal{U}(v,v')$. From (3), we have

$$\mathcal{U}(v,v'){=}w_v{+}w_{v'}{-}m_{vv'}{+}\textstyle\sum_{(u,u')\in\mathcal{C}_{vv'}}\mathcal{U}(u,u')\ , \tag{5}$$

where $\mathcal{C}_{vv'}$ denotes the set of selected consistent pairs of descendants of v and v'. Once computed, $\mathcal{U}(v,v')$ is used to recursively compute the utility of the parent node pair, (p,p'), if $(v,v')\in\mathcal{C}_{pp'}$.

As shown in [19], finding $\mathcal{C}_{vv'}$ is equivalent to solving the maximum weighted clique problem over an auxiliary weighted graph, whose nodes (u,u') are defined as the Cartesian product of all the descendants of v and v', and whose edges connect only consistent child pairs. The weight associated to each node (u,u') in this auxiliary graph is equal to utility $\mathcal{U}(u,u')$. To compute the maximum weighted clique of this graph, we use the replicator dynamics algorithm thoroughly discussed in [12].

Now, from the maximum weighted cliques obtained for each pair of nodes (v,v'), $v{\in}t$ and $v'{\in}t'$, we are in a position to compute $\mathcal{U}(v,v')$ using (5). Then, the pair $(v,v')^*$ with the largest utility determines the maximum subtree isomorphism between t and t'. In this maximum common subtree, the set of matched node pairs is determined by the clique $\mathcal{C}_{vv'^*}$. Since our goal is to extract all common, salient subimages in t and t', we select all common subtrees whose utility is above a specified threshold, as explained in Sec. 6.

5. Canonical Model

From the set of matched subtrees, we derive the canonical model as their tree-union, similar to the approach discussed in [18]. The tree-union is constructed by first finding the maximum common subtree among the trees in the set, and then by adding the extraneous nodes and edges that did not get matched. Unlike in [18], however, nodes in our tree-union are associated with the frequency of their occurrence in the set. Another difference is that we compute node attributes in the tree-union as the median of the corresponding sample attributes from the set, to reduce the influence of noise present. Below, we briefly present our algorithm for computing tree-unions from a different point of view than that given in [18], by relating it with a more general algorithm for finding the weighted minimum common supergraph, discussed in [3].

The *difference* between tree t and its subtree s, $t{-}s$, is obtained by removing all nodes and edges in s from t, and all the edges that connect s with the rest of t. The latter set of edges is called the *embedding* of s in t, and denoted as $\varepsilon(s,t)$. The union of two distinct trees t and t', $t\cup_E t'$, is a graph composed of t and t', joined by a given set of edges E. The maximum common subtree between t and t' is a tree τ, such that there exist subtree isomorphisms from τ to t and from τ to t', and there exists no other common subtree τ' of t and t' whose utility, given by (3), is larger than the utility of τ. The tree-union of t and t', is a directed graph, $\mathcal{T}{=}(V_\mathcal{T},E_\mathcal{T},\Omega_\mathcal{T})$, such that there exist the subtree isomorphisms from t to $\mathcal{T}$, characterized by utility $\mathcal{U}(t,\mathcal{T})$, and from t' to $\mathcal{T}$, with utility $\mathcal{U}(t',\mathcal{T})$, such that the sum of these utilities is maximum over all possible isomorphisms. Here, utility $\mathcal{U}$ is defined as in (3). The following theorem explains how to compute the tree union.

Theorem 1. *Given t and t', their tree-union is $\mathcal{T} = \tau{\cup_{\varepsilon_1}}(t{-}\tau){\cup_{\varepsilon_2}}(t'{-}\tau)$, where $\varepsilon_1{=}\varepsilon(\tau,t)$ and $\varepsilon_2{=}\varepsilon(\tau,t')$.*

The proof of Theorem 1 is similar to the proof given in [3] for the maximum common supergraph.

Finding $\mathcal{T}$ becomes infeasible over a large set of trees. Therefore, we resort to a suboptimal approach, where in each iteration the tree-union is extended by adding a new tree from the set. To diminish the effect of tree ordering, we first generate R random permutations, $\pi_1,\ldots,\pi_R$, of the tree set $\{t_1,\ldots,t_N\}$, and then compute tree-union $\mathcal{T}(\pi_i)$ for each permutation π_i, as summarized in Algorithm 1. Ultimately, the optimal $\mathcal{T}(\pi_i)$ is selected.

Algorithm 1: Finding $\mathcal{T}(\pi)$ for $\pi(\{t_1,\ldots,t_N\})$

```
1 Find $\tau$ of $t_1$ and $t_2$ as explained in Section 4;
2 $\varepsilon_1{=}\varepsilon(\tau,t_1)$; $\varepsilon_2{=}\varepsilon(\tau,t_2)$; $\mathcal{T}{=}\tau\cup_{\varepsilon_1}(t_1{-}\tau)\cup_{\varepsilon_2}(t_2{-}\tau)$;
3 for $i = 3, N$ do
4     Find $\tau$ of $\mathcal{T}$ and $t_i$ ;
5     $\varepsilon_1{=}\varepsilon(\tau,\mathcal{T})$; $\varepsilon_2{=}\varepsilon(\tau,t_i)$; $\mathcal{T}{=}\tau\cup_{\varepsilon_1}(\mathcal{T}{-}\tau)\cup_{\varepsilon_2}(t_i{-}\tau)$;
6 end
7 $\mathcal{T}(\pi) = \mathcal{T}$;
```

While Step 4 of Algorithm 1 involves solving an NP-complete problem when finding the maximum common subgraph, as discussed in [3], this is not the case in this paper, since the maximum common subtree is computed over directed graph $\mathcal{T}$ and tree t_i from the set. Indeed, even though $\mathcal{T}$ contains multiple paths between the same pairs of nodes, it is possible to treat all these paths as mutually exclusive [18], and to find the maximum subtree isomorphism between $\mathcal{T}$ and t_i, as explained in Section 4.

In [18], in Step 4 of Algorithm 1, all the node weights of previously matched trees from the set are kept, and associated with the corresponding node in the tree-union. Thus, suppose n trees from the set have already been added to the tree-union in Steps 3-6. Then, a vector $W_v{=}[w_v^{(1)},\ldots,w_v^{(n)}]$ is assigned to each node $v{\in}\mathcal{T}$, where $w_v^{(j)}{=}0$ if no node in tree t_j got matched with v, and $w_v^{(j)}{=}w_u$ if node $u{\in}t_j$ got matched with v. In Step 4 of the next $n{+}1$ iteration, to find the maximum subtree isomorphism between t_i and $\mathcal{T}$, $f{:}U_{t_i}{\to}U_\mathcal{T}$, where U_{t_i} and $U_\mathcal{T}$ are subsets of nodes in t_i and $\mathcal{T}$, respectively, the following utility function is maximized: $\mathcal{U}(U_{t_i}, U_{\mathcal{T}}) = \sum_{j=1}^{n} \sum_{(u,v)\in U_{t_i}\times U_{\mathcal{T}}} [w_u + w_v^{(j)} - m_{uv}^{(j)}]$.

The above formulation, however, is not resilient to noise present in the structure and node attributes of the trees in the set. To alleviate this problem, in this paper, we specify the attributes of v in $\mathcal{T}$ as the median of attributes of all those nodes in the set that got matched with v. For example, suppose in the first n iterations of Steps 3-6, ℓ nodes from the set, $\{u_1, \ldots, u_\ell\}$, have been matched with $v\in\mathcal{T}$, then Ω_v=median$(\Omega_{u_1}, \ldots, \Omega_{u_\ell})$. Once Ω_v is computed, we specify the weight of $v\in\mathcal{T}$, w_v, as in (1). Also, the cost of matching any node u in the set with $v\in\mathcal{T}$, m_{uv}, is defined as in (4). Given these values, Step 4 of the $(n{+}1)$th iteration maximizes the utility function

$$\mathcal{U}(U_{t_i}, U_{\mathcal{T}}) = \sum_{(u,v)\in U_{t_i}\times U_{\mathcal{T}}} [w_u + \eta_v w_v - m_{uv}], \quad (6)$$

where $\eta_v{=}\ell/n$ is the frequency of occurrence of $v\in\mathcal{T}$ in n previously added trees from the set to $\mathcal{T}$. The saliency of $v\in\mathcal{T}$, w_v, is weighted by η_v to force the matching algorithm in Step 4 to eliminate all those candidate nodes in $\mathcal{T}$ that represent structural noise in the set.

The best approximation of the tree-union is selected based on the following entropy function:

$$H_{\mathcal{T}}(\pi_i) = -\sum_{v\in\mathcal{T}(\pi_i)} \eta_v \log \eta_v \ , \quad (7)$$

which achieves a minimum for the sets containing all isomorphic trees. Thus, the permutation π_i for which $H_{\mathcal{T}}(\pi_i)$ is minimum over all $\pi_1, \ldots, \pi_R$ is selected to compute $\mathcal{T}(\pi_i)$ as the best approximation of the tree-union. In the case of multiple solutions, the $\mathcal{T}(\pi_i)$ with the smallest number of nodes is selected

6. Computational Complexity

Fig. 2 illustrates the two main steps of our approach to learning the canonical model of an unknown category, which, as stated earlier, is efficient even when the number of example images is small. Thanks to a small training set, it is feasible to conduct, in the first step, pairwise matching of all image trees by using the algorithm presented in Sec. 4. Suppose there are M training trees, which all differ in structure and number of nodes $|V|$. Next, recall that for matching two trees $t{=}(V_t, E_t, \Omega_t)$ and $t'{=}(V_{t'}, E_{t'}, \Omega_{t'})$ it is necessary to solve for $|V_t| \cdot |V_{t'}|$ maximum clique problems, each involving relaxation labeling [12]. Relaxation labeling in such problems typically converges after only a few iterations, where each iteration includes $O(|G|^2)$ multiplications, and where $|G|$ is the number of nodes in a graph whose maximum clique we are supposed to compute. Therefore, the complexity of the first step is $O(M^2|V|^4)$.

In this paper, we select the maximum matching subtrees by thresholding the measure of their match, i.e., their utility values. The threshold is set to one standard deviation below the maximum utility value obtained over all $M(M{-}1)$ image pairs. This produces a set of N maximum common subtrees in $2M(M{-}1)$ trees. In the second step, this set of extracted subtrees is used to construct the tree-union as explained in Sec. 5. Suppose no subtree has more than $|V_s|$ nodes ($|V_s|{\ll}|V|$). Then, for R permutations of the set of subtrees, the complexity of the second step is $O(RN|V_s|^4)$.

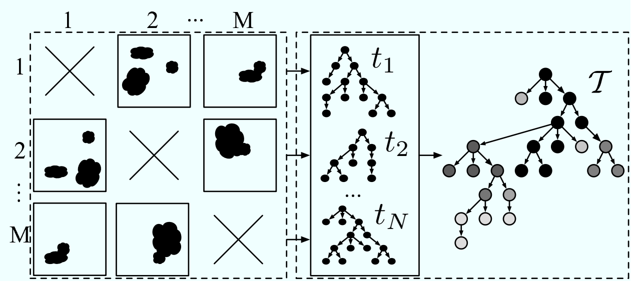

Figure 2. Learning: (1) Image pairwise matching, (2) Computing the tree-union; frequency of tree-union nodes is indicated by different shades of gray.

The segmentation algorithm of [2, 15] typically produces trees with no more than approximately 100 nodes, i.e., $|V|{\approx}100$. In our experiments, we set $R{=}10$. In case there are $M{=}20$ training images, as for example when learning the model of side-view cars in the UIUC database [4] (see Sec. 7), the computation time of our learning algorithm is less then two hours on a 2.4GHz, 2GB RAM PC. This does not include segmentation algorithm. Therefore, our learning algorithm is computationally feasible, and very efficient as demonstrated in the following section.

7. Experiments and Discussion

For empirical validation we use the following benchmark datasets: Caltech *faces* (435 images) and *cars rear view* (126 images) [5], and UIUC *cars side view* (single scale, 170 images) [4]. While the Caltech images contain only a single object from the category, the UIUC images contain multiple occurrences of cars. Both databases also contain background images, in which objects from the target category do not appear, referred to as negative examples. To test rotational invariance of our approach, we form a fourth dataset by randomly rotating Caltech *faces*, such that the size of the original scene is kept intact. The missing background of the new rotated image is “filled out” by pasting a randomly selected background image from the Caltech database, as illustrated in Fig. 3.

The original three datasets, containing objects from the category, are referred to as positive examples. Training is

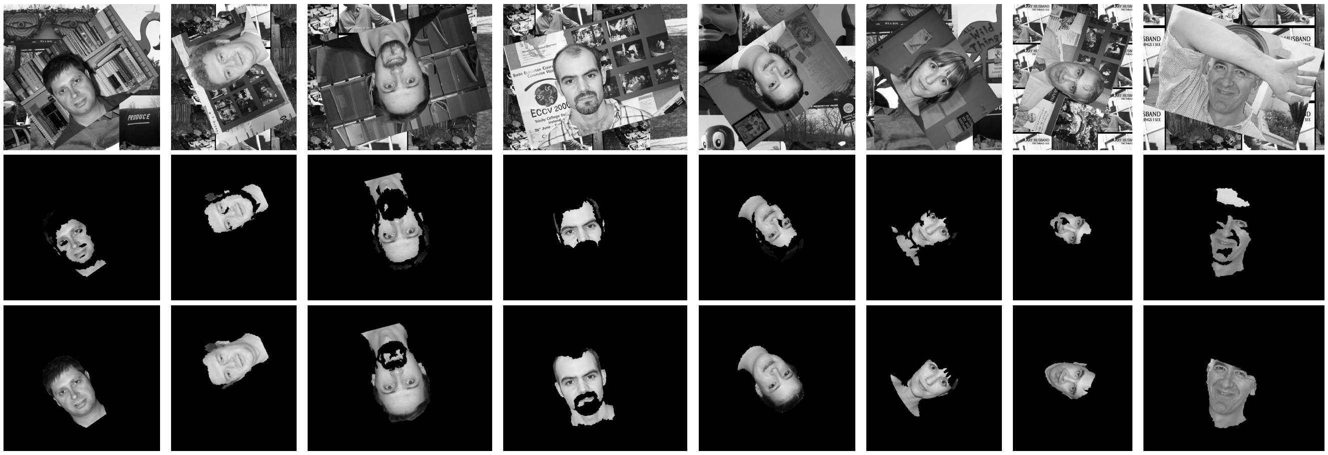

Figure 3. Detection and localization of Caltech *faces*: (top row) randomly rotated images; the results are obtained when the model is learned on $M/2$=3 (middle row), and $M/2$=6 (bottom row) positive training images, and for the highest F-measure. The training set did not contain a bearded face.

conducted on a number of randomly drawn positive and negative examples. This is done to ensure unsupervised training, where it is not a priori known whether a training image contains objects from a category. In the training set of $M = \{4, 6, 8, \ldots, 20\}$ images only half of the images are positive. After selecting the training data, the remaining positive examples are used as the test set. Each experiment is repeated 5 times for each value of M to estimate the average performance.

The performance is evaluated in terms of the accuracy of object detection and localization. Detection and localization in test images are performed jointly by matching the tree-union model with the test-image trees. Those common subtrees whose cost function is larger than a specified threshold are adjudged as detected objects. The threshold is varied to plot a recall-precision curve, as a preferred measure of performance with respect to object detection and localization, compared to those used by classification-based techniques (e.g., ROC curve, and equal error rate) [1]. We define: Recall $=$ TP$/n_P$, and Precision $=$ TP$/$(TP+FP). To obtain the ground truth, we manually delineated the outer contours of target objects in all test images, the total number of which is n_P. The matched subtree in a test image is said to be false positive (FP) in two cases: (1) if the intersection of the area it covers in the test image and the labeled object area is less than 75% of the true object area, or (2) if more than 25% of the matched subtree area lies outside of the true object contours. The remaining cases are declared true positives (TP). The reported performance results are computed for the highest F-measure, where $F = 2 \cdot$ Precision $\cdot$ Recall$/$(Precision+Recall). Localization error is defined as the total unmatched object area expressed as a percentage of the total area of all target objects in the test image, computed for the highest F-measure. We find that these performance criteria correspond closely with our own subjective judgement. Note that our evaluation is more strict than in [7, 6], where object detection is interpreted as image classification, and than the evaluation discussed in [1], where correct detection is required to lie within an ellipse of a certain size centered at the centroid of the true object area.

For the two *car* datasets, the mean gray level of segmented regions is excluded when computing the matching cost function, but not the regions' gray-level variance. Also, for rotated *faces*, the rotation-invariant formulation of shape histograms and context vectors is used, which is valid since the original *faces* do not undergo major changes in viewpoint and occlusion across the set. It may be desirable to optimize parameter λ, which represents the relative influences of photometric and geometric region properties on tree matching, to obtain the best detection performance. For the four datasets, our experiments, however, suggest that the optimal choice of λ is 0.5, as shown in Fig. 5. A plausible explanation of this could be that the choice of λ is indeed a function of region properties themselves (e.g. scales), and that the dominance of either photometric or geometric properties is averaged out over the large range of region property combinations present in our data. All results in the sequel are obtained for $\lambda = 0.5$.

Figs. 3, 4, 6 illustrate object detection results for the highest F-measure. In each, the top row consists of input test images. The middle and bottom rows present detection results of two experiments differing in the number of positive examples drawn to form the training set. The examples show that detection and localization are achieved simultaneously. The highlighted areas are the extracted subimages

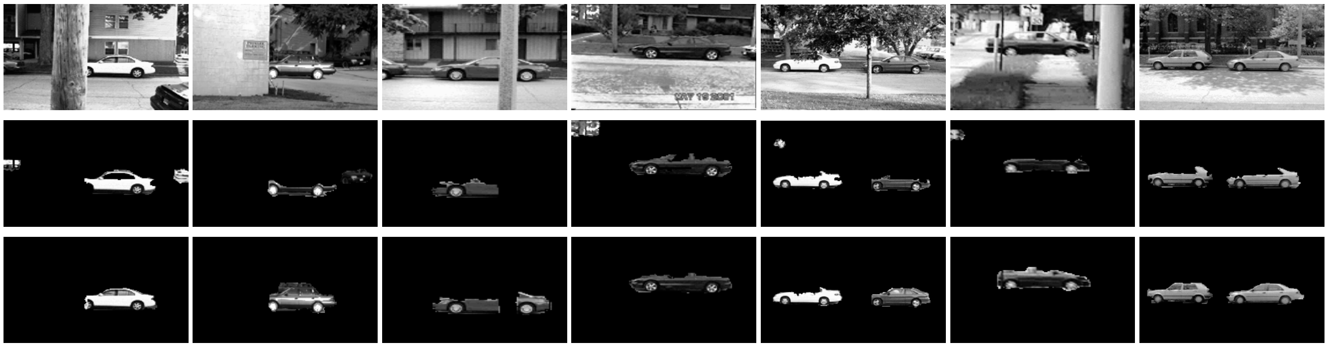

Figure 4. UIUC *cars side view*: the results are obtained for for the highest F-measure, using $M/2 = 5$ (middle row), and $M/2 = 10$ (bottom row) positive training images.

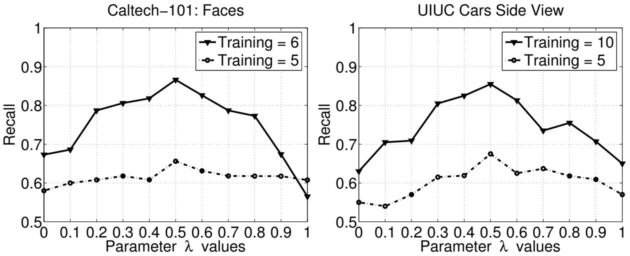

Figure 5. Recall over a range of λ values; $\lambda = 0.5$ is optimal. The legend shows the number of positive examples in the training set.

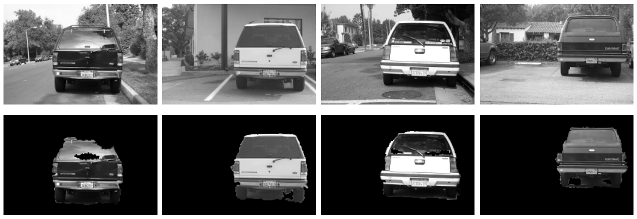

Figure 6. Results on Caltech *cars rear view* using $M/2 = 10$ positive training images.

	Faces		Cars side		Cars rear
# Posit.	6	5	10	5	10
Recall	84.6±1.3	71.2±2.5	85.5±2.7	67.5	80.2
Precision	78.2±1.8	73.8±2.8	87.5±2.4	62.4	72.2
Loc. error	6.8±1.5	11.4±3.1	9.3±3.4	13.4	12.8

Table 1. Average recall, precision, and localization error (in %). The second row shows the number of positive training examples.

identified as occupied by the detected object, and the dark areas are those where the cost of matching with the canonical model exceeds the threshold determined by the highest F-measure. As expected, the detection and localization performance improves as the number of positive training examples increases. To illustrate our measures of true and false positives defined above, observe the leftmost car image in Fig. 4. For $M/2 = 5$ positive training images, there are two false positives, and one true positive has large localization error (mismatch with the actual car region). For $M/2 = 10$, the two false positives in the previous case disappear and localization error of the true positive decreases.

Averages of object detection and localization results for the highest F-measure are summarized in Table 1. The average recall, precision and localization error obtained for rotated *faces* are less than one standard deviation different from the values of their corresponding quantities reported in Table 1 for *faces*. This small difference (in part due to a relatively small number of trials over which the results are averaged, and the quantization error accompanying rotation with arbitrary digital rotation angles) verifies that our approach is rotation invariant, when the orientation independent tree representation of the image structure is used. For *faces*, we obtain the best recall and precision, whereas the Caltech *cars rear view* dataset proves the most challenging. Huge variations in the appearance of the rear window, due to the reflections of surround, lead to the appearance of spurious regions in varying locations, not consistently present in training images, which do not become part of the learned model, and, therefore, are not matched with the model. Typically, these effects are large enough to penalize the corresponding matched subimage from being interpreted as a true positive, but localized enough for the subimage to be evaluated as a false positive.

Recall-precision curves are given in Figs. 7 and 8. As can be seen, only a small increase in the number of positive

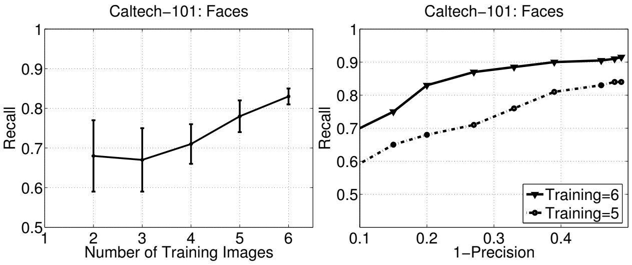

Figure 7. Caltech *faces*: (left) Recall using $M/2 = 2, 3, \ldots, 6$ positive training examples; (right) Recall-Precision curves for 5 and 6 positive training examples.

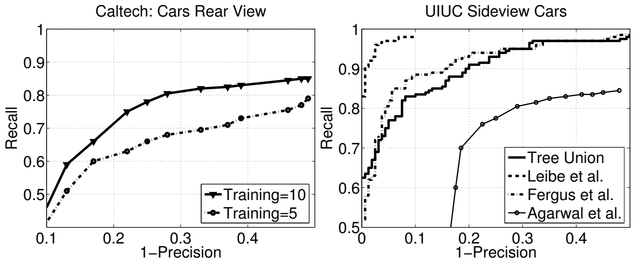

Figure 8. Recall-Precision curves: (left) Caltech *cars rear view*; (right) Comparison with prior work [10, 7, 1] on UIUC *cars side view*. "Tree-union" indicates our approach using 10 positive training images.

examples yields significant improvements in the detection performance. This is also illustrated in Fig. 7(left) where recall values are computed for increasing numbers of positive training examples. Apparently, the canonical model is capable of capturing the structural and geometric and photometric properties within the object category of *faces* from a very small training set ($.6 - .75$ recall for two positive training images). Moreover, the model improves considerably with each newly added positive example.

The following recall-precision results on the UIUC *cars side view* dataset have been reported in the literature: 79% in [1], 88.5% in [7], 97.5% in [10]. Since these results have been obtained by using less demanding evaluation criteria than ours, the question as to which method is better on the given data still remains to be examined carefully.

Acknowledgment

Himanshu Arora provided the segmentation code. The support of the Office of Naval Research under grant N00014-06-1-0101 is gratefully acknowledged.

References

[1] S. Agarwal, A. Awan, and D. Roth. Learning to detect objects in images via a sparse, part-based representation. *IEEE TPAMI*, 26(11):1475–1490, 2004.

[2] N. Ahuja. A transform for multiscale image segmentation by integrated edge and region detection. *IEEE TPAMI*, 18(12):1211–1235, 1996.

[3] H. Bunke, P. Foggia, C. Guidobaldi, and M. Vento. Graph clustering using the weighted minimum common supergraph. In *IAPR Workshop GbRPR, Springer LNCS*, volume 2726, pages 235–246, 2003.

[4] `l2r.cs.uiuc.edu/∼cogcomp/Data/Car/`.

[5] `vision.caltech.edu`.

[6] 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.

[7] R. Fergus, P. Perona, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. In *CVPR*, volume 2, pages 264–271, 2003.

[8] R. Fergus, P. Perona, and A. Zisserman. A sparse object category model for efficient learning and exhaustive recognition. In *CVPR*, volume 1, pages 380–387, 2005.

[9] Y. Keselman and S. Dickinson. Generic model abstraction from examples. *IEEE TPAMI*, 27(7):1141–1156, 2005.

[10] B. Leibe, A. Leonardis, and B. Schiele. Combined object categorization and segmentation with an implicit shape model. In *Workshop on Statistical Learning in Computer Vision, ECCV*, pages 17–32, 2004.

[11] A. Levinshtein, C. Sminchisescu, and S. Dickinson. Learning hierarchical shape models from examples. In *EMM-CVPR, Springer LNCS*, volume 3757, pages 251–267, 2005.

[12] M. Pelillo, K. Siddiqi, and S. W. Zucker. Matching hierarchical structures using association graphs. *IEEE TPAMI*, 21(11):1105–1120, 1999.

[13] B. Perrin, N. Ahuja, and N. Srinivasa. Learning multiscale image models of 2D object classes. In *ACCV, Springer LNCS*, volume 1352, pages 323–331, 1998.

[14] E. Sudderth, A. Torralba, and W. Freeman. Learning hierarchical models of scenes, objects, and parts. In *ICCV*, 2005.

[15] M. Tabb and N. Ahuja. Multiscale image segmentation by integrated edge and region detection. *IEEE Trans. Image Processing*, 6(5):642–655, 1997.

[16] S. Todorovic and M. C. Nechyba. Interpretation of complex scenes using generative dynamic-structure models. In *CVPR Workshop Generative-Model Based Vision*, volume 12, page 184, 2004.

[17] S. Todorovic and M. C. Nechyba. Dynamic trees for unsupervised segmentation and matching of image regions. *IEEE TPAMI*, 27(11):1762–1777, 2005.

[18] A. Torsello and E. R. Hancock. Matching and embedding through edit-union of trees. In *ECCV*, volume 3, pages 822–836, 2002.

[19] A. Torsello and E. R. Hancock. Computing approximate tree edit distance using relaxation labeling. *Pattern Recogn. Lett.*, 24(8):1089–1097, 2003.