The more you look, the more you see: towards general object understanding through recursive refinement

Jingyan Wang[1] Olga Russakovsky[1, 2] Deva Ramanan[1]
[1]Carnegie Mellon University [2]Princeton University
jingyanw@cs.cmu.edu, olgarus@cs.princeton.edu, deva@cs.cmu.edu

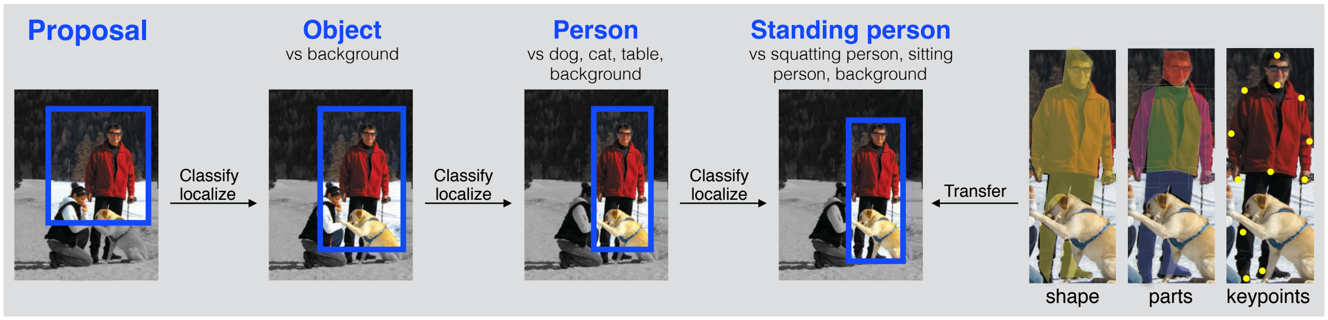


Figure 1: **Summary:** In this work, we present a general framework that tackles general object understanding. In our model, a refinement module recursively develops understanding across space and semantics: our model gradually classifies the region into finer categories ("standing person") and adaptively localizes the bounding box. As a result, our model is flexible for transferring annotations, such as shapes, parts and keypoints, from similar examples in the *train* set.

Abstract

Comprehensive object understanding is a central challenge in visual recognition, yet most advances with deep neural networks reason about each aspect in isolation. In this work, we present a unified framework to tackle this broader object understanding problem. We formalize a refinement module that recursively develops understanding across space and semantics — "the more it looks, the more it sees." More concretely, we cluster the objects within each semantic category into fine-grained subcategories; our recursive model extracts features for each region of interest, recursively predicts the location and the content of the region, and selectively chooses a small subset of the regions to process in the next step. Our model can quickly determine if an object is present, followed by its class ("Is this a person?"), and finally report fine-grained predictions ("Is this person standing?"). Our experiments demonstrate the advantages of joint reasoning about spatial layout and fine-grained semantics. On the PASCAL VOC dataset, our proposed model simultaneously achieves strong performance on instance segmentation, part segmentation and keypoint detection in a single efficient pipeline that does not require explicit training for each task. One of the reasons for our strong performance is the ability to naturally leverage highly-engineered architectures, such as Faster-RCNN, within our pipeline. Source code is available at https://github.com/jingyanw/recursive-refinement.

1. Introduction

Understanding objects is one of the central challenges in visual understanding, encompassing a diverse range of tasks such as object detection, fine-grained classification, 3D shape estimation, keypoint correspondence estimation, *etc.* Though recent history has seen tremendous strides in these tasks, they are typically addressed in specialized modules tuned for each. As we scale up recognition systems to the "open-world" of all possible visual (sub)categories and all possible tasks, and as such systems are applied on robotic platforms with limited computational budgets, learning and processing massive numbers of distinct modules will no longer be practical.

Challenges: One potential solution is combining all possible outputs of interest into a centralized multi-task recognition framework [6, 24]. Indeed, one of the attractive properties of deep networks is the ability to share features across a variety of target problems [12, 36]. We see three difficulties in naively doing so. The first is **computation**. Given tens of thousands of object subcategories and target output combinations, standard deep architectures will no longer scale. The second is **supervision**. It is difficult in practice to collect datasets with detailed annotations across a large number of objects to train these multi-task models. The final challenge is **learnability**. As we consider a large number of tasks, it becomes important to design architectures that take into account relationships between outputs of interest. Before we describe our proposed solution, we first review related work.

Semantic hierarchies: Much past work has explored the use of object hierarchies to scale visual recognition. For example, visual categories in ImageNet are naturally arranged in a semantic hierarchy (defined by the external knowledge base WordNet [38, 33]). The hierarchy can be used to reduce computation (*e.g.*, a system may first report if an image contains a vehicle, and if so and only then, evaluate a car and bus model [9, 1, 3, 10]) and to regularize learning (car and bus models may share features [42, 39]). One powerful hierarchical representation is a decision tree [4, 25], which can be scaled to large multi-class problems [46].

Spatial search: An important aspect of object understanding is spatial localization, which is most well-studied in the context of object detection. Historically, architectures based on exhaustive scanning window search have produced strong results [15]. Past work in object detection has explored sequential processing for reducing the number of spatial windows that need to be processed. This is often formalized as a search problem, addressed using branch and bound [27, 50], coarse-to-fine cascades [14, 35, 47], or active search [19, 5, 30]. Typically, these focus on efficient detection of a small fixed number of semantic categories.

Subcategories: Our approach is based on visual subcategories [11, 28, 34, 48], where a category of interest, such as a `person`, can be refined into subcategories based on visual appearance, such as `sitting` versus `standing`. In the extreme case, each training example can be its own subcategory (exemplars) [31, 26], which is shown to be effective for label transfer [29, 26, 43]. Typically, such approaches focus on a flat clustering of subcategories rather than a hierarchical structure, but we later show that a flat structure does not work well when the number of classes scales up (>1000).

Our approach: We combine all these three aspects into a single recognition framework that processes an image by recursing over semantics and space. We make two crucial observations. Firstly, we use a semantic hierarchy that includes both coarse-grained super-categories and fine-grained subcategories [32]. For example, the top-level of our hierarchy may categorize image regions into objects or background, which are then divided into cars and buses, which are then divided into subordinate categories based on viewpoint. Secondly, because coarse-grained and fine-grained classification require different amounts of spatial acuity, we interleave the search over spatial locations and semantics. As our model makes finer-grained semantic predictions, it re-extracts features from different spatial locations so as to support the granularity of the current task at hand. We visualize our overall framework in Fig. 1.

Our contribution: Our main contribution is on the *integration of semantic search (by a hierarchy) and spatial search (by recursive localization)* in a practical detection pipeline for object understanding. We show that our model can be used to support a wide variety of vision tasks including object detection, segmentation, part estimation, and keypoint localization. Though we do not outperform state-of-the-art specialized modules tuned for each task, we demonstrate strong performance using a single, scalable architecture. We demonstrate the importance of *jointly* recursing over hierarchical semantics and spatial layout. We formalize the design principles of the state-of-the-art object detection architecture Faster RCNN [37], which is a two-level example of our framework. Our analysis shows that extending such pipelines recursively (*e.g.*, learning spatially-refined classifiers for `side` versus `frontal` cars given `car` "proposals") yields a scalable, unified pipeline for general object understanding.

2. Recursive refinement

Problem setting: Given an image, we write a near exhaustive set of candidate regions as $s \in S$, where $s = [x_1, y_1, x_2, y_2]$. To be concrete, these may be the set of regions considered by a region proposal network (RPN) [37], and so they will span a large but discrete set of positions, scales, and aspect ratios. We want to efficiently process these regions to produce a distribution over (sub)class labels, as well as refined spatial estimates of each region. Importantly, the subclass distributions include a background class.

Model: The image will be processed with a hierarchical model represented by a tree graph $G = (V, E)$. Each node in the graph $v_i \in V$ represents a semantic class. As a running example, let v_i represent the person class. We parameterize this class by a classifier ω_i, a spatial regressor θ_i, and a threshold t_i for deciding when to stop refinement. We use these weights to define a softmax distribution over the children of node v_i:

$$p(v_j = 1 \mid v_i = 1) \propto e^{\omega_j \cdot F(s)}, \quad \forall v_j \in \text{children}(v_i)$$

where $F(s)$ corresponds to a (deep) image feature extracted from region s, and the dot denotes dot-product between the feature and the classifier (it should be a non-linear operation of *fc6* and *fc7* layers followed by the classifier, but we

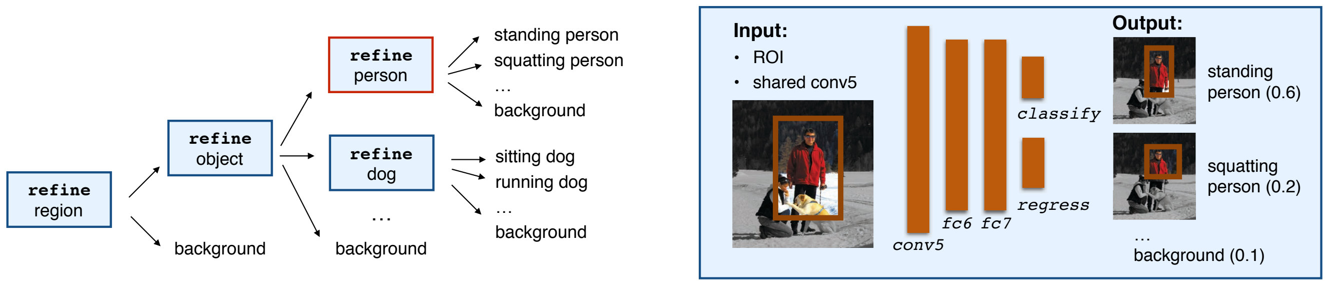


Figure 2: **Network architecture: Left:** We build an object hierarchy, and recursively call our refinement module on each node to efficiently reason about a large set of candidate regions. **Right:** A zoomed-in version of the refinement module on the `person` node. For this module, we take all the "person proposals" from the `object` node, localize each "person proposal" and classify it into finer subclasses. If a node is a leaf node, we output the final probability distribution; if a node is not a leaf node, we selectively pass the highly-confident proposals to its corresponding child nodes.

present a linear operation here for notation clarity). Node v_i additionally has a spatial regressor

$$s := \theta_i \cdot F(s)$$

which we write as parameterized by a vector to simplify notation, though formally it should be a matrix to ensure that the output remains a 4-tuple.

Hierarchical traversal: Given a region s that reaches node v_i, we recursively apply the module in Alg. 1.

Algorithm 1: **Recursive refinement.**

```
1 Function refine(s, v_i):
2     $s := \theta_i \cdot F(s)$;
3     foreach $v_j \in$ children($v_i$) do
4         if $p(v_j = 1 \mid v_i = 1) > t_i$ then
5             refine($s, v_j$);
6         end
7     end
```

Finally, if a region reaches a leaf node that is not background, we compute the final distribution by multiplying the conditionals along the path from the root to the leaf:

$$p(v_l = 1) = \prod_{i \in \text{path}(r \to l)} p(v_i = 1 \mid v_{\text{parent}(i)} = 1), \qquad (1)$$

where we define the root v_r to have probability $p(v_r) = 1$ for all regions s. We can also use Equation 1 to compute the probability of a region belonging to any intermediate node.

Background subclasses: We found it crucial to include background subclasses at each node of the hierarchy. This is because parent nodes suffer from false positives in practice. Then, such false positives can be pruned if classified as backgrounds downstream in hierarchy. Interestingly, we find that these background classes also have the potential to capture "hierarchical open-world" detections, where the model correctly detects a person, but realizes that their pose does not correspond to a previously-trained pose subclass, as will be shown in the experiments.

Hierarchical learning: We learn our model in a hierarchical, top-down fashion. If we assume that the parent node of v_i is already trained, we learn parameters for v_i by running the current hierarchical model to generate region proposals in the training set.

Given a fixed hierarchy and images with object bounding boxes labeled with their leaf class, it is straightforward to show that model parameters and (deep) image features

$$(\{w_i, \theta_i : v_i \in V\}, F)$$

can be optimized with gradient descent. Note that learning does not directly require detailed shape masks or keypoints, but these might be indirectly used to learn the hierarchy through hierarchical clustering. In this work, we generate subclass labels by clustering the object annotations, and explore the performance as a function of the type of the annotation used. In theory, joint learning of the hierarchy and the parameters is possible with gradient-based methods [25], but we perform a simpler discrete search.

Though hierarchical learning is often argued from a computational point-of-view, our experimental results demonstrate that it actually *improves* accuracy because it behaves as a natural form of hard-example mining during gradient-based learning [40]. Each softmax classifier is trained only on those regions that passed through its parent, and so easily-discardable regions are not used for training. We use back-propagation to learn both the classifiers, regressors, and image features.

Connection to Faster-RCNN: Faster-RCNN is a two-stage detector. In the first stage, the region proposal network (RPN) searches over all possible image locations and scales, generating a small number ($\sim$2000) of object proposals. Then, a second stage classifies these proposals into semantic categories or the background. Faster-RCNN naturally falls under our generalization: it exploits a two-level hierarchy, where the first level is $\{v_{obj}, v_{bkg}\}$, and the node v_{obj}

has children of semantic categories $\{v_{dog}, v_{person}, \cdots\}$. F is a network that extracts *conv5* feature maps, followed by ROI pooling for each region.

Realization: In this work, we extend this recursion by building another level of subclass categorization on top of the two-level detection hierarchy. Our proposed approach is generalizable to arbitrary definition of subclasses (pre-defined knowledge base such as WordNet, or clustering within each category), and arbitrary construction of hierarchies (any tree graph). Fig. 2(left) describes one realization of this architecture with a running example: the model firstly outputs object regions, secondly classifies them into semantic categories such as "person", and finally selects the fine-grained subcategory such as the "standing person". This "standing person" subcategory not only provides a subclass label, but, more importantly, also extra detailed information about the person, such as its shape, parts, keypoints, *etc*. For the rest of the paper, we stick with this three-level model.

3. Implementation

Clustering: We cluster each semantic category into subcategories by the k-medoid algorithm. We use the overlap of the object masks as the distance metric (1 − intersection-over-union (IoU)). We separately choose the cluster size so that the average intra-cluster distance is roughly 0.25. This results in 1142 clusters across the 20 categories in PASCAL. The cluster sizes vary for different categories: categories with diverse shapes such as chairs and planes have more clusters, whereas rectangular objects such as buses and TV monitors have very few (<5) clusters. Any metric that captures certain aspects of object similarity can be used, such as a pre-defined class hierarchy, a metric induced by the object viewpoints and/or keypoints, or simply the Euclidean distance between *fc7* features. We visualize the complete hierarchy in Fig. 3, with up to 5 clusters for each leaf node. A background class associated with each leaf node is omitted for clarity.

Architecture: We implement our model in MatConvNet [45] using the VGG-16 architecture [41]. The shared feature extractor (*conv1* through *conv5*) and the first two levels, classifying object vs. non-object and recognizing semantic object categories, are the same as Faster-RCNN [37]. To build the third-level subcategory classifiers, we initialize from ImageNet-pretraining an additional set of *fc6* and *fc7* layers on top of the shared *conv5* feature maps. Under each semantic class, we build a subcategory node (20 of them in PASCAL) on top of the *fc7* features shared across the subcategory nodes. Each node consists of a classifier and a regressor. The classifier is trained with the softmax loss; the regressor is trained with smooth L_1 loss, as in Fast-RCNN [17]. The full architecture along with a recursive refinement module is shown in Figure 2.

Definition of positives and negatives: During training, each ground-truth box is assigned exclusively with a class label and a subclass label under this class. In Faster-RCNN, boxes with IoU ≥ 0.5 are assigned as positives and boxes with IoU $\in [0.1, 0.5)$ are assigned as negatives as an approximation for hard-negative mining. We observed that our model handles hard-negative mining by design (gradually pruning easy negatives at each level), and the 0.1 IoU threshold is no longer a good characterization for all sources of error across all nodes, so we relax the negatives to be all regions with IoU < 0.5. This also allows more negative samples to be passed to the subcategory nodes.

Sampling: During training, each node performs two types of sampling: (1) sampling a subset of the examples to compute the loss at the node (*loss sampling*); (2) sampling a subset of the examples to pass to all the child nodes (*pathway sampling*). In loss sampling, for all classifiers in the hierarchy, we independently sample 256 regions from the windows this classifier sees, with $50\%/50\%$ for positives and negatives. In pathway sampling, instead of tuning the thresholds t_i, we let each node pass the top-K_i scoring regions to its children. Since there is less and less training data as we go down the hierarchy, each node passes all the positive examples to its children. For negatives, we randomly sample 500 negative regions out of the top 2000 RPN proposals at the first level, and the top 300 negative regions at the second level. This allows enough negative examples to reach the leaf nodes to compute their losses. In practice, we find our model insensitive to the hyperparameters mentioned.

During testing, we sample the top 300 regions from RPN, and then the top 100 regions at the category node. To further save computation, we threshold at probability 0.01 before passing the regions to the leaves. This leads to a few or a few tens of proposals for each subcategory node. Again, we find our model insensitive to these hyperparameters. In the end, we multiply the conditional probabilities in Equation 1 to obtain the final detection scores, and finally perform per-category NMS on the detection boxes with overlap threshold 0.3.

Training: we use a batch size of 1, and a learning rate of 1×10^{-3} for 5 epochs followed by 1×10^{-4} for 2 epochs. We use the approximate joint training scheme, and weigh the classification and regression losses of all nodes equally.

4. Evaluation

4.1. Main results

To demonstrate our model's capability for general object understanding, we evaluate performance on the tasks of instance segmentation, part segmentation, and keypoint estimation. We use the PASCAL VOC dataset [13], and follow the instance segmentation protocol [20] to split the

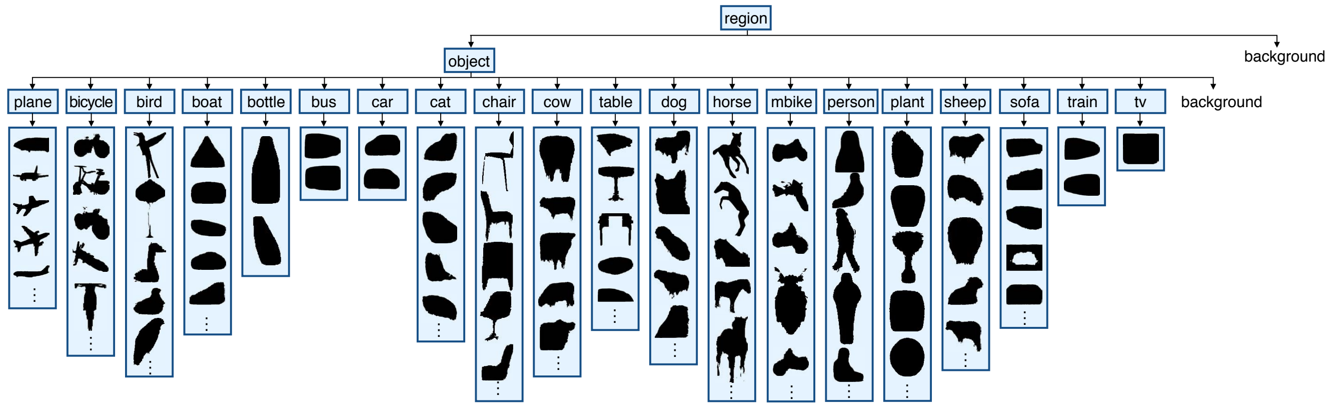


Figure 3: **Complete hierarchy:** We construct a three-level hierarchy, where the levels are objects vs. non-objects, semantic categories and subcategories, respectively. A background class is associated with each node. The background class associated with each leaf node is omitted for clarity.

data into 5623 *train* images and 5732 *val* images for all the experiments. We focus on the person category in PASCAL (where such rich annotations are common), though our approach applies to any category without modification. To generate such diverse outputs, we simply transfer annotations from the *train* subcategories (extracted from the mean of each cluster) to the *val* detections. While not state-of-the-art on any one task, *our single model achieves competitive results across all tasks, even when compared to specially-tuned models for each task.*

Instance segmentation: We evaluate instance segmentation across all PASCAL categories, reporting the mAP^r metric, the mean AP at the $0.5/0.7$ IoU thresholds. Results are in Table 1 and Fig. 4. Our model performs reasonably well compared to prior work, achieving mAP^r $54.5/19.5$ at the $0.5/0.7$ IoU thresholds. Since the average templates are coarse and cannot capture detailed shape variations, our model does not outperform the-state-of-the-art methods, particularly with a 0.7 threshold which emphasizes precise segmentation. However, we point out that our model takes a top-down approach without requiring any pixel-level computation. To this end, we significantly outperform the very recent work [23] that directly regresses objects to shapes. In addition, during training, we only use the IoU distance between masks as the supervision, without directly using the mask pixels themselves.

To show that our model benefits from low-level image cues, we conduct a simple experiment where we project our predicted binary masks to superpixels, following prior work [20, 21]. We observe an improvement of $+2.7/+9.3$, which is especially significant at the 0.7 threshold. Our model can likely benefit from additional post-processing techniques such as (sub)class-specific pixel-level classifiers [22], multi-scale processing [2] and multi-region pooling [16]. For example, concatenating multi-scale features from *conv4* and *conv5* brings an additional improvement of $+1.4/+0.9$.

method	mAP^r $(0.5/0.7)$
shape-regress† [23]	$34.6/15.0$
SDS [20]	$49.7/25.3$
Hypercolumn [21]	$60.0/40.4$
MNC [8]	$63.5/41.5$
Ours	$54.5/19.5$
Ours (superpixel)	$57.2/28.8$
Ours (superpixel + multiscale)	$58.6/29.7$

Table 1: **Instance segmentation:** We compare our method to prior work on the instance segmentation benchmark on the PASCAL *val* set. Our model is comparable to prior approaches specially tuned for this task when evaluated at the coarse overlap threshold, though not as competitive for the high threshold. Our model is not tailored specifically to the instance segmentation task and thus provides a more detailed analysis of the test object in a multi-task manner, as will be shown in the experiments on part segmentation and keypoint detection. † Reported on the 2857 SBD val images.

Finally, with the fine-grained subcategory predictions and the correspondence to object clusters, our model is not tailored specifically to the instance segmentation task and thus provides a more detailed analysis of the test object. This will be shown in the following experiments on part segmentation and keypoint detection.

Part segmentation: We present our results on part segmentation [7] on the person category of PASCAL. We follow the protocol in prior work [21], merging the annotations into head, torso, arms and legs, and evaluating each part using the instance segmentation mAP^r metric.

The results are in Fig. 5. Notably, we outperform the prior work significantly by $+11.2$ without even using the part annotations during training time. Our model correctly identifies good matches between the test objects and their corresponding training object clusters.

Keypoint detection: We present our results on keypoint localization by label transfer on the PASCAL-Person *val*

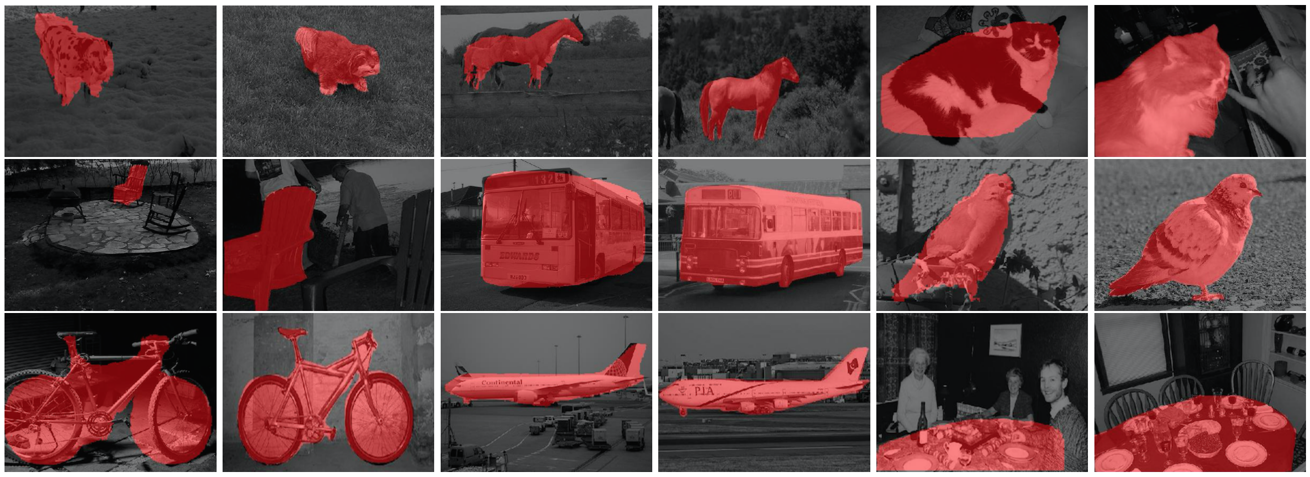
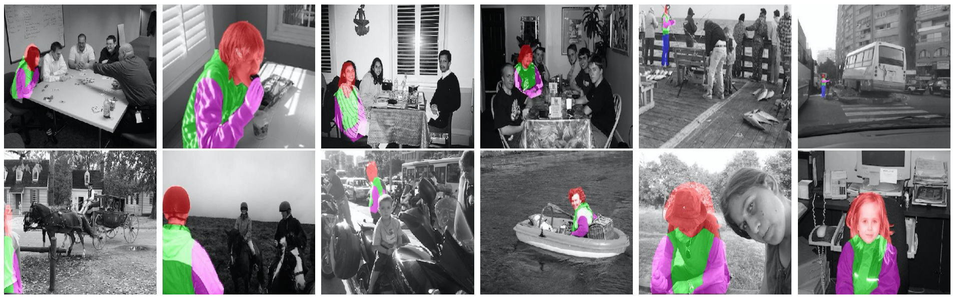


Figure 4: **Instance segmentation:** We visualize results on the PASCAL *val* set by label transfer, and the corresponding images from the *train* set. For each pair, the left image is in the *val* set; the right image is the corresponding cluster medoid from the *train* set. Fine boundaries are missing especially for the animal categories (first row), since the average masks are coarse and we do not use additional low-level appearance cues.



person	mAP^r (0.5/0.7)
Hypercolumn [21]	28.5/NA
Ours†	39.7/32.0

Figure 5: **Part segmentation:** We evaluate the label transfer results on the PASCAL Person *val* set. **Left:** For each pair, the left image is in the *val* set; the right image is the corresponding cluster medoid from the *train* set. **Right:** We compare our method to prior work on the part localization task. Our model outperforms the prior work, even without using part annotations during training time. †Evaluated on the 1805 out of 1818 images of the standard *part-val* that overlaps with *inst-val*. Trained on the 5623 images from *inst-train* without part annotations.

set. We consider the keypoint detection setting where the ground-truth bounding boxes are not given, and evaluate the APK metric [49]. The results are in Fig. 6. We perform comparably (0.19) to the prior work (0.22) [44]. Again, we do not use the keypoint annotations during training time but only during the label transfer phase.

4.2. Ablation studies

Recursive localization: We terminate the regression at different levels, and compare the detection and instance segmentation performance in Table 2. All the regressors contribute to better localization, where the last regressor especially focuses on accurate localization at high IoU thresholds. Since we obtain instance masks by directly warping the shape templates to the bounding boxes, an improvement in detection naturally leads to an improvement in instance segmentation.

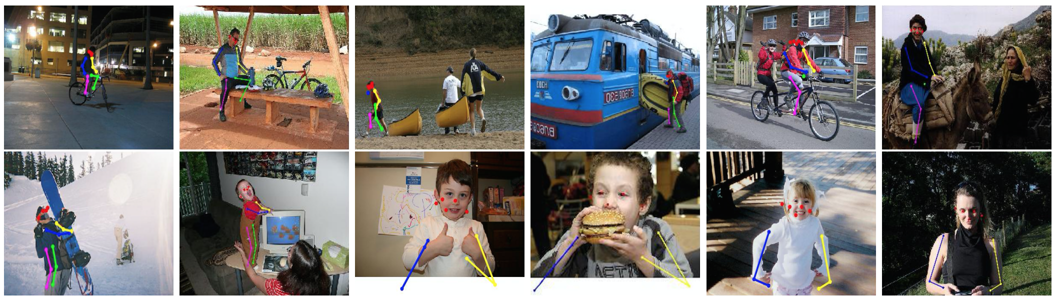
Hierarchical structure: We evaluate whether the hierarchical construction (Fig. 3) helps with better fine-grained subcategory classification. We compare our model to a two-level baseline (termed "flat"), where we remove the intermediate level and classify the object-like regions directly into fine-grained shapes tied to each specific object category.

The results on subcategory detection accuracy (detection AP averaged across all the subcategories, since the baseline model does not report categories anymore)[1] and

regressor			segment	detect
level1	level2	level3	mAP^r(0.5/0.7)	mAP(0.5/0.7)
✓			43.7/6.6	61.4/22.7
✓	✓		53.5/18.0	66.2/43.1
✓	✓	✓	**54.5/19.5**	**66.4/45.9**

Table 2: **Recursive localization:** We diagnose the effect of recursive localization at multiple levels of the hierarchy, on both semantic segmentation and detection accuracy. Past approaches (such as RCNN [18, 37]) make use of regressors at only 1 or 2 levels. Adding a third level leads to more precise localization and segmentation.

[1]We note that as the subcategories are defined by a clustering algorithm



person	APK
VP & KP [44]	0.22
Ours[†]	0.19

Figure 6: **Keypoint detection:** We evaluate the label transfer on PASCAL-Person-*val*. **Left:** For each pair, the left image is in the *val* set; the right image is the corresponding cluster medoid from the *train* set. **Right:** We compare our method to prior work on the keypoint detection task (without ground-truth bounding boxes) on the PASCAL person category. Our model performs comparably despite not using keypoint annotations at training time. [†] Evaluated on the 2082 out of 2093 images of the standard *kp-val* that overlaps with *inst-val*. Trained on the 5623 images from *inst-train* without keypoint annotations.

method	segment $\text{mAP}^r(0.5/0.7)$	detect $\text{mAP}^{sub}(0.5/0.7)$
flat	$49.6/15.1$	$5.4/4.1$
shared *fc6*+*fc7*	$53.4/18.8$	$6.4/5.1$
ours	$\mathbf{54.5/19.5}$	$\mathbf{6.7/5.7}$

Table 3: **Hierarchical construction:** Hierarchical models significantly outperform a flat classifier because hierarchical training naturally enforces hard-example mining (*i.e.* only hard examples from a parent node are selected for training its child nodes). Moreover, hierarchies allow us to learn specialized features for different nodes, which slightly outperforms a shared set of features.

cluster	shapes $\text{mAP}^r(0.5/0.7)$	parts $\text{mAP}^r(0.5/0.7)$	keypoints APK
avg	$48.3/14.0$	$24.2/16.2$	0.07
fc7	$48.1/14.3$	$35.5/27.5$	0.17
box	$46.2/13.6$	$31.4/23.6$	0.15
shape	$\mathbf{54.5/19.5}$	$\mathbf{39.7/32.0}$	0.19
kp	$47.6/14.0$	$37.2/29.2$	$\mathbf{0.21}$

Table 4: **Different hierarchies:** Our model is flexible with any clustering method to construct the hierarchy. All clustering methods are better than the *avg* baseline (one trivial cluster for each semantic class), even clustering by *fc7* features or the box dimensions that do not use any additional annotations. The closer the metric is to the task to be evaluated, the better the performance is.

the instance segmentation accuracy are in Table 4. Our model performs significantly better than the flat baseline. In particular, at $0.5/0.7$ IoU thresholds, our model achieves $54.5/19.5$ on the instance segmentation task, with an absolute improvement of $+5.0/+4.5$ compared to the flat baseline.

We reason that the advantage of the hierarchy comes from two aspects: (1) each classifier learns to solve a subproblem and sees a selective subset of the training data, so it is easier for the classifiers to specialize; (2) the features extracted from the shared *conv5* feature maps are separately computed for each level, so the features can simultaneously learn to adapt. For example, for the shape nodes, the features may focus on the lower-level details such as object boundaries. To separate these two factors, we conduct an additional experiment where we tie parameters across levels. Sharing both the *fc6* and *fc7* layers across the second and the third levels performs reasonably ($53.4/18.8$), showing that a hierarchy significantly outperforms a flat baseline even without additional parameters.

Clustering metrics: We explore different ways to define the subcategory clusters. We use the k-medoid algorithm to cluster objects of each class by (1) *fc7*: Euclidean distance on *fc7* features extracted from a Faster-RCNN detector trained on the same dataset; (2) *box*: IoU on the bounding box dimensions; (3) *shape*: pixel IoU on the shape masks; (4) *kp*: hamming distance on the keypoint visibility patterns (encoded as a binary vector); (5) *avg*: a baseline where all the objects in one category form one subcategory. In each setting, we generate roughly 1000 clusters in total, where each cluster has roughly the same intra-cluster distance.

The results are in Table 4. First, we observe that on part segmentation and keypoint detection, all of the clustering metrics perform better than the average baseline, including *fc7* and *box* which use no additional annotations during training. This shows the advantage of our hierarchical construction. Second, our model is more effective when more information is used during clustering. On parts and keypoints, *fc7* features (with Imagenet pre-training) performs better than using only the box coordinates. Using explicit annotations from the task of interest is the most effective – the shape clusters perform the best on instance segmentation, and the keypoint visibility clusters perform the best on keypoint detection. We believe that carefully combining different types of annotations along with the appearance features can further improve the performance.

and there may exist highly-similar clusters, the detection accuracy only serves as an intermediate metric.

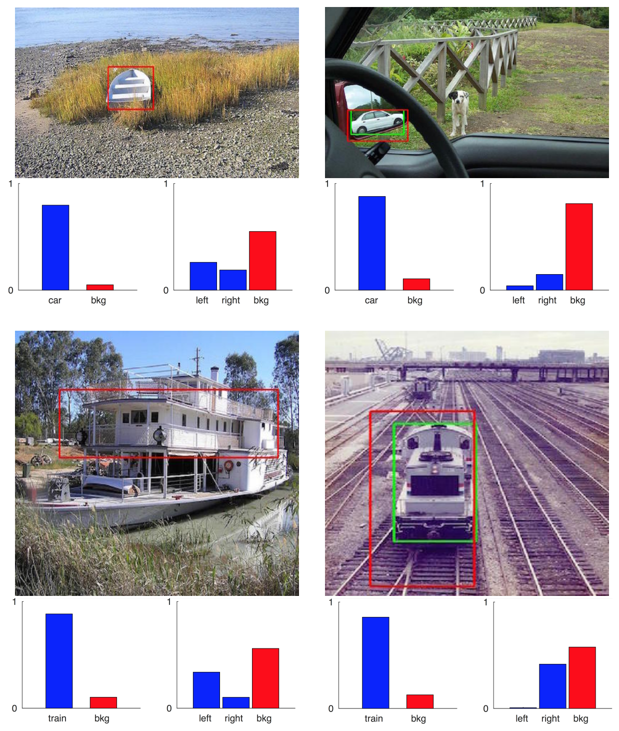


Figure 7: **Recursive rejection:** Some regions are originally classified as some foreground class at the category level, but rejected at the subcategory level. Under each image are two plots: the left plot shows the probability output at the category-level classifier; the right plot shows the probability output at the subcategory-level classifier. The classes of *bus, car* and *train* all have two shape subcategories each, denoted as "left" and "right". **Left column:** confusing categories are correctly rejected. **Right column:** examples novel in appearance, context, *etc.* are incorrectly rejected.

4.3. Analysis

Recursive rejection: In Fig. 7, we visualize some examples where the region is classified as a certain class at the category level, but classified as background (rejected) at the subcategory level. We observe two cases: (1) the category-level classifier makes mistakes between confusing categories, and these mistakes are corrected by the subcategory-level classifier; (2) the category-level classifier correctly identifies the category, but these predictions are incorrectly rejected by the subcategory-level classifier. These objects typically have novel appearance (*e.g.* the car appearing in the mirror and the train engine with no freights).

These observations suggest: (1) a feedback mechanism allowing each node to correct the predictions made by its parent will be beneficial; (2) our hierarchical model has the potential to provide partial information about unseen categories, and address "open-world" detections, where the model correctly detects an object, but realizes that its shape does not correspond to a previously-trained subclass.

Cascaded coarse-to-fine computation: Our model's multi-task and scalable nature comes from the efficiency of the hierarchy (though it does not directly improve the run time of Faster-RCNN or other special-purpose models): during training time, each node is trained if and only if there is positive training data for this node. During test time, each level of the hierarchy processes $\sim$10% of proposals from the previous level, and this leaves each subcategory-level node firing on $\sim$20% of the test images, with on average $\sim$30 category-specific regions per image when it fires.

Our recursive procedure is naturally coarse-to-fine and so can (in principle) also be used for *anytime* recognition. For example, early levels of the hierarchy immediately report which image regions contain an object or background, while later levels reveal object classes and their pose. Such sequential feedback maybe useful for say, an autonomous vehicle navigating in real-time. Such a perspective also suggests another criteria for designing a hierarchy; rather than maximizing recognition performance, maximize *anytime*-recognition performance.

Failure modes: We observe that, unsurprisingly, our model performs reasonably well on coarse estimation, but fails to capture all the fine details by naively transferring annotations, especially when objects exhibit atypical appearance. Our model works especially well on small objects, which inherits less appearance variations, and often are challenging for bottom-up approaches.

5. Conclusions

We present a generalized model for generic object understanding. In our model, a refinement module recursively develops understanding across space and semantic, by simultaneously localizing the object positions and classifying them into fine-grained subcategories. Our model, due to its recursive nature, is also efficient, selectively choosing a small subset of the regions to process in the next step. Our experiments on the PASCAL VOC dataset show promising results on the instance segmentation, part segmentation and keypoint detection tasks, where for the later two tasks the annotations are used only in the label transfer phase but not during training time. In addition, our recursive framework also provides a generalization of the design choices made by the Faster R-CNN detector.

Acknowledgements: This research was supported in part by the National Science Foundation (NSF) under grant IIS-1618903, the Defense Advanced Research Projects Agency (DARPA) under Contract No. HR001117C0051, and the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via IARPA R & D Contract No. D17PC00345. Additional support was provided by Google Research and the Intel Science and Technology Center for Visual Cloud Systems (ISTC-VCS). The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of ODNI, IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation thereon.

References

[1] E. Bart, M. Welling, and P. Perona. Unsupervised organization of image collections: taxonomies and beyond. *IEEE transactions on pattern analysis and machine intelligence*, 33(11):2302–2315, 2011. 2

[2] S. Bell, C. L. Zitnick, K. Bala, and R. Girshick. Inside-Outside Net: Detecting objects in context with skip pooling and recurrent neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 5

[3] S. Bengio, J. Weston, and D. Grangier. Label embedding trees for large multi-class tasks. In *Advances in Neural Information Processing Systems 23 (NIPS)*, pages 163–171, 2010. 2

[4] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001. 2

[5] J. C. Caicedo and S. Lazebnik. Active object localization with deep reinforcement learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2488–2496, 2015. 2

[6] R. Caruana. Multitask learning. In *Learning to learn*, pages 95–133. Springer, 1998. 1

[7] X. Chen, R. Mottaghi, X. Liu, S. Fidler, R. Urtasun, and A. Yuille. Detect what you can: Detecting and representing objects using holistic models and body parts. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 5

[8] J. Dai, K. He, and J. Sun. Instance-aware semantic segmentation via multi-task network cascades. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 5

[9] J. Deng, A. C. Berg, and L. Fei-Fei. Hierarchical semantic indexing for large scale image retrieval. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 785–792. IEEE, 2011. 2

[10] J. Deng, S. Satheesh, A. C. Berg, and F. Li. Fast and balanced: Efficient label tree learning for large scale object recognition. In *Advances in Neural Information Processing Systems 24 (NIPS)*, pages 567–575, 2011. 2

[11] S. K. Divvala. *Visual subcategories*. PhD thesis, Carnegie Mellon University, 2011. 2

[12] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. DeCAF: A deep convolutional activation feature for generic visual recognition. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2014. 2

[13] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88(2):303–338, 2010. 4

[14] P. F. Felzenszwalb, R. B. Girshick, and D. McAllester. Cascade object detection with deformable part models. In *Computer vision and pattern recognition (CVPR), 2010 IEEE conference on*, pages 2241–2248. IEEE, 2010. 2

[15] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE transactions on pattern analysis and machine intelligence*, 32(9):1627–1645, 2010. 2

[16] S. Gidaris and N. Komodakis. Object detection via a multi-region & semantic segmentation-aware CNN model. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2015. 5

[17] R. Girshick. Fast R-CNN. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2015. 4

[18] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 6

[19] A. Gonzalez-Garcia, A. Vezhnevets, and V. Ferrari. An active search strategy for efficient object class detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3022–3031, 2015. 2

[20] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik. Simultaneous detection and segmentation. In *European Conference on Computer Vision (ECCV)*, 2014. 4, 5

[21] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik. Hypercolumns for object segmentation and fine-grained localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 5, 6

[22] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask R-CNN. *Proceedings of the International Conference on Computer Vision (ICCV)*, 2017. 5

[23] S. Jetley, M. Sapienza, S. Golodetz, and P. H. Torr. Straight to shapes: Real-time detection of encoded shapes. *arXiv preprint arXiv:1611.07932*, 2016. 5

[24] I. Kokkinos. UberNet: Training a 'universal' convolutional neural network for low-, mid-, and high-level vision using diverse datasets and limited memory. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 1

[25] P. Kontschieder, M. Fiterau, A. Criminisi, and S. Rota Bulo. Deep neural decision forests. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1467–1475, 2015. 2, 3

[26] D. Kuettel and V. Ferrari. Figure-ground segmentation by transferring window masks. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 558–565. IEEE, 2012. 2

[27] C. H. Lampert, M. B. Blaschko, and T. Hofmann. Efficient subwindow search: A branch and bound framework for object localization. *IEEE transactions on pattern analysis and machine intelligence*, 31(12):2129–2142, 2009. 2

[28] T. Lan, M. Raptis, L. Sigal, and G. Mori. From subcategories to visual composites: A multi-level framework for object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 369–376, 2013. 2

[29] C. Liu, J. Yuen, and A. Torralba. Nonparametric scene parsing via label transfer. *Pattern Analysis and Machine Intelligence*, 33(12), 2011. 2

[30] Y. Lu, T. Javidi, and S. Lazebnik. Adaptive object detection using adjacency and zoom prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 2

[31] T. Malisiewicz, A. Gupta, and A. A. Efros. Ensemble of exemplar-SVMs for object detection and beyond. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 89–96. IEEE, 2011. 2

[32] C. B. Mervis and E. Rosch. Categorization of natural objects. *Annual review of psychology*, 32(1):89–115, 1981. 2

[33] G. A. Miller. WordNet: a lexical database for English. *Communications of the ACM*, 38(11):39–41, 1995. 2

[34] E. Ohn-Bar and M. M. Trivedi. Fast and robust object detection using visual subcategories. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 179–184, 2014. 2

[35] M. Pedersoli, A. Vedaldi, J. Gonzalez, and X. Roca. A coarse-to-fine approach for fast deformable object detection. *Pattern Recognition*, 48(5):1844–1853, 2015. 2

[36] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson. CNN features off-the-shelf: An astounding baseline for recognition. In *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 512–519, 2014. 2

[37] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015. 2, 4, 6

[38] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. Berg, and L. Fei-Fei. ImageNet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015. 2

[39] R. Salakhutdinov, J. B. Tenenbaum, and A. Torralba. Learning with hierarchical-deep models. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1958–1971, 2013. 2

[40] A. Shrivastava, A. Gupta, and R. Girshick. Training region-based object detectors with online hard example mining. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 761–769, 2016. 3

[41] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2015. 4

[42] E. B. Sudderth, A. Torralba, W. T. Freeman, and A. S. Willsky. Learning hierarchical models of scenes, objects, and parts. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 2, pages 1331–1338. IEEE, 2005. 2

[43] J. Tighe and S. Lazebnik. Finding things: Image parsing with regions and per-exemplar detectors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013. 2

[44] S. Tulsiani and J. Malik. Viewpoints and keypoints. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 6, 7

[45] A. Vedaldi and K. Lenc. MatConvNet – convolutional neural networks for MATLAB. In *Proceeding of the ACM Int. Conf. on Multimedia*, 2015. 4

[46] C. Vens, J. Struyf, L. Schietgat, S. Džeroski, and H. Blockeel. Decision trees for hierarchical multi-label classification. *Machine Learning*, 73(2):185, 2008. 2

[47] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages I–I. IEEE, 2001. 2

[48] Y. Xiang, W. Choi, Y. Lin, and S. Savarese. Subcategory-aware convolutional neural networks for object proposals and detection. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2017. 2

[49] Y. Yang and D. Ramanan. Articulated human detection with flexible mixtures of parts. *IEEE transactions on pattern analysis and machine intelligence*, 35(12):2878–2890, 2013. 6

[50] T. Yeh, J. J. Lee, and T. Darrell. Fast concurrent object localization and recognition. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 280–287. IEEE, 2009. 2