Supplementary material: 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

1. Evaluation

1.1. Average template

We first describe how we compute the average template for each subcategory cluster, used in our label transfer step.

Instance segmentation: For each subcategory, we compute a binary average template from all the training examples within the cluster, by warping their ground-truth masks to a fixed size and majority-voting at each pixel.

Part segmentation: To compute the cluster mean for parts, we first use the binary shape masks in the *train* set to majority-vote for each pixel and obtain an average binary mask. Then, for each foreground pixel in the average mask, we use the part segmentation in the *train* set to majority-vote for the part label.

Keypoint detection: We compute an average template for each keypoint in each cluster, by normalizing the keypoint coordinates to $[0, 1]$ with respect to the bounding box size. We note that the APK benchmark on the PASCAL VOC dataset treats invisible keypoints as negatives and penalizes those predictions. To mitigate this issue, for each cluster, we naively compute a prior which is the probability of each keypoint appearing in the cluster. We use the product of this visibility prior and the detection score as the score for each keypoint.

1.2. Cluster size

When constructing the clusters, one natural question would be: "how many clusters do we need, and how is the performance sensitive to the number of clusters?" In Fig. 1, we experiment with different cluster sizes and plot their instance segmentation performance. We observe that even a very small cluster size ($\sim$50 clusters across 20 categories) performs well on the coarse 0.5 threshold, but on the more precise 0.7 threshold, our model requires $\sim$1000-3000 clusters in total to perform well. At the rightmost point in the plot, we push our clusters to exemplars so that each object forms its own subcategory. The performance drops drastically due to severe overfitting. We believe that carefully incorporating regularization into the loss may alleviate this issue.

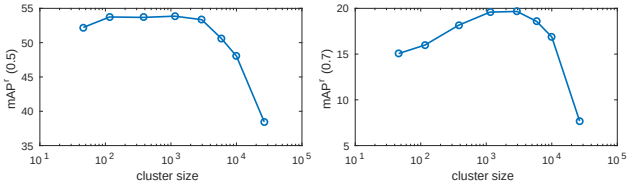

Figure 1: **Cluster size:** On the instance segmentation task, our model requires about 50 clusters to perform well on the coarse 0.5 threshold, and about 1000-3000 clusters to perform well on the precise 0.7 threshold.

method	mAP$^r(0.5)$	mAP$^r(0.7)$
LB	47.8	13.4
Ours	54.5	19.5
UB	63.7	26.5

Table 1: **Instance segmentation:** We compare our model to a lower bound (using the mean object mask for each object category) and an upper bound (choosing the best subcategory by an oracle). The gap between the lower bound and our model shows the gain from subcategory clusters; the gap between our model and the upper bound shows the space for improvement with better subcategory classification; the upper bound shows the limit of the subcategory clusters that we used.

2. Error analysis

We conclude that our model is primarily limited by the inflexibility of the clusters to represent fine variations of the shapes. For this reason, the deformable categories are especially challenging.

2.1. Instance segmentation

To understand the gain and the limit of our clustering approach, we conduct the following experiment: we use the bounding boxes and the prediction scores from our model, but replace the label transfer by (1) *lower-bound:* a binarized average mask for each category is transferred to each

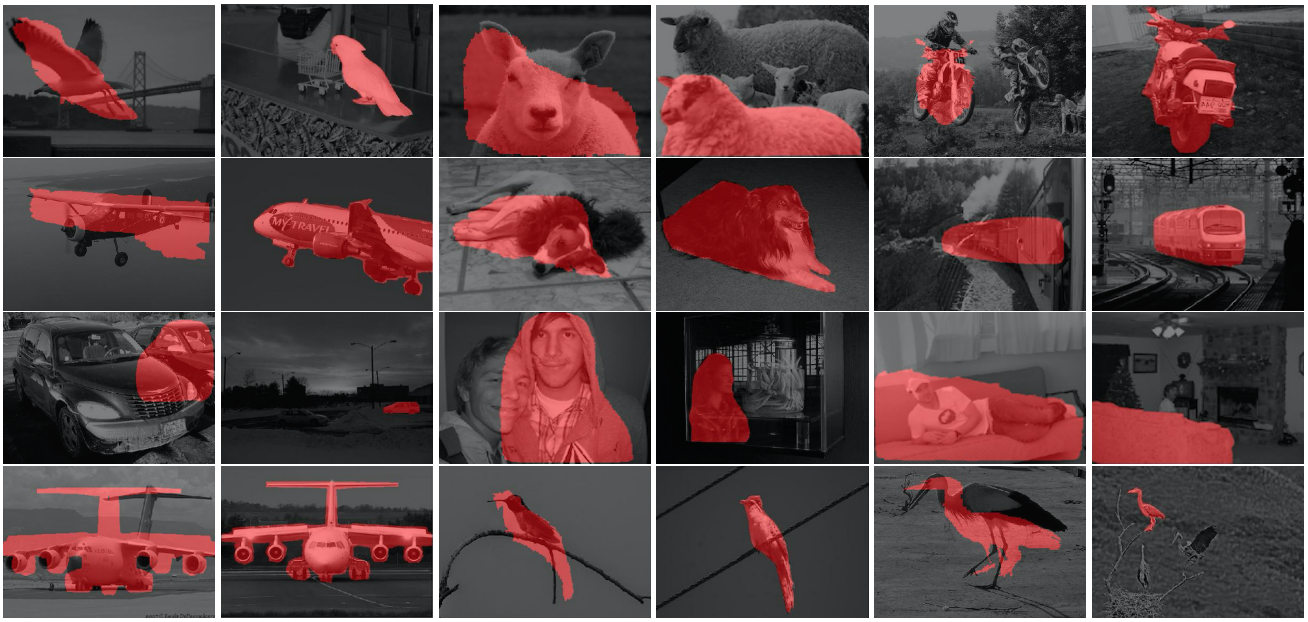

Figure 2: **Failure cases (instance segmentation):** We focus on the errors given correct box predictions, and visualize 4 failure modes. **Row 1:** matches incorrectly identified for common objects. **Row 2:** matches incorrectly identified for uncommon objects. **Row 3:** lack of occlusion handling. **Row 4:** matches correctly identified, but still inaccurate.

Figure 3: **Failure cases (part segmentation):** We focus on the errors given correct box predictions, and visualize 3 failure cases. **Row 1:** matches incorrectly identified for common poses. **Row 2:** test images with uncommon poses. **Row 3:** training images with uncommon poses.

detection box; (2) *upper-bound:* an oracle provides the best subcategory class of the highest IoU. The results are in Table 1. Comparing to the lower-bound, our model gives a significant gain by dividing each category into subcategories. The upper-bound shows the limit of naive label transfer: the collection of shapes during training does not sufficiently cover all possible shapes during test time, especially on the fine details. The test objects sometimes cannot find close shape subcategories, especially at high IoU threshold.

In Fig. 2, we visualize some failure cases. We first note that the naive label transfer approach suffers from detection errors, including class confusion, mislocalization and false positives. Here, we primarily focus on the cases where the box prediction is accurate but the segmentation mask is inaccurate. We visualize 4 types of errors: *Row 1*: the matches are incorrectly identified for common objects (the pair of *train* and *val* objects exhibit different shapes, poses, *etc.*); *Row 2*: the matches are incorrectly identified for uncommon shapes (antique airplane, dog lying on the side, *etc.*); *Row 3*: occlusion in either the training or the test image is not properly handled; *Row 4*: the matches are correctly identified, but the transferred masks are just not accu-

Figure 4: **Failure cases (keypoint detection):** We focus on errors given correct box predictions, and visualize 3 failure cases. **Row 1:** matches incorrectly identified for common poses. **Row 2:** test images with uncommon poses. **Row 3:** training images with uncommon poses.

rate enough.

2.2. Part segmentation

In Fig. 3, we visualize 3 types of errors for part segmentation: *Row 1:* the matches are incorrectly identified for common poses in the test images; *Row 2:* the matches are incorrectly identified for uncommon poses in the test images; *Row 3:* training images with uncommon poses "hallucinate" non-existing structures to the test images. In addition, the coarse nature of label transfer is another source of error.

2.3. Keypoint detection

In Fig. 4, we visualize the same 3 types of errors for keypoint detection. The coarse nature of label transfer is another source of error as before: our model almost never precisely predicts the locations of the eyes, ears and nose, marked by red dots in Fig. 4.